AFRL-OSR-VA-TR-2015-0012

MACHINE LEARNING CONTROL FOR HIGHLY RECONFIGURABLE HIGH-ORDER SYSTEMS

John Valasek
TEXAS ENGINEERING EXPERIMENT STATION COLLEGE STATION

01/02/2015
Final Report

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/ RTA
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
27-12-2014	Final Report	01-07-2011 to 29-09-2014

4. TITLE AND SUBTITLE

Machine Learning Control For Highly Reconfigurable High-Order Systems

5a. CONTRACT NUMBER
FA9550-11-1-0302

5b. GRANT NUMBER
FA9550-11-1-0302

5c. PROGRAM ELEMENT NUMBER
N/A

6. AUTHOR(S)

John Valasek and Suman Chakravorty

5d. PROJECT NUMBER
N/A

5e. TASK NUMBER
N/A

5f. WORK UNIT NUMBER
N/A

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Texas A&M Engineering Experiment Station
3470 TAMU
College Station, TX 77843-3577
(979) 458-7643

8. PERFORMING ORGANIZATION REPORT NUMBER

C12-00097

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Office of Science and Research
875 Randolph Street
Suite 325 Room 3112
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)

AFOSR/RTA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

N/A

12. DISTRIBUTION / AVAILABILITY STATEMENT

Distribution Unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This Grant addressed four focus areas toward advancing the state-of-the-art in learning control theory of robust and adaptive non-equilibrium control of highly nonlinear, higher-order, reconfigurable systems: 1. Extend Approximate Dynamic Programming (ADP) techniques to control of nonlinear, multiple time scale, non-affine systems in an Adaptive Control framework.; 2. Develop solution techniques for Markov Decision Problems (MDP) that scale to continuous state and control spaces with constraints; 3. Extend MDP techniques to solve multi-agent co-ordination and control problems in a decentralized fashion. 4. Develop solution techniques that scale to continuous state-space Partially Observable Markov Decision Problems (POMDP) and their multi-agent generalizations. The work produced the first significant results in the nonlinear control of multiple time-scale control in the last 25 years, and additionally made significant contributions to the analysis and control of systems that are non-affine in control, and non-minimum phase. The work also developed sampling based feedback planning techniques for the solution of Markov Decision Problems (MDP) and Partially Observed MDPs (POMDP).

15. SUBJECT TERMS

Approximate Dynamic Programming, Singular Perturbations, Nonlinear Control, Time-Scale Systems, Non-Affine, Non-Minimum

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			John Valasek
UU	UU	UU	UU	262	**19b. TELEPHONE NUMBER** *(include area code)* 979 - 845 - 1685

Final Technical Report

MACHINE LEARNING CONTROL OF NONLINEAR, HIGH-DIMENSIONAL RECONFIGURABLE SYSTEMS

FA9550-11-1-0302

Period of Performance 1 July 2011 – 29 September 2014

John Valasek
Aerospace Engineering Department
Texas A&M University

Suman Chakravorty
Aerospace Engineering Department
Texas A&M University

Executive Summary

During the course of this project we made significant progress in four focus areas toward advancing the state-of-the-art in learning control theory of robust and adaptive non-equilibrium control of highly nonlinear, higher-order, reconfigurable systems:

1. Extend Approximate Dynamic Programming (ADP) techniques to control of nonlinear, multiple time scale, non-affine systems in an Adaptive Control framework.
2. Develop solution techniques for Markov Decision Problems (MDP) that scale to continuous state and control spaces with constraints.
3. Extend MDP techniques to solve multi-agent co-ordination and control problems in a decentralized fashion.
4. Develop solution techniques that scale to continuous state-space Partially Observable Markov Decision Problems (POMDP) and their multi-agent generalizations.

Progress Reporting

Our work is presented as journal and conference papers, with an introduction to each paper and the actual paper included below. The work conducted by John Valasek and his students is presented first, followed by the work conducted by Suman Chakravorty and his students.

Acknowledgement/Disclaimer

This work was sponsored (in part) by the U.S. Air Force Office of Scientific Research under contract FA9550-08-1-0038. The views and finding contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Air Force Office of Scientific Research or the U. S. Government.

Personnel Supported During Duration of Grant

John Valasek	PI, Professor, Texas A&M University
Suman Chakravorty	Co-PI, Associate Professor, Texas A&M University
Anshu Siddarth	Ph.D. Student, Texas A&M University Graduated Fall 2012, now Assistant Professor, University of Washington, Seattle.
Aliakbar Aghamohammadi	Ph.D. Student, Texas A&M University, now Post-Doctoral Research Associate, MIT
Kenton Kirkpatrick	Ph.D. Student, Texas A&M University, now Member of the Technical Staff, United Space Alliance.
Elizabeth Rollins	Ph.D. Student, Texas A&M University, now Engineering Educator for Controls & Aerodynamics, Wichita State University.
Dipanjan Saha	Ph.D. Student, Texas A&M University
Saurav Agarwal	Ph.D. Student, Texas A&M University

Books and Book Chapters

Narang-Siddarth, Anshu, and Valasek, John, "**Nonlinear Multiple Time Scale Systems in Standard and Non-Standard Forms: Analysis and Control**," Society for Industrial and Applied Mathematics (SIAM), 2014.

Valasek, John (Ed.), "**Morphing Aerospace Vehicles and Structures**," John Wiley and Sons, Chichester, UK, 2012.

Valasek, John (Ed.), "**Advances in Intelligent and Autonomous Aerospace Systems**," AIAA Progress in Aeronautics & Astronautics Series, American Institute of Aeronautics and Astronautics, Reston, VA, 2012.

Lewis, Frank L., and Liu, Derong (Eds.), "**Reinforcement Learning and Approximate Dynamic Programming for Feedback Control**," John Wiley and Sons, 2012.

Chapter: Kirkpatrick, Kenton, and Valasek, John, "Reinforcement Learning Control With Time Dependent Agent Dynamics"

Honors & Awards Received

ASEE/AIAA John Leland Atwood Award, 2014, John Valasek

Texas A&M University Award for Outstanding Accomplishment in Research, Doctoral Level, 2012, Anshu Narang-Siddarth

Aerospace Engineering Department Award for Doctoral Research, 2012, Anshu Narang-Siddarth

Herbert H. Richardson Faculty Fellow for 2011 – 2012, John Valasek

AFRL Points of Contact
Siva Banda, AFRL/ WPAFB, OH
David Doman, AFRL/ WPAFB, OH
Michael Bolender, AFRL/ WPAFB, OH
Jonathan Muse, AFRL/ WPAFB, OH
Richard Erwin, AFRL/ RV, KAFB, NM

Transitions and New Discoveries
We have been awarded a Phase IV contract from the Raytheon Company, Intelligence and Information Systems Division to develop and flight test a Reinforcement Learning based approach for autonomous tracking of ground targets using a fixed wing Unmanned Air System (UAS). The capability of autonomously controlling both a UAS and an onboard motion video system to track a selected target can free a human operator to select viable targets, analyze images received, and prosecute mission objectives, rather than having to guide the UAS.

Point of Contact: Michael R. Moan, 972-205-8318, Michael_R_Moan@raytheon.com

TECHNICAL SUMMARIES – JOHN VALASEK

2011:

1. Siddarth, Anshu and John Valasek. 2011. **Kinetic State Tracking for a Class of Singularly Perturbed Systems**. *Journal of Guidance, Control, and Dynamics*, Vol. 34, No.3, pp. 734-749.

   The benefit of singular perturbation theory, specifically the Tikhonov theorem in flight control is at the level of modeling where it is used as a model reduction technique. This paper considers a class of nonlinear flight control problems that do not satisfy the underlying conditions of the Tikhonov theorem. Using insights from geometric singular perturbation theory and concept of center manifolds, this paper formulates a modified composite control law scheme that retains the benefits of the Tickhonov theorem for this general class of nonlinear problems. The main contribution of the developed result is that it is independent of the time scale separation between the translational and rotational dynamics.

2. Siddarth, Anshu and Valasek, John. 2011. **Global Tracking Control Structures for Nonlinear Singularly Perturbed Aircraft Systems,** in Advances in Aerospace Guidance, Navigation & Control, Florian Holzapfel and Stephan Theil, Eds, Springer Berlin Heidelberg, DOI: 10.1007/978-3-642-19817-5 19, pp 235-246.

   This paper is concerned with guaranteeing asymptotic tracking of both slow and fast dynamics of a nonlinear system – applications of which provides departure resistance capability for unmanned aerial vehicles. The synthesized controller takes advantage of singular perturbation theory and feedback linearization results to establish global exponential stability. The performance is demonstrated for a nonlinear, coupled, six degree-of-freedom F/A-18.

3. Siddarth, Anshu and Valasek, John. 2011. **Output Tracking of Non-Minimum Phase Systems**. AIAA-2011-6487, Proceedings of the 2011 AIAA Guidance, Navigation, and Control Conference, Portland, Oregon, 9 August 2011.

   The main challenge in synthesizing a controller for a tail-controlled vehicle is to achieve maximum performance it is capable of, without exciting the unstable internal dynamics. This paper revisits the problem and synthesizes a globally stable full-state feedback controller for desired output tracking. No modification to the output/or placement of a virtual inertial measurement unit is placed. The underlying approach is to induce a time scale separation between the output dynamics and the unstable internal dynamics. The main contributions of the synthesis are analytic conditions under which the time scale separation can be induced, and stability and robustness of the controller guaranteed.

2012:

4. Narang-Siddarth, Anshu and Valasek, John. 2012. **Tracking Control Design for Non-Standard Nonlinear Singularly Perturbed Systems**. WeA06.6, Proceedings of the 2012 American Control Conference, Montreal, Canada, 27 June 2012.

   In this paper novel state-feedback control laws are developed for a general class of two time scale continuous time systems which are nonlinear in both slow and fast states. No assumption concerning the type of non-linearity of the system is made and the technique is applicable to both standard and non-standard forms of singularly perturbed systems. Non-standard forms of singularly perturbed systems violate fundamental conditions for Tikchonov theorem – hindering use of model reduction for control synthesis. Asymptotic stabilization of the proposed 'indirect manifold construction approach' is proven using Lyapunov methods. Results show that the proposed technique applies both to standard and non-standard forms and guarantees asymptotic stabilization with quantifiable robustness bound for the singular perturbation parameter.

5. Valasek, John, Kirkpatrick, Kenton, and May, James, "**Intelligent Motion Video Guidance for Unmanned Air System Ground Target Surveillance**," AIAA 2012-2587, Proceedings of the 2012 AIAA Infotech@Aerospace Conference, Garden Grove, CA 21 June 2012.

   This paper develops an algorithm for surveillance of ground targets by UAS with fixed pan and tilt cameras, in the presence of winds. The specific Reinforcement Learning algorithm used is Q-learning, and the objective of the approach is to bring any target located in an image captured by a camera into the center of the image using the learned control policy. The learning agent determines offline (initially) how to control the UAS and camera to get a target from any point in the image to the center and hold it there. A feature of this approach is that the learning agent will continue to learn and refine and update the previously offline learned control policy, during actual operation.

6. Dunn, Caroline, Kirkpatrick, Kenton, and Valasek, John, "**Unmanned Air System Search and Localization Guidance Using Reinforcement Learning**," AIAA 2012-2589, Proceedings of the 2012 AIAA Infotech@Aerospace Conference, Garden Grove, CA 21 June 2012.

   This paper investigates aircraft flight path guidance for search and localization of Regions of Interest, consisting of atmospheric phenomena. The problem is posed as an offline agent learning problem, of localizing atmospheric thermal locations and then guiding an Unmanned Air Vehicle to soar from one to another. Q-learning is used as the learning algorithm. The computational navigation solution used here is a basic grid algorithm that assigns thermal locations and intensities, with the representation being specified states, actions, goals, and rewards that are used to accomplish the agent learning.

7. Narang-Siddarth, Anshu and Valasek, John. 2012. **Tracking Control for a Non-Minimum Phase Autonomous Helicopter**. AIAA 2012-4453, Proceedings of the 2012 AIAA Guidance, Navigation & Control Conference. Minneapolis, MN, 13 August 2012.

   This paper presents an application of authors' prior work on control of singularly perturbed systems (2012, American Control Conference) to an autonomous helicopter to mitigate non-minimum phase behavior. Fundamentally, this behavior is the technical term for the helicopter's inherent nature to induce instability when commanded to hover over a specific point in space. This research identifies that non-minimum phase behavior is actually related to non-standard singularly perturbed systems, and stable control is only possible if the helicopter exhibits standard singularly perturbed behavior. This key inference allows implementation of the 'indirect manifold construction approach' control scheme, which commands desired control action to perform the required transformation. The novel aspect of this implementation is that it provides a mechanism for feedback gain selection in nonlinear settings.

8. Valasek, John, Akella, Maruthi R., Siddarth, Anshu, and Rollins, Elizabeth. 2012. **Adaptive Dynamic Inversion Control of Linear Plants with Control Position Constraints**. *IEEE Transactions on Control Systems Technology* , Vol.20, No.4, pp 918-933.

   This paper is concerned with designing control actions for a vehicle to follow a prescribed guidance solution under design constraints. To fully take advantage of the design constraints, the proposed switching control action gives priority to safety over guidance when the vehicle is close to instability. This switching action, however, has been known to induce instability when implemented incorrectly. To avoid this the paper introduces two novel concepts: 'domain of control authority' and 'direction consistent control mechanism' that together identify boundary of control actions that always guarantee safety and required tracking and always restrict the commanded control action within this boundary. Controller performance is demonstrated with numerical examples of a two degree-of-freedom dynamic model and an F-16XL aircraft model.

2013:

9. Kirkpatrick, Kenton, and Valasek, John, "**Approximation of Agent Dynamics Using Reinforcement Learning**," AIAA 2013-0875, Proceedings of the 51st AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, Grapevine, TX, 9 January 2013.

   In this paper, Reinforcement Learning-based algorithms are developed for learning agents' time dependent dynamics while also learning to control them. Three algorithms are introduced. Sampled-Data Q-learning is an algorithm that learns the optimal sample time for controlling an agent without a prior model. First-Order Dynamics Learning is an algorithm that determines the proper time

constants for agents known to have first-order dynamics, while Second-Order Dynamics Learning is an algorithm for learning natural frequencies and damping ratios of second-order systems.

10. Narang-Siddarth, Anshu and Valasek, John. 2013. **A Constructive Stabilization Approach for Open-Loop Unstable Non-Affine Systems**. Proceedings of the 2013 American Control Conference, Washington D.C, 19 June 2013.

    This paper pursues to find a constructive feedback control strategy to address the inherent nonlinearities and complexities of underlying physical systems that form the basis of today's heterogeneous applications (for example: flapping-wing systems). Dynamics of such systems are nonlinear in the control input, and are more commonly referred as control-nonaffine systems in the literature. This special structure however violates collinearity; a key property essential to employ nonlinear control techniques such as feedback linearization, backstepping and Lyapunov-based redesign methods. The main contribution of this article is that it formulates a generalization of the famous Kalman-Yakubovich-Popov lemma for non-affine systems under mild restrictions. This new result helps to determine whether or not an input-output description of a nonlinear system is passive. It is expected that this generalization will play a vital role in developing adaptive control laws for nonlinear systems based on Lyapunov's direct method analogous to its linear counterpart.

11. Narang-Siddarth, Anshu and Valasek, John. 2013. **Necessary Conditions for Feedback Passivation of Nonaffine-in-Control Systems**. Proceedings of the 2013 SIAM Conference on Control & Its Applications.

    This research explores a universal stabilization formula for an unstable non-affine system by developing a novel composite control law structure. The intuitive idea behind this control form is to introduce stiffness and damping into the system. The major contribution comes in identifying principles that converts an open-loop unstable system into stable in the Lyapunov sense closed-loop system through static-feedback alone. Most importantly, conditions under which a nonlinear system be rendered passive through static state-feedback without making any assumptions about the nature of the control influence are developed in this paper.

12. Rollins, Elizabeth, Valasek, John, Muse, Jonathan, and Bolender, Michael, "**Nonlinear Adaptive Dynamic Inversion Applied to a Generic Hypersonic Vehicle**," AIAA 2012-5234, Proceedings of the 2013 AIAA Guidance, Navigation, and Control Conference, Boston, MA, 22 August 2013.

    Previous work on control design for hypersonic vehicles often uses linearized or simplified nonlinear dynamical models of the vehicle, and very little work has been done on recovering from unstart events. Using a generic hypersonic vehicle as a control design and simulation model, this paper develops a nonlinear adaptive dynamic inversion control architecture with a control allocation scheme to track

realistic flight path angle trajectories. A robustness analysis is performed on the initial control architecture design, which shows that the control architecture is able to handle time delays, perturbations in stability derivatives, and reduced control surface effectiveness. The control architecture then is evaluated for its ability to handle inlet unstart.

13. Henrickson, James, Kirkpatrick, Kenton, and Valasek, John, "**Rapid Characterization of Shape Memory Alloy Material Parameters Using Computational Intelligence Methods**," SMASIS2013-3016, Proceedings of the 15th ASME 2013 Conference on Smart Materials, Adaptive Structures and Intelligent Systems, Snowbird, UT, 16 September 2013.

    Shape memory alloys are capable of delivering advantageous solutions to a wide range of engineering-based problems. Implementation of these solutions, however, is often complicated by the hysteretic, non-linear, thermo-mechanical behavior of the material. Although existing shape memory alloy constitutive models are largely accurate in describing this unique behavior, they require prior characterization of the material parameters. Consequently, before thorough modeling and simulation can occur for a shape memory alloy-based project, one must first go through the process of identifying several material parameters unique to shape memory alloys. Current characterization procedures necessitate extensive experimentation, data collection, and data processing. As a result, these methods simultaneously create a high barrier of entry for engineers new to active materials and impede the advanced study of shape memory alloy material parameter evolution. This paper develops a novel method in which computational intelligence methods are used to rapidly identify shape memory alloy material parameters. Specifically, an artificial neural network is trained to identify transformation temperatures and stress influence coefficients of given shape memory alloy specimens using strain-temperature coordinates as inputs. After generating training data through the use of a constitutive model, the resulting trained artificial neural network was used to identify parameters for a number of randomly generated theoretical shape memory alloys. Results show that the artificial neural network was able to rapidly identify both transformation temperatures and stress influence coefficients with satisfactory accuracy. The generation of training data was then repeated using Taguchi methods. Further results show that the artificial neural network trained with the Taguchi-based training data yielded improved characterization accuracy while using less training data.

2014:

14. Woodbury, Timothy, Dunn, Caroline, and Valasek, John, "**Autonomous Soaring Using Reinforcement Learning for Trajectory Generation**," AIAA-2014-0990, Proceedings of the AIAA Science and Technology Forum and Exposition 2014: 52nd Aerospace Sciences Meeting, National Harbor, MD, 15 January 2014.

This paper develops an approach for planar lateral/directional guidance of a linear dynamic gliding aircraft to a known thermal location. Reinforcement learning is utilized to generate reference bank angle commands for directing the aircraft to close proximity of the updraft, and from there the aircraft follows a circling trajectory centered on the thermal to gain energy. A Lyapunov-based feedback control law is used to generate bank angle commands when circling the thermal. By using reinforcement learning the problem of online trajectory generation is reduced to a simple search in a static state-action value table. This approach has the advantage of low computational burden/overhead in practice. Furthermore, the need for a precise aircraft model for learning and simulation is reduced. Monte Carlo results presented in the paper demonstrate that the reinforcement learning guidance agent can consistently navigate the aircraft to the thermal. Reliable navigation is achieved after a relatively small number of learning episodes. An analysis of typical energy gains circling a thermal of constant shape and size is also presented.

15. Siddarth, Anshu, Peter, Florian, Holzapfel, Florian, and Valasek, John, "**Autopilot for a Nonlinear Non-Minimum Phase Tail-Controlled Missile Using Time Scale Separation**," AIAA-2014-1293, Proceedings of the AIAA Science and Technology Forum and Exposition 2014: 52nd Aerospace Sciences Meeting, National Harbor, MD, 16 January 2014.

    Acceleration control of highly agile, aerodynamically-controlled missiles is a well-known non-minimum phase control problem. This problem is revisited here for a planar tail-controlled generic missile, and a globally stable nonlinear autopilot command structure is synthesized to maximize performance. For the first time the non-minimum phase characteristics of the vehicle are addressed by making no modification to the output definition by inducing an inherent time scale separation in the closed-loop dynamics. Unlike previous time scale control techniques, results presented here are based on theoretical advancements made in control of nonlinear singularly perturbed systems. Conditions under which the induced time scale separation can be employed for a stable autopilot design are also discussed. The state feedback controller proposed is real-time implementable, independent of operating condition and desired output trajectory. Simulation results show that the approach is able to accomplish perfect tracking while keeping all closed-loop signals bounded.

JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 34, No. 3, May–June 2011

Kinetic State Tracking for a Class of Singularly Perturbed Systems

Anshu Siddarth and John Valasek
Texas A&M University, College Station, Texas 77843-3141

DOI: 10.2514/1.52127

The trajectory-following control problem for a general class of nonlinear multi-input/multi-output two time-scale system is revisited. While most earlier works used singular perturbation theory and assumed that an isolated real root exists for the nonlinear set of algebraic equations that constitute the slow subsystem, here, two time-scale systems are analyzed in the context of integral manifolds. It is shown that the singularly perturbed system has a center manifold and, for small values of the slow state, an approximate solution of the nonlinear set of transcendental equations can be computed. Geometric singular perturbation theory is used as the model-reduction technique, and modified composite control design is used to formulate the stabilizing control laws for slow state tracking. The control laws are independent of the scalar perturbation parameter and an upper bound for it, and the closed-loop error signals are determined such that uniform boundedness of the closed-loop system is guaranteed. Additionally, asymptotic stabilization is shown for the nonlinear regulation problem. The methodology is demonstrated through numerical simulation of a nonlinear generic two-degree-of-freedom kinetic model and a nonlinear, coupled, six-degree-of-freedom model of the F/A-18A Hornet. Results demonstrate that the methodology permits close tracking of a reference trajectory while maintaining all control signals within specified bounds.

Nomenclature

A, A_f	=	positive gain matrices
b	=	wingspan, ft
c	=	mean aerodynamic chord, ft
C_D	=	drag coefficient
C_L	=	lift coefficient
C_Y	=	side force coefficient
C_l, C_m, C_n	=	roll, pitch, and yaw moment coefficients
D	=	domain of subscripted variable
g	=	gravity acceleration, ft/s^2
I_x, I_y, I_z	=	principal axis inertias for aircraft, slug ft^2
M	=	Mach number
m	=	mass, slug
m	=	number of slow variables
$(M\phi)$	=	nonlinear map
$\mathcal{M}_\epsilon$	=	invariant manifold of full-order system
$\mathcal{M}_0$	=	invariant manifold of reduced-order subsystems
n	=	number of fast variables
$\mathcal{O}()$	=	order symbol

Anshu Siddarth is a Graduate Research Assistant and Ph.D. Candidate in the Aerospace Engineering Department at Texas A&M University. Her current research interests include adaptive control, nonlinear control, and singularly perturbed systems. She was a recipient of the Academic Proficiency Award in 2006 and the Tata Consultancy Services-Indian Institute of Technology, Madras (IITM) best-in-class fellowship in 2007. Anshu earned a B.Tech. in electrical and electronics engineering (with honors) from Jawaharlal Nehru Technological University, Hyderabad, in 2006, and a M.S. in aerospace engineering from the IITM in 2008. She is a Member of AIAA.

John Valasek is Director of the Vehicle Systems and Control Laboratory and a Professor of Aerospace Engineering at Texas A&M University. His research focuses on bridging the gap between computer science and aerospace engineering, encompassing machine learning and multiagent systems, intelligent autonomous control, vision-based navigation systems, fault tolerant adaptive control, and cockpit systems and displays. John was previously a Flight Control Engineer for the Northrop Corporation, Aircraft Division, where he worked in the Flight Controls Research Group and on the AGM-137 Tri-Services Standoff Attack Missile program. He was also a Summer Faculty Researcher at NASA Langley Research Center in 1996 and a U.S. Air Force Office of Scientific Research Summer Faculty Research Fellow in the Air Vehicles Directorate, U.S. Air Force Research Laboratory, in 1997. John has served as Chair of Committee to 32 completed graduate degrees, and his students have won national and regional student research competitions in topics ranging from aircraft design to smart materials to computational intelligence. He was Faculty Advisor to the Texas A&M student branch of AIAA from 2000–2009, and he received the National Faculty Advisor Award from AIAA in 2005. John is an Associate Editor of the *Journal of Guidance, Control, and Dynamics* and a current member of the Unmanned Systems Technical Program Committee; the Guidance, Navigation, and Control Technical Committee; and the Intelligent Systems Technical Committee. John earned his B.S. degree in aerospace engineering from California State Polytechnic University, Pomona, in 1986 and his M.S. degree with honors and Ph.D. in aerospace engineering from the University of Kansas, in 1990 and 1995, respectively. He is an Associate Fellow of AIAA.

Presented as Paper 2010-8159 at the AIAA Guidance, Navigation, and Control Conference, Toronto, Canada, 2–5 August 2010; received 22 August 2010; revision received 22 November 2010; accepted for publication 29 November 2010.

p = number of control variables
p, q, r = body roll, pitch, and yaw rates, deg/s
r = degree of smoothness
S = reference area, ft^2
t = slow time scale
T_m = maximum thrust, lb
t_0 = initial time
t^* = some finite time; greater than t_0
$\mathbf{u}$ = control vector
$V(t, \tilde{\mathbf{x}})$ = Lyapunov function for closed-loop reduced slow subsystem
v_s = speed of sound, ft/s
$\mathbf{w}$ = vector $[\mathbf{x}, \epsilon]^T$
$W(t, \tilde{\mathbf{z}})$ = Lyapunov function for closed-loop reduced fast subsystem
$\mathbf{x}, \mathbf{z}$ = state variables of full-order system
$\tilde{\mathbf{x}}$ = tracking error
$\tilde{\mathbf{z}}$ = error between fast variable and exact manifold
α = angle of attack, deg
β = sideslip angle, deg
$\Delta\mathbf{w}, \Delta\mathbf{z}$ = perturbation quantities
$\delta_e, \delta_a, \delta_r$ = elevator, aileron, and rudder control inputs, deg
ϵ = scalar perturbation parameter
$\epsilon^*(\epsilon_s^*)$ = upper bound for scalar perturbation parameter (for stabilization problem)
η = throttle input
θ = pitch attitude angle, deg
μ, γ = wind-axes orientation angles, deg
$\nu(t, \tilde{\mathbf{x}}, \tilde{\mathbf{z}})$ = Lyapunov function for complete system
ρ = density of air, slug/ft^3
τ = fast time scale
$\Phi(.)$ = approximation of exact manifold
ϕ = roll attitude angle, deg
$\phi(.)$ = approximation to exact manifold
ψ = heading angle, deg
$\dot{}$ = derivative with respect to slow time scale
$'$ = derivative with respect to fast time scale
$\|.\|$ = Euclidean norm

Subscripts

b = bound on variable
r = reference

Superscripts

S = stable
U = unstable

I. Introduction

MATHEMATICAL modeling of many physical systems requires high-order dynamic equations. The presence of parameters such as spring constant, mass, and moments of inertia are the cause of stiffness and increased order of these equations. It is difficult to arrive at exact analytical solutions of these nonlinear governing equations with known, and sometimes unknown, variable coefficients, so an approximate solution is often computed. Singular perturbation theory is a scheme used to simplify systems that inherently possess both fast and slow dynamics. Such systems are characterized by a small parameter ϵ multiplying the highest derivative. Suppression of this small parameter reduces the order of the system, and thus the label of singularly perturbed. Singular perturbation theory dates back to the 1904 work of Prandtl [1] on fluid boundary layers; subsequently, applications of perturbation methods were explored for control design [2–4].

The main contribution of perturbation methods is at the level of modeling, where it has been used as a model-reduction technique as well as a means of removing the numerical stiffness in the original system. In particular, the method of matched asymptotic expansions reduces the study of the full-order system of equations to the study of two other degenerate models. The first model captures the dominant phenomena, and the neglected phenomena is handled in the second. For the full-order system of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{z}, \epsilon) \qquad \epsilon\dot{\mathbf{z}} = \boldsymbol{l}(\mathbf{x}, \mathbf{z}, \epsilon) \tag{1}$$

the lower-order models are developed to be the following:

Reduced slow subsystem,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{z}, \epsilon) \qquad \mathbf{0} = \boldsymbol{l}(\mathbf{x}, \mathbf{z}, \epsilon) \tag{2}$$

Reduced fast subsystem,

$$\mathbf{x}' = \mathbf{0} \qquad \mathbf{z}' = \boldsymbol{l}(\mathbf{x}, \mathbf{z}, \epsilon) \tag{3}$$

where ϵ represents the scalar perturbation parameter, and $'$ represents the derivative with respect to the fast time scale $\tau = (t - t_0)/\epsilon$. It has been shown that the behavior of the complete system of Eqs. (1) is constrained within the $\mathcal{O}(\epsilon)$ bound of the reduced slow subsystem, provided the dynamics of the reduced fast subsystem are stabilizing [5]. One problem evident with the reduced slow subsystem is the solution of the transcendental or algebraic set of equations for the fast states $\mathbf{z}$. It is known that there may be many solutions satisfying this set of equations. The standard singular perturbation model assumes that, in the domain of interest, these solutions be isolated real roots.

Tracking properties of standard singularly perturbed systems were first studied by Grujic [6] in 1982. This work laid the foundations of tracking theory in a Lyapunov sense. Later, in 1988, this work was extended for nonlinear time-varying singularly perturbed systems [7]. However, it is assumed that separate controls are available for both the reduced slow and the reduced fast subsystems, and the algebraic set of equations have a trivial solution. Christofides et al. [8] developed robust controller designs for systems with a stabilizable fast subsystem, and input/output linearizable slow subsystems with input-to-state stable inverse dynamics. This work considered a general class of nonlinear time-varying singularly perturbed systems that have dynamics linear in the fast states. Another approach to tracking was presented by Heck [9] in 1991. He addressed the design of sliding-mode controllers for a class of linear time-invariant systems where tracking of slow variables is desired. For both reduced subsystems, a sliding-mode controller is designed, and a composite of these controls is then implemented on the full-order system. The concept of composite control, or designing separate controllers, for each of the subsystems and then implementing their cumulative to the full higher-order system was initiated by Suzuki and Miura [10], and since then, this concept has been extensively used by researchers for robust stabilization of systems with time-scale properties [11–13].

In the aircraft literature, the rotational equations of motion constitute the fast subsystem. These equations are highly coupled and nonlinear; thus, there exist multiple solutions for the set of nonlinear algebraic equations. Tracking of slow variables for these systems is achieved by making two key assumptions. First, the control surface deflections do not affect the slow states. Second, the fast variables are the actuators for the slow subsystem. Pioneering work in this area was published by Menon et al. [14] in 1987. Reference [14] designed a flight-test trajectory control system using dynamic inversion. The output variables to be tracked were total velocity, angle of attack, sideslip, and altitude. Once the desired angular rates were calculated, the dynamic inversion was applied to the fast subsystem to compute the aerodynamic control surface deflections. This work was extended to overactuated systems by Snell et al. [15]. More recently, the same concept has been employed to design longitudinal windshear flight-control laws [16] and for control of generic reentry vehicles [17].

Although all of the systems studied fall under the category of Eqs. (1), different design methodologies have been developed for varied physical systems, and several different control techniques have been employed. The control laws developed for a general form of physical systems assume the existence of a unique solution of the transcendental equation. For general dynamical system models, the existence of isolated roots for the fast states is not guaranteed. Although aircraft literature addresses this problem by employing

assumptions about the plant model, there is no general methodology in the literature to date to design tracking control structures for singularly perturbed systems that are nonlinear, both in the slow and the fast states. The open-loop study of these systems has been the focus of the geometric singular perturbation theory [18]. This theory has been employed in the past for transforming dynamical systems into singular perturbation form [Eq. (1)] [19,20] and to develop reduced-order models [21]. Work by Sharkey and O'Reilly [22] used this approach to design stabilizing control laws for a special of class of singularly perturbed systems wherein the control appears only in the fast dynamics. The global nature of the preceding stabilization results was proved by Chen [23] later on in 1998.

In this paper, the use of geometric methods is extended to a general class of time-varying singularly perturbed systems that are nonlinear in both the slow and the fast states. The problem of control for this general class of singularly perturbed systems is addressed for the first time in a systematic manner. The paper makes two major contributions. First, this work is not restricted to systems that have a unique solution for the nonlinear algebraic set of equations of the slow subsystem. The presence of multiple roots is accounted for by proving that a center manifold exists for the slow subsystem. This allows for the incorporation of results from the center manifold theory that are helpful in obtaining approximate roots of the transcendental equations. Tracking control laws are designed for both the slow and the fast subsystems to track the desired reference and computed approximation, respectively, using a composite control methodology. Second, the composite control law is not a function of the scalar perturbation parameter, nor does it require knowledge of it. This is an important consideration for systems such as aircraft, where quantifying this parameter can be difficult. The proposed control scheme is able to guarantee asymptotic stabilization of states for a general class of nonlinear regulation problems and uniform bounded stability for the trajectory-following problem. Using Lyapunov theory, a conservative upper bound ϵ^* is derived for the singular perturbation parameter for which these results hold. From the stability analysis, it is shown that this approach applies to all classes of singularly perturbed systems, with tracking properties of standard singular perturbation models being a special case. The approach and methodology is demonstrated with simulation examples for a nonlinear generic two-degree-of-freedom kinetic model and a nonlinear, coupled six-degree-of-freedom F/A-18A Hornet aircraft.

The paper is organized as follows. Section II describes the class of systems considered and formulates the control problem. Section III presents the necessary concepts of geometric singular perturbation theory and motivates this work. Section IV makes an important observation about the existence of a center manifold for the singularly perturbed system and details the procedure to compute this manifold. Section V develops the reduced-order models and formulates the tracking control laws. The proof of stability and main results are also presented in this section. Numerical simulations are presented in Sec. VI, and conclusions are discussed in Sec. VII.

II. Problem Formulation

The dynamic system considered is the nonlinear affine in the control singularly perturbed system, mathematically expressed as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{z}) + \mathbf{g}(\mathbf{x}, \mathbf{z})\mathbf{u} \tag{4}$$

$$\epsilon\dot{\mathbf{z}} = \boldsymbol{l}(\mathbf{x}, \mathbf{z}) + \mathbf{k}(\mathbf{x}, \mathbf{z})\mathbf{u} \tag{5}$$

where $\mathbf{x} \in \mathbb{R}^m$ is the set of slow variables of the system, $\mathbf{z} \in \mathbb{R}^n$ is the vector of the fast variables, and $\mathbf{u} \in \mathbb{R}^p$ is the set of the control variable. The singular perturbation parameter satisfies $0 < \epsilon \ll 1$ and $\epsilon \in \mathbb{R}^+$. The vector fields $\mathbf{f}(.)$, $\mathbf{g}(.)$, $\boldsymbol{l}(.)$, and $\mathbf{k}(.)$ are such that the closed-loop system is twice continuously differentiable with respect to their arguments. The control objective is to control the slow state to asymptotically track a specified twice continuously differentiable time-varying bounded trajectory, or $\mathbf{x}(t) \to \mathbf{x}_r(t)$ as $t \to \infty$.

Remark 1: The functions $\mathbf{g}(\mathbf{x}, \mathbf{z})$ and $\mathbf{k}(\mathbf{x}, \mathbf{z})$ represent the control-influence terms, while all other terms such as inertial coupling and gravitational forces are all contained in $\mathbf{f}(\mathbf{x}, \mathbf{z})$ and $\boldsymbol{l}(\mathbf{x}, \mathbf{z})$.

Remark 2: For a rigid body, $\mathbf{x}$ are the translational velocities while $\mathbf{z}$ represents the angular velocities. The rotational dynamics for a rigid body contain the nonlinear inertial coupling terms. The function $\boldsymbol{l}(\mathbf{x}, \mathbf{z})$ captures this nonlinearity in the fast states.

III. Background: Geometric Singular Perturbation Theory

Singular perturbation theory is a tool used to obtain the reduced-order approximations of the full-order equations of motion, which are difficult to analyze. The theory is valid so long as the parameter ϵ remains sufficiently small and the time-scale behavior is preserved. The method of matched asymptotic expansions [24] and its variation, the method of composite expansions [24], have been the foremost methods employed to develop these reduced-order models. The alternative geometric approach describes the motion of the full-order system using the concept of invariant manifolds. Both approaches produce the exact same reduced-order models but with different assumptions about the system. Asymptotic methods assume that the dynamical system possesses isolated roots, while the geometric approach is more general and takes into consideration multiple nonisolated roots of nonlinear systems.

To introduce the necessary concepts of geometric singular perturbation theory for an open-loop dynamical system, consider the nonlinear autonomous system:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{z}) \tag{6}$$

$$\epsilon\dot{\mathbf{z}} = \boldsymbol{l}(\mathbf{x}, \mathbf{z}) \tag{7}$$

Note that the following results also apply to nonautonomous systems. Equations (6) and (7) can be rewritten in the fast time scale $\tau = (t - t_0)/\epsilon$ as

$$\mathbf{x}' = \epsilon\mathbf{f}(\mathbf{x}, \mathbf{z}) \tag{8}$$

$$\mathbf{z}' = \boldsymbol{l}(\mathbf{x}, \mathbf{z}) \tag{9}$$

The independent variables t and τ are referred to as the slow and the fast time scales, respectively, and Eqs. (6–9) (referred to as the slow and the fast systems, respectively) are equivalent whenever $\epsilon \neq 0$. First, the system is studied for $\epsilon = 0$. The fast system reduces to n dimensions with variables $\mathbf{x}$ as constant parameters, producing the reduced fast subsystem,

$$\mathbf{x}' = \mathbf{0} \tag{10}$$

$$\mathbf{z}' = \boldsymbol{l}(\mathbf{x}, \mathbf{z}) \tag{11}$$

On the other hand, the order of the slow system reduces to m dimensions and results in a set of differential-algebraic equations, producing the reduced slow subsystem,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{z}) \tag{12}$$

$$\mathbf{0} = \boldsymbol{l}(\mathbf{x}, \mathbf{z}) \tag{13}$$

The reduced slow system appears to be a locally flattened vector space of the complete slow system. Thus, the set of points $(\mathbf{x}, \mathbf{z}) \in \mathbb{R}^m \times \mathbb{R}^n$ is expected to have a $\mathcal{C}^r$ smooth manifold $\mathcal{M}_0$ of dimension m inside the zero set of function $\boldsymbol{l}(.)$, provided the functions $\mathbf{f}(.)$ and $\boldsymbol{l}(.)$ are assumed to be $\mathcal{C}^r$.

Assumption 1: The functions $\mathbf{f}(\mathbf{x}, \mathbf{z})$ and $\boldsymbol{l}(\mathbf{x}, \mathbf{z})$ are sufficiently smooth so that $\mathcal{C}^r$ with $r \geq 1$.

The requirement to be continuous and at least once differentiable assures smoothness of the manifold $\mathcal{M}_0$. The flow on this manifold evolves as

$$\dot{\mathbf{x}} = \mathbf{f}[\mathbf{x}, \mathbf{h}_0(\mathbf{x})] \tag{14}$$

where $\mathbf{h}_0(\mathbf{x})$ is the solution of the algebraic part [Eq. (13)] that defines the manifold,

$$\mathcal{M}_0\text{: } \mathbf{z} = \mathbf{h}_0(\mathbf{x}); \qquad \mathbf{x} \in \mathbb{R}^m, \qquad \mathbf{z} \in \mathbb{R}^n \tag{15}$$

When viewed from the perspective of the reduced fast subsystem, the manifold $\mathcal{M}_0$ is the set of fixed points $[\mathbf{x}, \mathbf{h}_0(\mathbf{x})]$; therefore, $\mathcal{M}_0$ is trivially invariant. If every fixed point $[\mathbf{x}, \mathbf{h}_0(\mathbf{x})]$ of the reduced fast subsystem is assumed to be hyperbolic, then starting from arbitrary initial conditions, the flow will settle down exponentially fast onto the manifold, after which the flow evolves according to Eq. (14). Equivalently, the flow normal to the manifold is faster than that tangential to it. Such a manifold is said to be normally hyperbolic. Furthermore, a normally hyperbolic invariant manifold has local, $\mathcal{C}^r$ smooth stable, and unstable manifolds: $\mathcal{W}^S_{\text{loc}}(\mathcal{M}_0)$ and $\mathcal{W}^U_{\text{loc}}(\mathcal{M}_0)$. These manifolds are unions over all $(\mathbf{x})$ in $\mathcal{M}_0$ of the local stable and unstable manifolds of the reduced fast subsystem's hyperbolic fixed points $[\mathbf{x}, \mathbf{h}_0(\mathbf{x})]$.

To show these concepts, consider the following example. Let

$$\dot{x}_1 = -x_1 \qquad \dot{x}_2 = -x_2 \qquad \epsilon\dot{z} = -z \tag{16}$$

so that the reduced slow subsystem is

$$\dot{x}_1 = -x_1 \qquad \dot{x}_2 = -x_2 \qquad -z = 0 \tag{17}$$

and the reduced fast subsystem is

$$x_1 = 0 \qquad x_2 = 0 \qquad z' = -z \tag{18}$$

The solution of the algebraic equation (17) is $z = 0$, which is also the fixed point of Eq. (18). The invariant manifold is given by $\mathcal{M}_0$: $z = 0$, which is the complete x_1-x_2 plane. The origin is the stable hyperbolic equilibrium of the reduced slow subsystem, so any trajectory starting on the manifold approaches the origin in forward time, as seen in Fig. 1. Studying the reduced fast subsystem suggests that, for any point with nonzero initial condition $z(0)$, the flow approaches normal to the manifold. Intuitively, one may conclude that, for initial conditions not on the manifold, the reduced fast subsystem describes the transition to the manifold, after which the system evolves according to the reduced slow subsystem (seen in Fig. 2). Furthermore, since all points (x_1, x_2, z) approach the manifold at an exponential rate forward in time, the complete space is the stable manifold $\mathcal{W}^S(\mathcal{M}_0)$.

For the full-order system, similar inferences can be made. The presence of ϵ in Eq. (7) indicates that the fast variables grow relatively faster than the other states of the system. If their open-loop system is stabilizing, these states quickly settle down to their equilibrium. The other variables continue to evolve in time with the fast variables fixed by an equilibrium hypersurface. Mathematically, $\exists\ t^*$: $t^* > t_0$, after which the solutions $\mathbf{x}(t, \epsilon)$ and $\mathbf{z}(t, \epsilon)$ lie on a distinct m dimensional-invariant manifold $\mathcal{M}_\epsilon$:

$$\mathcal{M}_\epsilon\text{: } \mathbf{z} = \mathbf{h}(\mathbf{x}, \epsilon); \qquad \mathbf{x} \in \mathbb{R}^m, \qquad \mathbf{z} \in \mathbb{R}^n \tag{19}$$

For the system of Eqs. (16), the invariant manifold continues to be the x_1-x_2 plane. In addition, the family of lines parallel to the z axes still describe the flow normal to the manifold. Consider Fig. 3 to study this behavior. To generate this figure, ϵ was chosen to be 0.05. For a fixed initial condition, the flow evolves in two parts: one component along the manifold $\mathcal{M}_\epsilon$, which is governed by the reduced slow subsystem, and the other component in the normal direction, for which the flow is governed by the reduced fast subsystem. Points that are already on the manifold are seen to evolve similar to the flow sketched in Fig. 1. Thus, the reduced-order models provide good insight into the behavior of the full-order system. It is apparent that if the reduced fast subsystem were unstable, then an initial condition not on the manifold would move farther away in time. For the example considered, the manifolds $\mathcal{M}_0$ and $\mathcal{M}_\epsilon$ were obtained to be identically equal, but this is not generally the case with nonlinear systems.

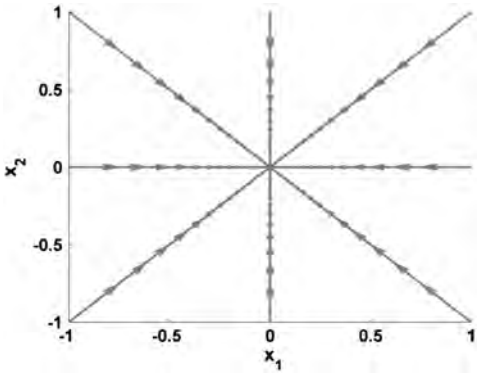


Fig. 1 Reduced slow subsystem.

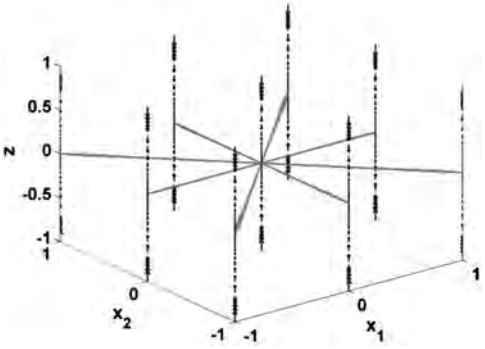


Fig. 2 The reduced slow and the fast subsystems.

The geometric constructs discussed previously are formal statements of Fenichel's persistence theory [18]. First, the following assumptions about the slow system are made:

Assumption 2: There exists a set $\mathcal{M}_0$ that is contained in $\{(\mathbf{x}, \mathbf{z})\text{: } \boldsymbol{l}(\mathbf{x}, \mathbf{z}) = \mathbf{0}\}$, such that $\mathcal{M}_0$ is a compact boundaryless manifold.

Assumption 3: $\mathcal{M}_0$ is normally hyperbolic relative to the reduced fast subsystem and, in particular, it is required that for all points $\mathbf{z} \in \mathcal{M}_0$, there are k (respectively, l) eigenvalues of $D_{\mathbf{z}}\boldsymbol{l}(0, \mathbf{z})$ with positive (respectively, negative) real parts that are bounded away from zero, where $k + l = n$.

The following theorem from Fenichel [18] is for compact boundaryless manifolds. Let the slow system satisfy Assumptions 1, 2, and 3. If $\epsilon > 0$ is sufficiently small, then there exists a manifold $\mathcal{M}_\epsilon$ that is $\mathcal{C}^{r-1}$ smooth locally invariant under the fast system and $\mathcal{C}^{r-1}$ $\mathcal{O}(\epsilon)$ close to $\mathcal{M}_0$. In addition, there exist perturbed local stable and

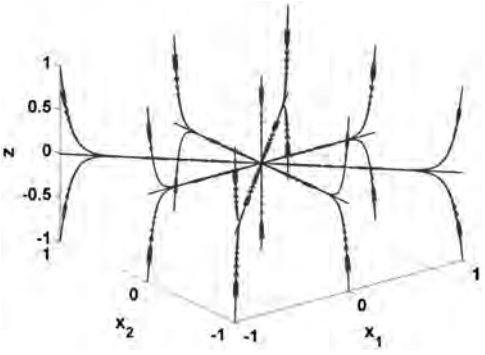


Fig. 3 Flow of system of equations (16) when $\epsilon \neq 0$.

unstable manifolds of $\mathcal{M}_\epsilon$, and they are $\mathcal{C}^r$ $\mathcal{O}(\epsilon)$ close, for all $r < \infty$, to their unperturbed counterparts.

IV. Center Manifold and Computation

Fenichel's theorem [18] is a powerful tool to study the behavior of stiff dynamical systems. It asserts the presence of an invariant manifold $\mathcal{M}_\epsilon$ that is $\mathcal{O}(\epsilon)$ close to $\mathcal{M}_0$, but it does not provide the procedure to compute the manifold. Since $\mathcal{M}_\epsilon$ is invariant for some $t \geq t^*$, the solutions follow the curve specified in Eq. (19). Differentiating this expression with respect to t,

$$\dot{\mathbf{z}} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \dot{\mathbf{x}} \tag{20}$$

and multiplying Eq. (20) with ϵ and substituting for $\dot{\mathbf{x}}$ and $\dot{\mathbf{z}}$ from Eqs. (6) and (7) results in

$$\epsilon \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \mathbf{f}[\mathbf{x}, \mathbf{h}(\mathbf{x}, \epsilon)] = \boldsymbol{l}[\mathbf{x}, \mathbf{h}(\mathbf{x}, \epsilon)] \tag{21}$$

Equation (21) is called the manifold condition. Note that substituting $\epsilon = 0$ in the manifold condition returns Eq. (13), which is satisfied by the manifold $\mathcal{M}_0$. To employ Fenichel's results [18], the manifold condition needs to be solved. Exact computation is impossible, since solving this condition is equivalent to solving the complete nonlinear system. One approximate approach is to substitute a perturbation expansion for $\mathbf{h}(\mathbf{x}, \epsilon) = \mathbf{h}_0(\mathbf{x}) + \epsilon \mathbf{h}_1(\mathbf{x}) + \mathcal{O}(\epsilon^2)$ into Eq. (21) and then solve order by order for $\mathbf{h}(\mathbf{x}, \epsilon)$. If the domain of interest is known, then the implicit function theorem may be employed. It is usually the inverse problem that is encountered, which is to find $\mathbf{h}(\mathbf{x}, \epsilon)$ as a smooth function of its arguments. In this paper, the approach proposed in [25] is used, and the following discusses its computation procedure.

The computation procedure proposed in [25] has been laid out for dynamical systems with center manifolds. For completeness, the first step is to check whether the manifold $\mathcal{M}_\epsilon$ is the center manifold of the singularly perturbed system. To study this behavior, rewrite the fast system using the technique called suspension [26] as

$$\mathbf{x}' = \epsilon \mathbf{f}(\mathbf{x}, \mathbf{z}) \tag{22}$$

$$\epsilon' = 0 \tag{23}$$

$$\mathbf{z}' = \boldsymbol{l}(\mathbf{x}, \mathbf{z}) \tag{24}$$

Assume that the origin is the fixed point of the preceding system that is $\mathbf{f}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$ and $\boldsymbol{l}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$. Then, the perturbed system obtained by linearizing these equations about the origin [$\epsilon = 0, \mathbf{x} = \mathbf{0}, \mathbf{h}(\mathbf{0}, 0) = \mathbf{0}$] is written in compact form as

$$\Delta \mathbf{w}' = F\mathbf{w} + F_1 \mathbf{z} \qquad \Delta \mathbf{z}' = L\mathbf{z} + L_1 \mathbf{w} \tag{25}$$

where $\mathbf{w} = [\mathbf{x}, \epsilon]^T$, $\Delta \mathbf{w}$, and $\Delta \mathbf{z}$ denote the perturbation quantities, and F, F_1, L, and L_1 are constant matrices of appropriate size. If all eigenvalues of F have zero real parts while all eigenvalues of L have negative real parts, then the manifold $\mathcal{M}_\epsilon$ is precisely the center manifold, and it spans the generalized eigenvectors associated with eigenvalues with zero real parts. This manifold is defined for all small values of the slow state $\mathbf{x}$ and the perturbation parameter ϵ. The requirement on eigenvalues of F supports the existence of time scales in the system, for if the eigenvalues were nonzero, then all states would be fast variables, and the system is not singularly perturbed. This suggests that the eigenvalue restriction on F is always satisfied by systems with the multiple time-scale property. The other requirement of negative eigenvalues of L is to ensure that the trajectories not on the manifold approach it in forward time.

From the preceding analysis, $\mathbf{h}(\mathbf{x}, \epsilon)$ is known to be the center manifold. If the origin is the fixed point of the linearized system, then the theorem from [25] asserts that one can approximate $\mathbf{h}(\mathbf{x}, \epsilon)$ to any degree of accuracy. For functions ϕ: $\mathbb{R}^m \times \mathbb{R} \to \mathbb{R}^n$, which are $\mathcal{C}^{r-1}$ (r defined as in Assumption 1) in the neighborhood of the origin, define

$$(M\phi)(\mathbf{x}, \epsilon) = \epsilon \frac{\partial \phi}{\partial \mathbf{x}} \mathbf{f}[\mathbf{x}, \phi(\mathbf{x}, \epsilon)] - \boldsymbol{l}[\mathbf{x}, \phi(\mathbf{x}, \epsilon)] \tag{26}$$

Note that, by Eq. (21), $(M\mathbf{h})(\mathbf{x}, \epsilon) = \mathbf{0}$.

The following is the theorem from [25]. Let ϕ: $\mathbb{R}^m \times \mathbb{R} \to \mathbb{R}^n$ satisfy $\phi(\mathbf{0}, 0) = \mathbf{0}$ and $|(M\phi)(\mathbf{x}, \epsilon)| = \mathcal{O}[C(\mathbf{x}, \epsilon)]$ for $|\mathbf{x}| \to 0$ and $\epsilon \to 0$, where $C(.)$ is a polynomial of degree greater than one. Then,

$$|\mathbf{h}(\mathbf{x}, \epsilon) - \phi(\mathbf{x}, \epsilon)| = \mathcal{O}[C(\mathbf{x}, \epsilon)]$$

This theorem implies that an approximate function $\phi(\mathbf{x}, \epsilon)$ can be determined for small values of $\mathbf{x}$ and ϵ. The condition $\phi(\mathbf{0}, 0) = \mathbf{0}$ is to ensure that the origin remains the fixed point. To demonstrate the procedure, consider the example from [25]:

$$\dot{x} = xz + ax^3 + bz^2 x \qquad \epsilon \dot{z} = -z + cx^2 + dx^2 z \tag{27}$$

Linearizing this system about the origin,

$$\Delta x' = 0 \qquad \Delta \epsilon' = 0 \qquad \Delta z' = -1 \tag{28}$$

It is seen that the system possesses a center manifold $z = h(x, \epsilon)$. To approximate h, define

$$(M\phi)(x, \epsilon) = \epsilon \frac{\partial \phi}{\partial x}[x\phi(x, \epsilon) + ax^3 + b\phi^2(x, \epsilon)x] + \phi(x, \epsilon) - cx^2 - dx^2\phi(x, \epsilon) \tag{29}$$

Hence, if $\phi(x, \epsilon) = cx^2$, then $(M\phi)(x, \epsilon) = \mathcal{O}(|x^4| + |\epsilon x^4|)$, and from the preceding theorem, $h(x, \epsilon) = cx^2 + \mathcal{O}(|x^4| + |\epsilon x^4|)$. Since the fast subsystem is stabilizing, geometric singular perturbation theory says that stability of the complete system can be analyzed by studying the flow on the manifold [Eq. (14)]:

$$\dot{x} = (a + c)x^3 + bcx^5 + \mathcal{O}(|x^5| + |\epsilon x^5|) \tag{30}$$

V. Control Formulation and Stability Analysis

The central idea is the following. If the reduced fast subsystem is stabilizing about the manifold $\mathcal{M}_0$, the complete system dynamics remain $\mathcal{O}(\epsilon)$ close to the reduced slow subsystem. This fact is employed to develop a stable closed-loop system. It is proposed that two separate stabilizing controllers be designed for each of the subsystems and their composite be fed to the complete system. It is shown that, in fact, this composite control uniformly stabilizes the complete system. This approach has been shown in the literature to guarantee asymptotic stability for singularly perturbed systems with unique manifolds $\mathcal{M}_0$ [10]. In the following subsections, control laws for a general class of nonlinear singularly perturbed systems are formulated, and closed-loop system stability is analyzed.

A. Control Law Development

The objective is to augment the two time-scale system with state feedback controllers such that the system follows a specified continuous twice differentiable bounded trajectory $\mathbf{x}_r(t)$. The first step is to transform the system [Eqs. (4) and (5)] into a non-autonomous stabilization problem. Define the error signal as $\tilde{\mathbf{x}}(t) = \mathbf{x}(t) - \mathbf{x}_r(t)$. Then,

$$\dot{\tilde{\mathbf{x}}} = \mathbf{f}(\tilde{\mathbf{x}} + \mathbf{x}_r, \mathbf{z}) + \mathbf{g}(\tilde{\mathbf{x}} + \mathbf{x}_r, \mathbf{z})\mathbf{u} - \dot{\mathbf{x}}_r \tag{31}$$

$$\epsilon \dot{\mathbf{z}} = \boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, \mathbf{z}) + \mathbf{k}(\tilde{\mathbf{x}} + \mathbf{x}_r, \mathbf{z})\mathbf{u} \tag{32}$$

The objective is to seek the control vector of the form $\mathbf{u} = \mathbf{u}_s + \mathbf{u}_f$, where

$$\mathbf{u}_s = \Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) \tag{33}$$

and

$$\mathbf{u}_f = \Gamma_f(\tilde{\mathbf{x}}, \mathbf{z}, \mathbf{x}_r, \dot{\mathbf{x}}_r) \tag{34}$$

Substituting the controls into Eqs. (31) and (32) produces

$$\dot{\tilde{\mathbf{x}}} = \mathbf{f}(\tilde{\mathbf{x}} + \mathbf{x}_r, \mathbf{z}) + \mathbf{g}(\tilde{\mathbf{x}} + \mathbf{x}_r, \mathbf{z})[\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \Gamma_f(\tilde{\mathbf{x}}, \mathbf{z}, \mathbf{x}_r, \dot{\mathbf{x}}_r)] - \dot{\mathbf{x}}_r \tag{35}$$

$$\epsilon\dot{\mathbf{z}} = \boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, \mathbf{z}) + \mathbf{k}(\tilde{\mathbf{x}} + \mathbf{x}_r, \mathbf{z})[\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \Gamma_f(\tilde{\mathbf{x}}, \mathbf{z}, \mathbf{x}_r, \dot{\mathbf{x}}_r)] \tag{36}$$

Assume that the right-hand side of Eqs. (35) and (36) is $\mathcal{C}^2$; that is, the vector fields satisfy Assumption 1 with $r = 2$. From Fenichel's theorem [18], it can be concluded that there exists a manifold

$$\mathcal{M}_\epsilon\colon \mathbf{z} = \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r) \tag{37}$$

that satisfies the manifold condition,

$$\epsilon\frac{\partial \mathbf{h}}{\partial t} + \epsilon\frac{\partial \mathbf{h}}{\partial \tilde{\mathbf{x}}}\dot{\tilde{\mathbf{x}}} = \boldsymbol{l}[\tilde{\mathbf{x}} + \mathbf{x}_r, \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)] + \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\Gamma_f[\tilde{\mathbf{x}}, \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r), \mathbf{x}_r, \dot{\mathbf{x}}_r] \tag{38}$$

Note that the manifold is time dependent, since the system under consideration is nonautonomous due to the time varying $\mathbf{x}_r(t)$. Define the error between the fast states and the manifold $\mathcal{M}_\epsilon$ as $\tilde{\mathbf{z}} = \mathbf{z} - \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)$.

The transformed system with the origin as the equilibrium is expressed as

$$\dot{\tilde{\mathbf{x}}} = \mathbf{f}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)] + \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\Gamma_f[\tilde{\mathbf{x}}, \tilde{\mathbf{z}} + \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r), \mathbf{x}_r, \dot{\mathbf{x}}_r] - \dot{\mathbf{x}}_r \tag{39}$$

$$\epsilon\dot{\tilde{\mathbf{z}}} = \boldsymbol{l}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)] + \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\Gamma_f[\tilde{\mathbf{x}}, \tilde{\mathbf{z}} + \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r), \mathbf{x}_r, \dot{\mathbf{x}}_r] - \epsilon\frac{\partial \mathbf{h}}{\partial t} - \epsilon\frac{\partial \mathbf{h}}{\partial \tilde{\mathbf{x}}}\dot{\tilde{\mathbf{x}}} \tag{40}$$

Note that the error $\tilde{\mathbf{z}} = \mathbf{0}$ when the manifold condition is satisfied. It is known that the exact manifold $\mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)$ is impossible to compute. Let $\phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)$ be an approximate manifold obtained using the procedure presented in Sec. IV. The approximate manifold is chosen to contain terms independent of ϵ, similar to the example considered at the end of Sec. IV. Define

$$(M\phi)(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s, \Gamma_f) = \epsilon\frac{\partial \phi}{\partial t} + \epsilon\frac{\partial \phi}{\partial \tilde{\mathbf{x}}}\dot{\tilde{\mathbf{x}}} - \boldsymbol{l}[\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] - \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) - \mathbf{k}(\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s))\Gamma_f(\tilde{\mathbf{x}}, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s), \mathbf{x}_r, \dot{\mathbf{x}}_r) \tag{41}$$

and let $(M\phi)(t, \tilde{\mathbf{x}}, \epsilon) = \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]$, which depends on the choice of controls Γ_s and Γ_f. Furthermore, the following is assumed:

Assumption A: The choice of controls Γ_s and Γ_f leads to $\mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon = 0, \mathbf{x}_r, \dot{\mathbf{x}}_r)] = \mathbf{0}$.

With the preceding choice of $\phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)$, the exact manifold is given as

$$\mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r) = \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]$$

Substituting the approximate expression for the manifold into Eqs. (39) and (40),

$$\dot{\tilde{\mathbf{x}}} = \mathbf{f}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} + \mathbf{g}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\}\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{g}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\}\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) - \dot{\mathbf{x}}_r \tag{42}$$

$$\epsilon\dot{\tilde{\mathbf{z}}} = \boldsymbol{l}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} + \mathbf{k}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r]\})\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{k}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\}\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) - \epsilon\frac{\partial\{\phi + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\}}{\partial t} - \epsilon\frac{\partial\{\phi + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\}}{\partial \tilde{\mathbf{x}}}\dot{\tilde{\mathbf{x}}} \tag{43}$$

Note that Γ_f is a function of Γ_s due to the choice of $\phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)$. The reduced slow and fast subsystems for the system of Eqs. (42) and (43) are obtained by substituting $\epsilon = 0$, resulting in the reduced slow subsystem,

$$\dot{\tilde{\mathbf{x}}} = \mathbf{f}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] + \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) - \dot{\mathbf{x}}_r \tag{44}$$

$$\mathbf{0} = \boldsymbol{l}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] + \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \tag{45}$$

and the reduced fast subsystem,

$$\tilde{\mathbf{x}}' = \mathbf{0} \tag{46}$$

$$\tilde{\mathbf{z}}' = \boldsymbol{l}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] + \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \tag{47}$$

In general, the composite control approach first computes the control Γ_s required to maintain reduced slow subsystem stability by assuming that the fast states lie upon the manifold and $\Gamma_f = \mathbf{0}$. In the next step, the control Γ_f is designed to satisfy two conditions: guarantee uniform convergence of the fast states onto the manifold and remain inactive when the fast state remains on the manifold. The second condition is implemented to avoid affecting the conclusions drawn about the reduced slow subsystem stability. In the proposed control scheme, the second condition is avoided by designing Γ_f ahead of Γ_s. Thus, design $\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)$ as a function of Γ_s, such that Eq. (47) is transformed into the closed-loop reduced fast subsystem,

$$\tilde{\mathbf{z}}' = -\mathbf{L_f}(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{K_f}(\tilde{\mathbf{z}}) \tag{48}$$

such that $-\mathbf{L_f}(\tilde{\mathbf{x}}, \mathbf{0}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{K_f}(\mathbf{0}) = \mathbf{0}$. With this choice of Γ_f and assumptions about vector fields $\mathbf{L_f}$ and $\mathbf{K_f}$, $\tilde{\mathbf{z}} = \mathbf{0}$ becomes the

unique root of Eq. (45). Therefore, the reduced slow subsystem reduces to

$$\begin{aligned}\dot{\tilde{\mathbf{x}}} &= \mathbf{f}[\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] \\ &\quad + \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) \\ &\quad + \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \mathbf{0}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) - \dot{\mathbf{x}}_r \end{aligned} \tag{49}$$

The only unknown in Eq. (49) is Γ_s; therefore, it may be designed to transform the reduced slow subsystem into the closed-loop reduced slow subsystem,

$$\dot{\tilde{\mathbf{x}}} = -\mathbf{F}_\mathbf{s}(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{G}_\mathbf{s}(\tilde{\mathbf{x}}) \tag{50}$$

and exact forms of $\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r)$, $\phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r)$, and correspondingly $C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)$, can be determined through Eqs. (48) and (41), respectively.

Remark 3: In the reduced subsystems obtained, $\tilde{\mathbf{z}} = \mathbf{z} - \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r)$ by Assumption A. Thus, at the implementation level, the control Γ_f is a function of known quantities.

The complete closed-loop system is obtained by rewriting Eqs. (42) and (43) as

$$\begin{aligned}\dot{\tilde{\mathbf{x}}} &= \mathbf{f}[\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] + \mathbf{g}[\tilde{\mathbf{x}} \\ &\quad + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{g}[\tilde{\mathbf{x}} \\ &\quad + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \mathbf{0}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) - \dot{\mathbf{x}}_r + \mathbf{f}[\tilde{\mathbf{x}} \\ &\quad + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] - \mathbf{f}(\tilde{\mathbf{x}} + \mathbf{x}_r, \phi[\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] \\ &\quad + \{\mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] - \mathbf{g}[\tilde{\mathbf{x}} \\ &\quad + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\}\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) - \mathbf{g}[\tilde{\mathbf{x}} \\ &\quad + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \mathbf{0}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathbf{f}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{f}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] + (\mathbf{g}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \\ &\quad + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + (\mathbf{g}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \end{aligned} \tag{51}$$

$$\begin{aligned}\epsilon\dot{\tilde{\mathbf{z}}} &= \boldsymbol{l}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] + \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \boldsymbol{l}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \boldsymbol{l}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] + (\mathbf{k}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \\ &\quad + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + (\mathbf{k}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \\ &\quad - \epsilon\frac{\partial\{\phi + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\}}{\partial t} - \epsilon\frac{\partial\{\phi + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\}}{\partial\tilde{\mathbf{x}}}\dot{\tilde{\mathbf{x}}} \end{aligned} \tag{52}$$

Using the closed-loop reduced subsystems of Eqs. (48) and (50), Eqs. (51) and (52) become the closed-loop complete system,

$$\begin{aligned}\dot{\tilde{\mathbf{x}}} &= -\mathbf{F}_\mathbf{s}(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{G}_\mathbf{s}(\tilde{\mathbf{x}}) + \mathbf{f}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] \\ &\quad - \mathbf{f}[\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] + \{\mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] - \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\}\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) \\ &\quad + \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) - \mathbf{g}[\tilde{\mathbf{x}} \\ &\quad + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \mathbf{0}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathbf{f}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{f}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] + (\mathbf{g}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \\ &\quad + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + (\mathbf{g}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} \\ &\quad + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \end{aligned} \tag{53}$$

$$\begin{aligned}\epsilon\dot{\tilde{\mathbf{z}}} &= -\mathbf{L}_\mathbf{f}(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{K}_\mathbf{f}(\tilde{\mathbf{z}}) + \boldsymbol{l}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \\ &\quad + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \boldsymbol{l}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] \\ &\quad + (\mathbf{k}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} \\ &\quad - \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + (\mathbf{k}\{\tilde{\mathbf{x}} \\ &\quad + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{k}[\tilde{\mathbf{x}} \\ &\quad + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \\ &\quad - \epsilon\frac{\partial\{\phi + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\}}{\partial t} - \epsilon\frac{\partial\{\phi + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\}}{\partial\tilde{\mathbf{x}}}\dot{\tilde{\mathbf{x}}} \end{aligned} \tag{54}$$

Remark 4: If $\phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r)$ is the unique manifold for the complete system, then the terms of $\mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]$ are identically zero, and the closed-loop complete system in Eqs. (53) and (54) takes the form as in [27,28], which has been proven to be closed-loop stable.

B. Stability Analysis

1. Tracking Problem

The following theorem summarizes the main result of the paper.

Theorem 1: Suppose the controls $\mathbf{u}_s$ and $\mathbf{u}_f$ are designed according to Eqs. (48) and (50) and Assumptions A–I hold. Then for all initial conditions, $(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}) \in D_x \times D_z$, the composite control $\mathbf{u} = \mathbf{u}_s + \mathbf{u}_f$ uniformly stabilizes the nonlinear singularly perturbed system in Eqs. (4) and (5) for all $\epsilon < \epsilon^*$, where ϵ^* is given by the inequality equation (68), and the error signals $\tilde{\mathbf{x}}(t)$ and $\tilde{\mathbf{z}}(t)$ are uniformly bounded by Eqs. (69) and (70), respectively.

Proof: Closed-loop system stability is analyzed using the composite Lyapunov function approach [29]. It is required to prove that the closed-loop system behavior remains close to the closed-loop reduced slow subsystem. Suppose that there are Lyapunov functions $V(t, \tilde{\mathbf{x}}) = \frac{1}{2}\tilde{\mathbf{x}}^T\tilde{\mathbf{x}}$ and $W(t, \tilde{\mathbf{z}}) = \frac{1}{2}\tilde{\mathbf{z}}^T\tilde{\mathbf{z}}$ for the closed-loop reduced-order models (50) and (48), respectively, satisfying the following eight assumptions:

B) $V(t, \tilde{\mathbf{x}})$ is positive definite and decrescent; that is,

$$c_1\|\tilde{\mathbf{x}}\|^2 \le V(t, \tilde{\mathbf{x}}) \le c_2\|\tilde{\mathbf{x}}\|^2, \qquad \tilde{\mathbf{x}} \in D_x \subset \mathbb{R}^m \tag{55}$$

C)

$$\frac{\partial V}{\partial\tilde{\mathbf{x}}}[-\mathbf{F}_\mathbf{s}(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{G}_\mathbf{s}(\tilde{\mathbf{x}})] \le -\alpha_1\|\tilde{\mathbf{x}}\|^2 - b_1\|\tilde{\mathbf{x}}\|, \quad \alpha_1 > 0, \; b_1 \ge 0 \tag{56}$$

D) There exists a constant $\beta_1 > 0$, such that

$$\frac{\partial V}{\partial \tilde{\mathbf{x}}}\{\mathbf{f}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] - \mathbf{f}[\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\} + \frac{\partial V}{\partial \tilde{\mathbf{x}}}\{\mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)] - \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\}\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \frac{\partial V}{\partial \tilde{\mathbf{x}}}\{\mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) - \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]\Gamma_f(\tilde{\mathbf{x}}, \mathbf{0}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)\} \leq \beta_1\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{z}}\| \quad (57)$$

E) There exist constants $\beta_2 > 0$, $\beta_3 > 0$, and $\beta_4 \geq 0$, such that

$$\frac{\partial V}{\partial \tilde{\mathbf{x}}}(\mathbf{f}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{f}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]) + \frac{\partial V}{\partial \tilde{\mathbf{x}}}(\mathbf{g}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \frac{\partial V}{\partial \tilde{\mathbf{x}}}(\mathbf{g}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{g}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \leq \epsilon\beta_2\|\tilde{\mathbf{x}}\|^2 + \epsilon\beta_3\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{z}}\| + \epsilon\beta_4\|\tilde{\mathbf{x}}\| \quad (58)$$

F) $W(t, \tilde{\mathbf{z}})$ is positive definite and decrescent scalar function satisfying,

$$c_3\|\tilde{\mathbf{z}}\|^2 \leq W(t, \tilde{\mathbf{z}}) \leq c_4\|\tilde{\mathbf{z}}\|^2, \qquad \tilde{\mathbf{z}} \in D_z \subset \mathbb{R}^n \quad (59)$$

G)

$$\frac{\partial W}{\partial \tilde{\mathbf{z}}}[-\mathbf{L}_\mathbf{f}(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \mathbf{K}_\mathbf{f}(\tilde{\mathbf{z}})] \leq -\alpha_2\|\tilde{\mathbf{z}}\|^2, \qquad \alpha_2 > 0 \quad (60)$$

H) There exist scalars $\beta_5 > 0$, $\beta_6 > 0$, and $\beta_7 \geq 0$, such that

$$\frac{\partial W}{\partial \tilde{\mathbf{z}}}(1\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \boldsymbol{I}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)]) + \frac{\partial W}{\partial \tilde{\mathbf{z}}}(\mathbf{k}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_s(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r) + \frac{\partial W}{\partial \tilde{\mathbf{z}}}(\mathbf{k}\{\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\} - \mathbf{k}[\tilde{\mathbf{x}} + \mathbf{x}_r, \tilde{\mathbf{z}} + \phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s)])\Gamma_f(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_s) \leq \epsilon\beta_5\|\tilde{\mathbf{z}}\|^2 + \epsilon\beta_6\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{z}}\| + \epsilon\beta_7\|\tilde{\mathbf{z}}\| \quad (61)$$

I) There exist constants $\beta_8 \geq 0$ and $\beta_9 > 0$, such that

$$-\frac{\partial W}{\partial \tilde{\mathbf{z}}}\left[\epsilon\frac{\partial\{\phi + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\}}{\partial t} + \epsilon\frac{\partial\{\phi + \mathcal{O}[C(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r)]\}}{\partial \tilde{\mathbf{x}}}\dot{\tilde{\mathbf{x}}}\right] \leq \epsilon\beta_8\|\tilde{\mathbf{z}}\| + \epsilon\beta_9\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{z}}\| \quad (62)$$

Remark 5: Assumptions B, C, F, and G are conditions for asymptotic stability of closed-loop reduced-order models. The constant b_1 in Assumption C depends upon the bounds of the specified trajectory $\mathbf{x}_r(t)$ and its derivative $\dot{\mathbf{x}}_r$. If the control Γ_s is designed to maintain asymptotic stability of the closed-loop slow subsystem, then $b_1 = 0$. Additionally, Assumptions D, E, H, and I are interconnection conditions obtained by assuming the vector fields are locally Lipschitz. The constants β_4, β_7, and β_8 appear due to the time-varying nature of the manifold and depend upon the bounds of $\mathbf{x}_r(t)$ and its derivative $\dot{\mathbf{x}}_r$. The constant β_8 also depends upon the derivative $\ddot{\mathbf{x}}_r$, which is known to be bounded by the choice of the reference trajectory. Consider the Lyapunov function candidate,

$$\nu(t, \tilde{\mathbf{x}}, \tilde{\mathbf{z}}) = V(t, \tilde{\mathbf{x}}) + W(\mathbf{t}, \tilde{\mathbf{z}}) \quad (63)$$

for the closed-loop system of Eqs. (53) and (54). From the properties of V and W, it follows that $\nu(t, \tilde{\mathbf{x}}, \tilde{\mathbf{z}})$ is positive definite and decrescent. The derivative of ν along the trajectories of Eqs. (53) and (54) is given by

$$\dot{\nu} = \frac{\partial V}{\partial \tilde{\mathbf{x}}}\dot{\tilde{\mathbf{x}}} + \frac{1}{\epsilon}\frac{\partial W}{\partial \tilde{\mathbf{z}}}\tilde{\mathbf{z}}' \quad (64)$$

Substituting Assumptions B–I into Eq. (64),

$$\dot{\nu} \leq -\alpha_1\|\tilde{\mathbf{x}}\|^2 - b_1\|\tilde{\mathbf{x}}\| + \beta_1\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{z}}\| + \epsilon\beta_2\|\tilde{\mathbf{x}}\|^2 + \epsilon\beta_3\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{z}}\| + \epsilon\beta_4\|\tilde{\mathbf{x}}\| - \frac{\alpha_2}{\epsilon}\|\tilde{\mathbf{z}}\|^2 + \beta_5\|\tilde{\mathbf{z}}\|^2 + \beta_6\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{z}}\| + \beta_7\|\tilde{\mathbf{z}}\| + \beta_8\|\tilde{\mathbf{z}}\| + \beta_9\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{z}}\| \quad (65)$$

Collecting like terms

$$\dot{\nu} \leq -(\alpha_1 - \epsilon\beta_2)\|\tilde{\mathbf{x}}\|^2 - (b_1 - \epsilon\beta_4)\|\tilde{\mathbf{x}}\| + (\beta_1 + \epsilon\beta_3 + \beta_6 + \beta_9)\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{z}}\| - \left(\frac{\alpha_2}{\epsilon} - \beta_5\right)\|\tilde{\mathbf{z}}\|^2 - (-\beta_7 - \beta_8)\|\tilde{\mathbf{z}}\| \quad (66)$$

Rearrange Eq. (66) to get

$$\dot{\nu} \leq \begin{bmatrix} \|\tilde{\mathbf{x}}\| \\ \|\tilde{\mathbf{z}}\| \end{bmatrix}^T \begin{bmatrix} -(1-d)(\alpha_1 - \epsilon\beta_2) & \frac{1}{2}(\beta_1 + \epsilon\beta_3 + \beta_6 + \beta_9) \\ \frac{1}{2}(\beta_1 + \epsilon\beta_3 + \beta_6 + \beta_9) & -(1-d)\left(\frac{\alpha_2}{\epsilon} - \beta_5\right) \end{bmatrix} \times \begin{bmatrix} \|\tilde{\mathbf{x}}\| \\ \|\tilde{\mathbf{z}}\| \end{bmatrix} - \|\tilde{\mathbf{x}}\|\{d(\alpha_1 - \epsilon\beta_2)\|\tilde{\mathbf{x}}\| - (\epsilon\beta_4 - b_1)\} - \|\tilde{\mathbf{z}}\|\left\{d\left(\frac{\alpha_2}{\epsilon} - \beta_5\right)\|\tilde{\mathbf{z}}\| - (\beta_7 + \beta_8)\right\}, \qquad 0 < d < 1 \quad (67)$$

The matrix becomes negative definite when

$$(1-d)^2(\alpha_1 - \epsilon\beta_2)\left(\frac{\alpha_2}{\epsilon} - \beta_5\right) < \frac{1}{4}(\beta_1 + \epsilon\beta_3 + \beta_6 + \beta_9)^2 \quad (68)$$

Thus, there exists an upper bound ϵ^* and upper bounds on the errors $\tilde{\mathbf{x}}_b$ and $\tilde{\mathbf{z}}_b$,

$$\tilde{\mathbf{x}}_b = \frac{\epsilon\beta_4 - b_1}{d(\alpha_1 - \epsilon\beta_2)} \quad (69)$$

$$\tilde{\mathbf{z}}_b = \frac{\beta_7 + \beta_8}{d(\frac{\alpha_2}{\epsilon} - \beta_5)} \quad (70)$$

for which

$$\dot{\nu} \leq 0 \quad (71)$$

From the Lyapunov theorem, it can then be concluded that the closed-loop signals $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}$ are uniformly bounded for all initial conditions $(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}) \in D_x \times D_z$. Consequently, the control vector $\mathbf{u} = \mathbf{u}_s + \mathbf{u}_f$ is bounded. Furthermore, since the trajectory $\mathbf{x}_r(t)$ is bounded, the manifold $\mathbf{h}(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \mathbf{x}_r, \dot{\mathbf{x}}_r)$ and the closed-loop signals $\mathbf{x}(t)$ and $\mathbf{z}(t)$ are bounded. □

2. *Special Case: Regulation Problem*

The following theorem summarizes the main result for the stabilization problem.

Theorem 2: Suppose the controls $\mathbf{u}_s$ and $\mathbf{u}_f$ are designed according to Eqs. (48–50), and Assumptions A–I hold with $\tilde{\mathbf{x}} = \mathbf{x}$ and $\tilde{\mathbf{z}} = \mathbf{z}$. Then, for all initial conditions $(\mathbf{x}, \mathbf{z}) \in D_x \times D_z$, the composite control $\mathbf{u} = \mathbf{u}_s + \mathbf{u}_f$ asymptotically stabilizes the nonlinear singularly perturbed systems in Eqs. (4) and (5) for all $\epsilon < \epsilon_s^*$, where ϵ_s^* is given by the inequality equation (68) with $d = 0$.

Proof: Note that, in this case, the manifold $\mathbf{h}(\mathbf{x}, \epsilon)$ is not time varying, and $\tilde{\mathbf{x}} = \mathbf{x}$ and $\tilde{\mathbf{z}} = \mathbf{z}$. Since this problem is autonomous, the decrescent conditions on the Lyapunov functions V and W can be relaxed. The constants β_4, β_7, and β_8 in Assumptions E, H, and I are all equal to zero, and the constant $b_1 = 0$, since $\mathbf{x}_r = \mathbf{0}$ and $\dot{\mathbf{x}}_r = \mathbf{0}$. With these modifications, Eq. (67) is modified as

$$\dot{v} \leq \begin{bmatrix} \|\mathbf{x}\| \\ \|\mathbf{z}\| \end{bmatrix}^T \begin{bmatrix} -(\alpha_1 - \epsilon\beta_2) & \frac{1}{2}(\beta_1 + \epsilon\beta_3 + \beta_6 + \beta_9) \\ \frac{1}{2}(\beta_1 + \epsilon\beta_3 + \beta_6 + \beta_9) & -\left(\frac{\alpha_2}{\epsilon} - \beta_5\right) \end{bmatrix} \times \begin{bmatrix} \|\mathbf{x}\| \\ \|\mathbf{z}\| \end{bmatrix} \tag{72}$$

Therefore, there exists an ϵ_s^* such that

$$\dot{v} < 0 \tag{73}$$

where ϵ_s^* satisfies the inequality equation (68) with $d = 0$. □

Remark 6: Theorems 1 and 2 depend upon the approximation of the invariant manifold, leading to local results. If it were possible to obtain the expression of the exact manifold, these results would be valid globally.

Remark 7: Fenichel's theorem [18] implies that the behavior of the complete nonlinear system remains close to the reduced slow subsystem if the reduced fast subsystem is stable. Theorems 1 and 2 state the same result for the closed-loop singularly perturbed system.

VI. Numerical Examples

A. Purpose and Scope

The preceding theoretical developments are demonstrated with simulation. The first example is a generic planar nonlinear system. This planar example enables the study of the geometric constructs, which are generally difficult to visualize in higher-dimension problems. A step-by-step procedure of controller development is detailed for the system to track a desired slow kinetic state. A comparison between the manifold approximation and the attained actual fast state is made. The closed-loop results are studied for a sinusoidal time-varying trajectory and the regulator problem. The second example develops control laws for a nonlinear F/A-18A Hornet model. The objective of this example is to test the performance of the controller for a highly nonlinear, two time-scale system. It is required to perform a turning maneuver while maintaining zero sideslip and tracking a specified angle-of-attack profile.

B. Generic Two-Degree-of-Freedom Nonlinear Kinetic Model

The fast dynamics are modified to include an arbitrarily chosen quadratic nonlinearity in the fast state, and a pseudocontrol term with unit effectiveness is introduced. For this example, $x \in \mathbb{R}$ and $z \in \mathbb{R}$ represent the slow and the fast states, respectively. The control $u \in \mathbb{R}$ is developed to track a desired smooth trajectory $x_r(t)$:

$$\dot{x} = -x + (x + 0.5)z + u \tag{74}$$

$$\epsilon\dot{z} = x - (x + 1)z + z^2 + u \tag{75}$$

The value $\epsilon = 0.2$ is retained in the modified model [26].

1. Controller Design

Define the errors $\tilde{x} = x - x_r$ and $\tilde{z} = z - h(\tilde{x}, \epsilon, x_r, \dot{x}_r)$, and transform Eqs. (74) and (75) into error coordinates equivalent to Eqs. (39) and (40):

$$\dot{\tilde{x}} = -(\tilde{x} + x_r) + (\tilde{x} + x_r + 0.5)[\tilde{z} + h(\tilde{x}, \epsilon, x_r, \dot{x}_r)] - \dot{x}_r + \Gamma_s + \Gamma_f \tag{76}$$

$$\epsilon\dot{\tilde{z}} = (\tilde{x} + x_r) - (\tilde{x} + x_r + 1)[\tilde{z} + h(\tilde{x}, \epsilon, x_r, \dot{x}_r)] + [\tilde{z} + h(\tilde{x}, \epsilon, x_r, \dot{x}_r)]^2 + \Gamma_s + \Gamma_f \tag{77}$$

Rearrange Eqs. (76) and (77), dropping arguments of h:

$$\dot{\tilde{x}} = -\tilde{x} + \tilde{x}h(.) + 0.5h(.) - x_r + x_r h(.) + (\tilde{x} + x_r + 0.5)\tilde{z} - \dot{x}_r + \Gamma_s + \Gamma_f \tag{78}$$

$$\epsilon\dot{\tilde{z}} = -(\tilde{x} + x_r + 1)\tilde{z} + \tilde{z}^2 + 2\tilde{z}h(.) + \tilde{x} + x_r - (\tilde{x} + x_r + 1)h(.) + h(.)^2 + \Gamma_s + \Gamma_f - \epsilon\frac{\partial h}{\partial t} - \epsilon\frac{\partial h}{\partial \tilde{x}}\dot{\tilde{x}} \tag{79}$$

Comparing with Eqs. (39) and (40),

$$\begin{aligned} \mathbf{f}(.) &= -\tilde{x} + \tilde{x}h(.) + 0.5h(.) - x_r + x_r h(.) + (\tilde{x} + x_r + 0.5)\tilde{z} \\ \mathbf{g}(.) &= 1 \\ \boldsymbol{l}(.) &= -(\tilde{x} + x_r + 1)\tilde{z} + \tilde{z}^2 + 2\tilde{z}h(.) + \tilde{x} + x_r - (\tilde{x} + x_r + 1)h(.) + h(.)^2 \\ \mathbf{k}(.) &= 1 \end{aligned} \tag{80}$$

Let $\phi(\tilde{x}, x_r, \dot{x}_r, \Gamma_s)$ be the approximate manifold. Define the manifold condition:

$$(M\phi)(\tilde{x}, x_r, \dot{x}_r, \Gamma_s, \Gamma_f) = \epsilon\frac{\partial \phi}{\partial t} + \epsilon\frac{\partial \phi}{\partial \tilde{x}}\dot{\tilde{x}} - \tilde{x} - x_r + (\tilde{x} + x_r)\phi(.) + \phi(.) - \phi(.)^2 - \Gamma_s - \Gamma_f \tag{81}$$

Select

$$\phi(\tilde{x}, x_r, \dot{x}_r, \Gamma_s) = \tilde{x} + x_r + \Gamma_s \tag{82}$$

so that

$$(M\phi)(\tilde{x}, x_r, \dot{x}_r, \Gamma_s) = \epsilon\frac{\partial \phi}{\partial t} + \epsilon\frac{\partial \phi}{\partial \tilde{x}}\dot{\tilde{x}} + (\tilde{x} + x_r)(\tilde{x} + x_r + \Gamma_s) - \phi(.)^2 - \Gamma_f \tag{83}$$

and

$$\mathcal{O}[C(\tilde{x}, \epsilon, x_r, \dot{x}_r)] = (M\phi)(\tilde{x}, x_r, \dot{x}_r, \Gamma_s)$$

To design the control Γ_f, develop the reduced fast subsystem equivalent to Eqs. (46) and (47):

$$\begin{aligned} \tilde{x}' &= 0 \\ \tilde{z}' &= -(\tilde{x} + x_r + 1)\tilde{z} + \tilde{z}^2 + 2\tilde{z}\phi(.) + \tilde{x} + x_r - (\tilde{x} + x_r + 1)\phi(.) + \phi(.)^2 + \Gamma_s + \Gamma_f \end{aligned} \tag{84}$$

Design

$$\Gamma_f = -A_f\tilde{z} - 2\tilde{z}\phi(.) + (\tilde{x} + x_r + 1)\phi(.) - \tilde{x} - x_r - \phi^2 - \Gamma_s \tag{85}$$

where A_f is a feedback gain. Then, the closed-loop reduced fast subsystem becomes

$$\tilde{z}' = -(\tilde{x} + x_r + 1 + A_f)\tilde{z} + \tilde{z}^2 \tag{86}$$

Comparing with Eq. (48),

$$\mathbf{L}_f(.) = (\tilde{x} + x_r + 1 + A_f)\tilde{z} \qquad \mathbf{K}_f(.) = \tilde{z}^2 \tag{87}$$

The next step is to determine the control Γ_s. Develop the reduced-order slow subsystem equivalent to Eqs. (44) and (45):

$$\dot{\tilde{x}} = -\tilde{x} + \tilde{x}\phi(.) + 0.5\phi(.) - x_r + x_r\phi(.) + (\tilde{x} + x_r + 0.5)\tilde{z} - \dot{x}_r + \Gamma_s + \Gamma_f \tag{88}$$

$$0 = -(\tilde{x} + x_r + 1)\tilde{z} + \tilde{z}^2 + 2\tilde{z}\phi(.) + \tilde{x} + x_r - (\tilde{x} + x_r + 1)\phi(.) + \phi(.)^2 + \Gamma_s + \Gamma_f \tag{89}$$

Substituting for Γ_f from Eq. (85) in Eqs. (88) and (89),

$$\dot{\tilde{x}} = -2\tilde{x} + \tilde{x}\phi(.) + 0.5\phi(.) - 2x_r + x_r\phi(.) + (\tilde{x} + x_r + 0.5)\tilde{z} - \dot{x}_r - \phi(.)^2 - 2\tilde{z}\phi(.) + (\tilde{x} + x_r + 1)\phi(.) - A_f\tilde{z} \tag{90}$$

$$0 = -(\tilde{x} + x_r + 1 + A_f)\tilde{z} + \tilde{z}^2 \tag{91}$$

Since $\tilde{z} = 0$ is the root of the algebraic solution, the reduced slow subsystem is obtained as

$$\dot{\tilde{x}} = -2\tilde{x} + \tilde{x}\phi(.) + 0.5\phi(.) - 2x_r + x_r\phi(.) - \dot{x}_r - \phi(.)^2 + (\tilde{x} + x_r + 1)\phi(.) \tag{92}$$

Substituting the expression for $\phi(.)$ from Eq. (82) in Eq. (92),

$$\dot{\tilde{x}} = -2\tilde{x} - 2x_r - \dot{x}_r + (2\tilde{x} + 1.5 + 2x_r)(\tilde{x} + x_r + \Gamma_s) - (\tilde{x} + x_r + \Gamma_s)^2 \tag{93}$$

Design Γ_s as

$$\Gamma_s = -\tilde{x} - x_r + \dot{x}_r - A\tilde{x} \tag{94}$$

where A is the feedback gain. Thus, the resulting closed-loop reduced slow subsystem is

$$\dot{\tilde{x}} = -(2 - 2\dot{x}_r + 2Ax_r + 1.5A - 2A\dot{x}_r)\tilde{x} + (-A^2 - 2A)\tilde{x}^2 + (-2x_r + 0.5\dot{x}_r + 2x_r\dot{x}_r - \dot{x}_r^2) \tag{95}$$

where A is the feedback gain. Comparing Eq. (95) with Eq. (50),

$$\mathbf{F_s}(.) = (2 - 2\dot{x}_r + 2Ax_r + 1.5A - 2A\dot{x}_r)\tilde{x} - (-2x_r + 0.5\dot{x}_r + 2x_r\dot{x}_r - \dot{x}_r^2)$$
$$\mathbf{G_s}(.) = (-A^2 - 2A)\tilde{x}^2 \tag{96}$$

Note that this control only guarantees bounded tracking for the slow subsystem. To implement the control laws, substitute for Γ_s from Eq. (94) into Eq. (82):

$$\phi = \dot{x}_r - A\tilde{x} \tag{97}$$

and use Eqs. (85) and (97):

$$\Gamma_f = (-A^2 - A)\tilde{x}^2 + \tilde{x}(\dot{x}_r + 2A\dot{x}_r - Ax_r) + 2A\tilde{x}\,\tilde{z} - \tilde{z}(2\dot{x}_r + A_f) - \dot{x}_r^2 + x_r\dot{x}_r \tag{98}$$

Recall that these controllers are designed using the reduced-order subsystems $\tilde{z} = z - \phi(.)$, where $\phi(.)$ is given by Eq. (97). Then, the control laws Γ_s and Γ_f can be expressed in original coordinates as

$$\Gamma_s = -x + \dot{x}_r - A(x - x_r) \tag{99}$$

$$\Gamma_f = (-A^2 - A)(x - x_r)^2 + (x - x_r)(\dot{x}_r + 2A\dot{x}_r - Ax_r) + 2A(x - x_r)[z - \phi(.)] - [z - \phi(.)](2\dot{x}_r + A_f) - \dot{x}_r^2 + x_r\dot{x}_r \tag{100}$$

Using the manifold condition equation (83),

$$(M\phi)(\tilde{x}, x_r, \dot{x}_r, \Gamma_s) = \epsilon\frac{\partial\phi}{\partial t} + \epsilon\frac{\partial\phi}{\partial\tilde{x}}\{-\mathbf{F_s}(\tilde{x}, x_r, \dot{x}_r) + \mathbf{G_s}(\tilde{x})\} \tag{101}$$

Thus, by the choice of controls Γ_s and Γ_f, $\mathcal{O}[C(\epsilon = 0, \tilde{x}, x_r, \dot{x}_r)] = 0$.

2. *Results and Discussion*

Case 1A: Controller performance for tracking a continuously time-varying sine wave of $0.2\sin(0.2t)$ is presented in Fig. 4. The feedback gains chosen are $A = 3$ and $A_f = 1$. The domains of the errors are $D_x = [-0.3 \quad 0.3]$ and $D_z = [-1.5 \quad 1.5]$. Several constants in Assumptions B–I are computed as $\alpha_1 = 1$, $b_1 = 0.26$, $\beta_1 = 1.4$, $\beta_2 = 30$, $\beta_3 = 0$, $\beta_4 = 0.686$, $\alpha_2 = 1$, $\beta_5 = 1.96$, $\beta_6 = 250$, $\beta_7 = 0.5096$, $\beta_8 = 3.778$, and $\beta_9 = 250$. These values and a choice of $d = 0.3$ results in $\epsilon^* = 2000 \gg 1$. From the simulation results, it is seen that the system response is bounded for all time. Additionally, for simulations with $\epsilon = 0.2$, the bounds

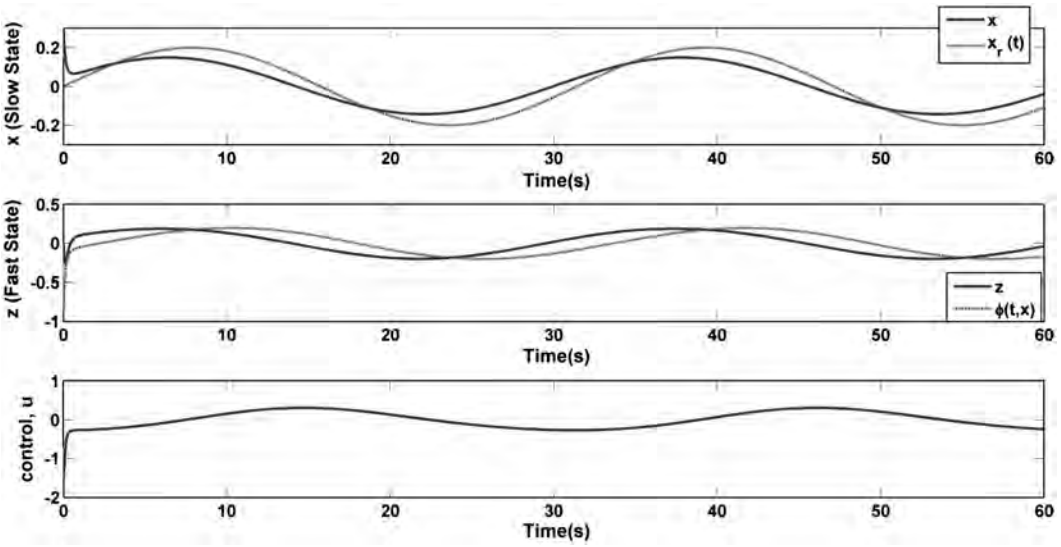


Fig. 4 Case 1A: kinetic slow state compared with specified sine-wave reference, and fast state compared with manifold approximation and computed control.

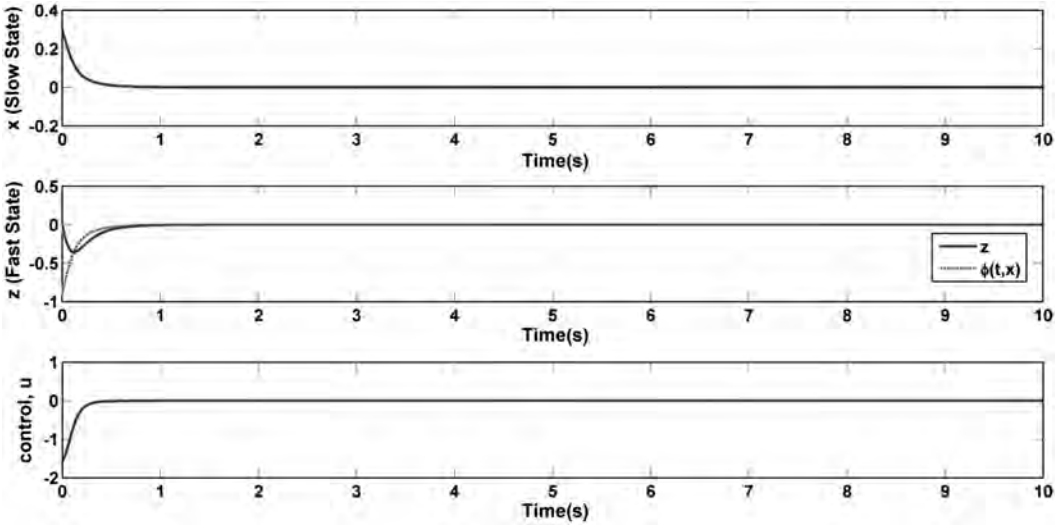


Fig. 5 Case 1B: kinetic slow state, fast state compared with manifold approximation and computed control (regulator problem).

Fig. 6 F/A-18A Hornet external physical characteristics.

$\tilde{x}_b = 0.0818$ and $\tilde{z}_b = 4.701$, and the control is bounded for all time. The initial overshoot may be avoided by adding actuator dynamics and adjusting the feedback gains. Note that the fast state response remains close to its approximation $\phi(t, x)$.

Case 1B: This case simulates the regulator problem with $x_r = 0$ and $\dot{x}_r(t) = 0$. The control laws are the same as derived in Eqs. (99) and (100). The constants $b_1 = 0, \beta_4 = 0, \beta_7 = 0$, and $\beta_8 = 0$, while the other constants have the same values as in Case 1A. With the choice of $d = 0, \epsilon_s^* = 1000 \gg 1$. The results are presented in Fig. 5, which shows that the system asymptotically settles down to the origin.

C. Lateral/Directional Maneuver for F/A-18A Hornet Aircraft

The complete nonlinear dynamic model in the stability axes is represented by the nine states $(M, \alpha, \beta, p, q, r, \phi, \theta, \psi)$ and four controls $(\eta, \delta_e, \delta_a, \delta_r)$. For this example, $[M, \alpha, \beta, \phi, \theta, \psi]^T$ comprise the slow states, and the angular rates $[p, q, r]^T$ comprise the fast states. The aerodynamic database for the symmetric F/A-18A Hornet (seen in Fig. 6) is used [30]. The aerodynamic coefficients are given as analytical functions of the sideslip angle, angle of attack, angular rates, and the control surface deflections. Considering the number of controls available, only three of the six slow states can be controlled. Throttle is maintained constant at $\eta = 0.523$ and is not used as a control. This is a result of using dynamic inversion [31]. The control objective is to perform a 45 deg turn at or near zero sideslip angle while tracking a specified angle-of-attack profile. Pitch attitude angle θ and bank angle ϕ are left uncontrolled.

1. Controller Design

The control laws are developed according to the theory developed in the previous sections. For brevity, only the equations required for incorporating the control law in the simulation are presented here. Since the aircraft equations of motion are highly coupled, the first step is to transform them into slow and fast sets. Let $\mathbf{x} = [\alpha, \beta, \psi]^T$ represent the subset of the slow states and $\mathbf{u} = [\delta_e, \delta_a, \delta_r]^T$ represent the control variables,

$$\dot{\mathbf{x}} = \underbrace{\mathbf{f_1}(\mathbf{x}, M, \theta, \phi) + \mathbf{f_2}(\mathbf{x}, \theta, \phi)\mathbf{z}}_{\mathbf{f}(.)} + \mathbf{g}(\mathbf{x}, M)\mathbf{u} \tag{102}$$

$$\epsilon\dot{\mathbf{z}} = \underbrace{\boldsymbol{l}(\mathbf{z}) + \boldsymbol{l}(\mathbf{x}, M) + \boldsymbol{l}(\mathbf{x}, M)\mathbf{z321}}_{\boldsymbol{l}(.)} + \mathbf{k}(\mathbf{x}, M)\mathbf{u} \tag{103}$$

The parameter ϵ is introduced on the left-hand side of Eq. (103) to indicate the time-scale difference between body-axis angular rates and the other states [14]. In the translational equations of motion, functions such as gravitational forces and aerodynamic forces due to angle of attack and sideslip angle are collectively represented as $\mathbf{f_1}(\mathbf{x}, M, \theta, \phi)$. Terms in the translational equations of motion due to the cross products between the angular rates and the slow states are labeled $\mathbf{f_2}(\mathbf{x}, \theta, \phi)\mathbf{z}$. The remaining terms in the slow state equations are the control effectiveness terms labeled $\mathbf{g}(\mathbf{x}, M)$. The nonlinearity in the fast dynamics due to the cross product between the angular rates is represented by $\boldsymbol{l}(\mathbf{z})\mathbf{1}$. The aerodynamic moment terms that depend solely upon the slow state are denoted as $\boldsymbol{l}(\mathbf{x}, M)\mathbf{2}$, and the aerodynamic moment terms that depend linearly on the angular rates are denoted as $\boldsymbol{l}(\mathbf{x})\mathbf{3}$. The term $\mathbf{k}(\mathbf{x}, M)$ is the control effectiveness term in the angular rate dynamics. The exact form of these functions is derived in the Appendix. Define the errors $\tilde{\mathbf{x}} = \mathbf{x} - \mathbf{x}_r$ and $\tilde{\mathbf{z}} = \mathbf{z} - \mathbf{h}(\tilde{\mathbf{x}}, \epsilon, \mathbf{x}_r, \dot{\mathbf{x}}_r, M)$ and transform Eqs. (102) and (103) into error coordinates equivalent to Eqs. (39) and (40):

$$\dot{\tilde{\mathbf{x}}} = \underbrace{\mathbf{f_1}(\tilde{\mathbf{x}} + \mathbf{x}_r, M, \theta, \phi) + \mathbf{f_2}(\tilde{\mathbf{x}} + \mathbf{x}_r, \theta, \phi)[\tilde{\mathbf{z}} + \mathbf{h}(.)]}_{\mathbf{f}(.)} + \mathbf{g}(\tilde{\mathbf{x}} + \mathbf{x}, M)[\Gamma_\mathbf{s} + \Gamma_\mathbf{f}] - \dot{\mathbf{x}}_r \tag{104}$$

$$\epsilon\dot{\tilde{\mathbf{z}}} = \underbrace{\boldsymbol{l}[\tilde{\mathbf{z}} + \mathbf{h}(.)] + \boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M) + \boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)[\tilde{\mathbf{z}} + \mathbf{h}(.)]\mathbf{321}}_{l(.)} + \mathbf{k}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)[\Gamma_\mathbf{s} + \Gamma_\mathbf{f}] - \epsilon\frac{\partial \mathbf{h}}{\partial t} - \epsilon\frac{\partial \mathbf{h}}{\partial \tilde{\mathbf{x}}}\dot{\tilde{\mathbf{x}}} - \epsilon\frac{\partial \mathbf{h}}{\partial M}\dot{M} \tag{105}$$

Note that, for an aircraft example, the manifold will also be a function of Mach number. Let $\Phi(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_\mathbf{s}, M)$ be the approximate manifold. Then, Eq. (38) expresses the manifold condition. In this case, select

$$\boldsymbol{\Phi}(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_\mathbf{s}) = -\boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)[\boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M) + \mathbf{k}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)\Gamma_\mathbf{s}]\mathbf{23} - 1 \tag{106}$$

such that

$$(M\Phi)(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_\mathbf{s}) = \epsilon\frac{\partial \Phi}{\partial t} + \epsilon\frac{\partial \boldsymbol{\Phi}}{\partial \tilde{\mathbf{x}}}\dot{\tilde{\mathbf{x}}} + \epsilon\frac{\partial \boldsymbol{\Phi}}{\partial M}\dot{M} - \boldsymbol{l}[\boldsymbol{\Phi}(.)] - \mathbf{k}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)\Gamma_\mathbf{f}\mathbf{1} \tag{107}$$

To design Γ_f, develop the reduced fast subsystem,

$$\tilde{\mathbf{x}}' = \mathbf{0} \tag{108}$$

$$\tilde{\mathbf{z}}' = \boldsymbol{l}[\tilde{\mathbf{z}} + \boldsymbol{\Phi}(.)] + \boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M) + \boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)[\tilde{\mathbf{z}} + \boldsymbol{\Phi}(.)] + \mathbf{k}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)[\Gamma_\mathbf{s} + \Gamma_\mathbf{f}]\mathbf{321} \tag{109}$$

Using dynamic inversion and Eq. (106), design

$$\Gamma_\mathbf{f} = \mathbf{k}^{-1}(\mathbf{x} + \mathbf{x}_r, M)\{-A_f\tilde{\mathbf{z}} - \boldsymbol{l}[\tilde{\mathbf{z}} + \Phi(.)] - \boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)\tilde{\mathbf{z}}\}\mathbf{31} \tag{110}$$

where A_f is the chosen feedback gain. Then, the closed-loop reduced subsystem becomes

$$\tilde{\mathbf{z}}' = -A_f\tilde{\mathbf{z}} \tag{111}$$

Comparing with Eq. (48),

$$\mathbf{L_f}(.) = A_f\tilde{\mathbf{z}} \tag{112}$$

$$\mathbf{K_f}(.) = 0 \tag{113}$$

Similarly, develop the reduced-order slow subsystem,

$$\dot{\tilde{\mathbf{x}}} = \mathbf{f_1}(\tilde{\mathbf{x}} + \mathbf{x}_r, M, \theta, \phi) - \mathbf{f_2}(\tilde{\mathbf{x}} + \mathbf{x}_r, \theta, \phi)\boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)\boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M) - \mathbf{g}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)\mathbf{k}^{-1}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)\boldsymbol{l}(\Phi) - \dot{\mathbf{x}}_r + [-\mathbf{f_2}(\tilde{\mathbf{x}} + \mathbf{x}_r, \theta, \phi)\boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)^{-1}\mathbf{k}(\tilde{\mathbf{x}} + \mathbf{x}_r, M) + \mathbf{g}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)]\Gamma_s\mathbf{3123} - 1 \tag{114}$$

Then, the control law for the reduced slow subsystem is computed as

$$\Gamma_\mathbf{s} = \mathbf{B}^{-1}\{-A\tilde{\mathbf{x}} + \dot{\mathbf{x}}_r\} + \mathbf{B}^{-1}\{-\mathbf{f_1}(\tilde{\mathbf{x}} + \mathbf{x}_r, M, \theta, \phi) + \mathbf{f_2}(\tilde{\mathbf{x}} + \mathbf{x}_r, \theta, \phi)\boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)\boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)\}\mathbf{23} - 1 \tag{115}$$

where

$$\mathbf{B} = [-\mathbf{f_2}(\tilde{\mathbf{x}} + \mathbf{x}_r, \theta, \phi)\boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)^{-1}\mathbf{k}(\tilde{\mathbf{x}} + \mathbf{x}_r, M) + \mathbf{g}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)]\mathbf{3}$$

A is the feedback gain, and the resulting closed-loop system is

$$\dot{\tilde{\mathbf{x}}} = -A\tilde{\mathbf{x}} - \mathbf{g}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)\mathbf{k}^{-1}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)\boldsymbol{l}[\Phi(.)]\mathbf{1} \tag{116}$$

where $\Phi(.)$ is obtained from Eq. (106). Note by the choice of Γ_f, Eq. (107) becomes

$$(M\Phi)(\tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \Gamma_\mathbf{s}) = \epsilon\frac{\partial \Phi}{\partial t} + \epsilon\frac{\partial \boldsymbol{\Phi}}{\partial \tilde{\mathbf{x}}}\dot{\tilde{\mathbf{x}}} + \epsilon\frac{\partial \boldsymbol{\Phi}}{\partial M}\dot{M} \tag{117}$$

and thus $\mathcal{O}[C(\epsilon = 0, \tilde{\mathbf{x}}, \mathbf{x}_r, \dot{\mathbf{x}}_r)] = 0$. Furthermore, since the aerodynamic moments are a function of the angular rates, matrix $\boldsymbol{l}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)\mathbf{3}$ is full rank. The control effectiveness terms $\mathbf{k}(\tilde{\mathbf{x}} + \mathbf{x}_r, M)$ represent the aerodynamic moment coefficients due to control effector deflections, which are nonzero.

Remark 8: The aircraft example assumes that the Mach number, pitch attitude angle, and bank angle are input stabilizable. Although the angular rates are bounded by the reference trajectory, the Euler angles remain bounded through the exact kinematic relationships. Additionally, since the angle of attack is being tracked and thrust remains constant, the Mach number remains bounded.

2. *Results and Discussion*

Case 2: The specified maneuver is a 45 deg turn at or near zero sideslip angle while simultaneously tracking a step input in the angle of

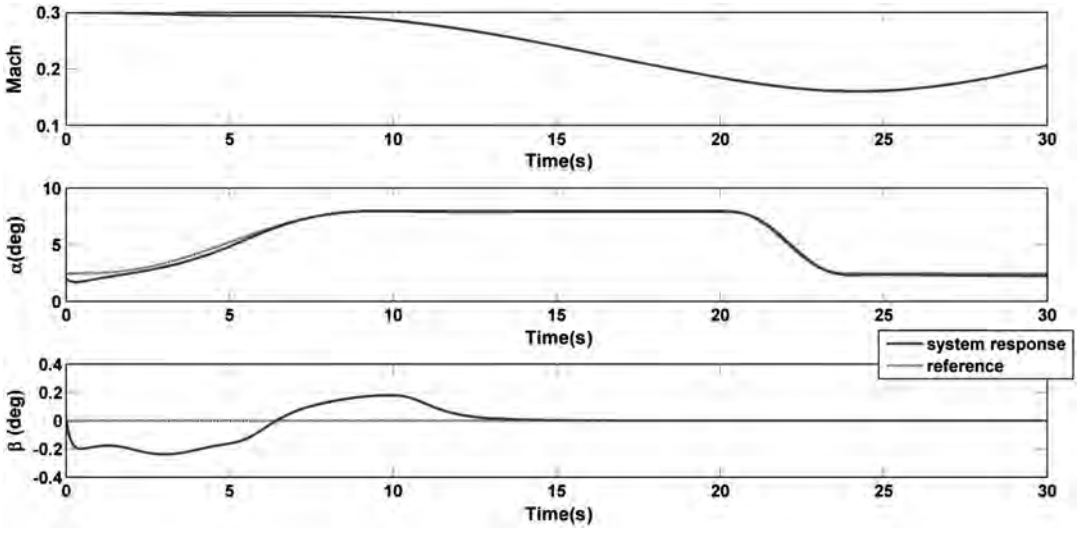


Fig. 7 Case 2: F/A-18A Mach number, angle of attack, and sideslip angle responses; 0.3/20k.

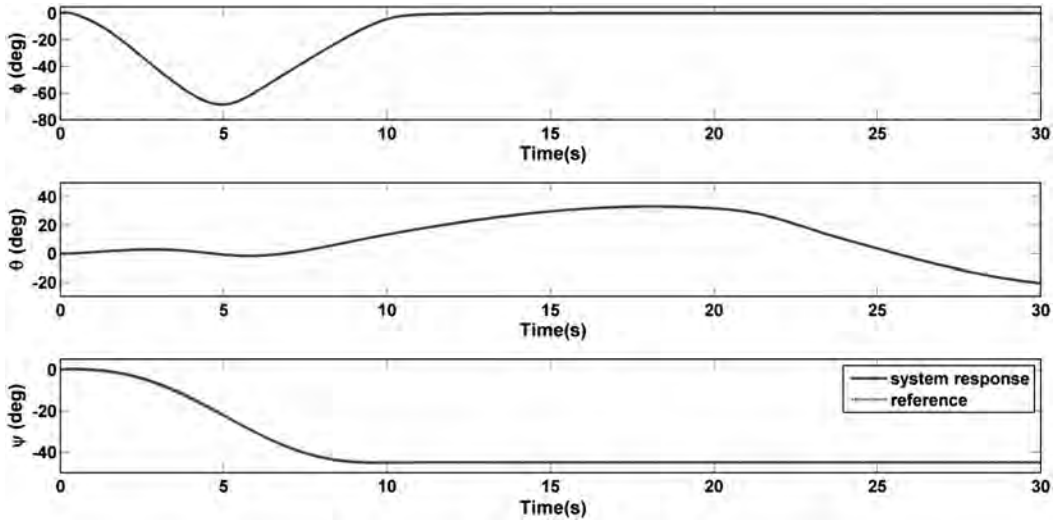


Fig. 8 Case 2: F/A-18A kinematic angle responses; 0.3/20k.

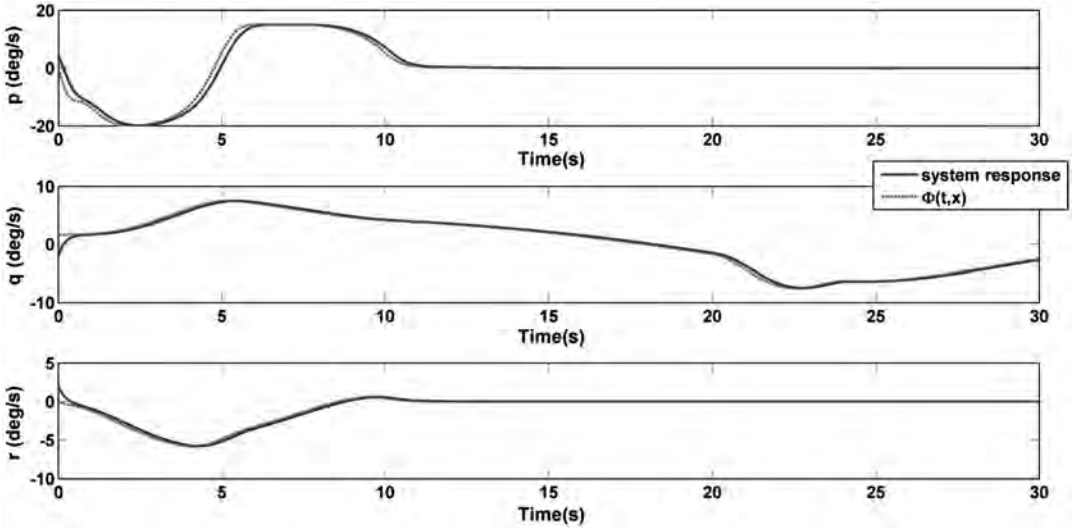


Fig. 9 Case 2: F/A-18A angular rate responses; 0.3/20k.

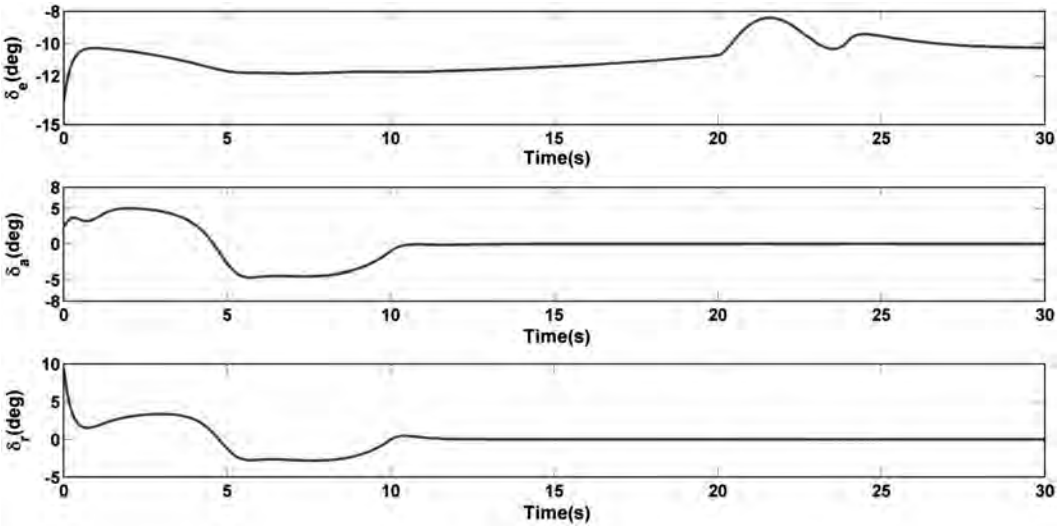


Fig. 10 Case 2: F/A-18A control responses; 0.3/20k.

attack. The flight condition is Mach 0.3 at 20,000 ft altitude (0.3/20k). The trim and initial conditions are $\alpha(1) = 2$ deg, $p(0) = 4$ deg/s, $q(0) = -2$ deg/s, and $r(0) = 2$ deg/s. The feedback gain matrices are

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad A_f = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 5 \end{bmatrix} \tag{118}$$

Theorem 1 guarantees the existence of the bound ϵ^*, but the nonlinearity of this example restricts its analytical computation. Note also that, for an aircraft, the parameter ϵ is normally only introduced in the modeling stage to take advantage of the presence of different time scales in the system. In reality, this parameter is a function of the flight condition and is difficult to quantify. Thus, it is advantageous to derive and implement controllers that do not require knowledge of this parameter.

Figures 7–10 evaluate control law performance for the specified maneuver. After initial transients settle out, the angle of attack, sideslip angle, and heading angle states closely track the reference. The angle-of-attack error is within ± 0.2 deg, and the sideslip angle tracking error is within ± 0.2 deg throughout the maneuver. The heading angle is maintained within ± 0.25 deg. Close tracking of the slow states implies that the fast states are successfully being driven onto the approximate manifold, as is seen in Fig. 9. The angular rates are smooth, and errors are within ± 2 deg/s. The control surface deflections are within bounds and generate the desired nonzero angular rates. The uncontrolled states M,θ, and ϕ are well behaved and remain bounded throughout the maneuver.

VII. Conclusions

A control formulation for tracking the slow states of a general class of nonlinear singularly perturbed systems was developed based upon the study of its geometric constructs. For a given set of nonlinear algebraic equations, an approximate analytical form of the system manifold was computed. Control laws for each of the subsystems and boundedness of closed-loop signals was demonstrated with a composite Lyapunov function approach, and asymptotic stabilization was shown for the general class of nonlinear singularly perturbed systems. Controller performance was demonstrated through numerical simulation for two nonlinear examples.

Based upon the results presented in the paper, tracking error for the nonlinear planar example was demonstrated to remain within $|0.08|$ at all times, as predicted by the analytically computed bound. It was also shown that, for all values of ϵ, the controller maintains bounded stability and the asymptotic convergence of the errors to origin for the regulator problem. Nonlinear simulations of an F/A-18A Hornet demonstrate that the controller is capable of closely tracking heading, sideslip angle, and angle of attack. The angular rates were within bounds and seen to track the desired manifold approximations well. Even though the Mach number, bank angle, and pitch attitude angle were not controlled, their magnitudes remained bounded as expected. The aircraft example demonstrates the advantage of developing controllers independent of the scalar perturbation parameter ϵ.

Appendix

The nonlinear mathematical model of the aircraft is represented by the following dynamic and kinematic equations:

$$\dot{M} = \frac{1}{mv_s}\left[T_m\eta\cos\alpha\cos\beta - \frac{1}{2}C_D(\alpha, q, \delta e)\rho v_s^2 M^2 S - mg\sin\gamma\right] \tag{A1}$$

$$\dot{\alpha} = q - \frac{1}{\cos\beta}\{(p\cos\alpha + r\sin\alpha)\sin\beta\} + \frac{1}{\cos\beta}\left\{\frac{1}{mv_s M}\left[T_m\eta\sin\alpha + \frac{1}{2}C_L(\alpha, q, \delta e)\rho v_s^2 M^2 S - mg\cos\mu\cos\gamma\right]\right\} \tag{A2}$$

$$\dot{\beta} = \frac{1}{mv_s M}\left[-T_m\eta\cos\alpha\sin\beta + \frac{1}{2}C_Y(\beta, p, r, \delta e, \delta a, \delta r)\rho v_s^2 M^2 S + mg\sin\mu\cos\gamma\right] + (p\sin\alpha - r\cos\alpha) \tag{A3}$$

$$\dot{p} = \frac{I_y - I_z}{I_x}qr + \frac{1}{2I_x}\rho v_s^2 M^2 SbC_l(\beta, p, r, \delta e, \delta a, \delta r) \tag{A4}$$

$$\dot{q} = \frac{I_z - I_x}{I_y}pr + \frac{1}{2I_y}\rho v_s^2 M^2 ScC_m(\alpha, q, \delta e) \tag{A5}$$

$$\dot{r} = \frac{I_x - I_y}{I_z}pq + \frac{1}{2I_z}\rho v_s^2 M^2 SbC_n(\beta, p, r, \delta e, \delta a, \delta r) \tag{A6}$$

$$\dot{\phi} = p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \tag{A7}$$

$$\dot{\theta} = q\cos\phi - r\sin\phi \tag{A8}$$

$$\dot{\psi} = (q\sin\phi + r\cos\phi)\sec\theta \tag{A9}$$

Wind-axes orientation angles μ and γ are defined as follows:

$$\sin\gamma = \cos\alpha\cos\beta\sin\theta - \sin\beta\sin\phi\cos\theta - \sin\alpha\cos\beta\cos\phi\cos\theta \tag{A10}$$

$$\sin\mu\cos\gamma = \sin\theta\cos\alpha\sin\beta + \sin\phi\cos\theta\cos\beta - \sin\alpha\sin\beta\cos\phi\cos\theta \tag{A11}$$

$$\cos\mu\cos\gamma = \sin\theta\sin\alpha + \cos\alpha\cos\phi\cos\theta \tag{A12}$$

To write the equations in the form of Eqs. (102) and (103),

$$\mathbf{f_1}(\mathbf{x}, M, \theta, \phi) = \begin{bmatrix} -\frac{1}{mv_s M\cos\beta}\left[\frac{1}{2}C_L(\alpha)\rho v_s^2 M^2 S - mg\cos\mu\cos\gamma\right] \\ \frac{1}{mv_s M}\left[\frac{1}{2}C_Y(\beta)\rho v_s^2 M^2 S + mg\sin\mu\cos\gamma\right] \\ 0 \end{bmatrix} \tag{A13}$$

$$\mathbf{f_2}(\mathbf{x}, \theta, \phi) = \begin{bmatrix} -\cos\alpha\tan\beta & 1 & -\sin\alpha\tan\beta \\ \sin\alpha & 0 & -\cos\alpha \\ 0 & \sec\theta\sin\phi & \cos\phi\sec\theta \end{bmatrix} \tag{A14}$$

$$\mathbf{g}(\mathbf{x},M) = \begin{bmatrix} -\frac{1}{2\cos\beta}\rho(v_s M)^2 S C_{L_{\delta_e}} & 0 & 0 \\ 0 & \frac{1}{2mv_s M} C_{Y_{\delta_a}} \rho v_s^2 M^2 S & \frac{1}{2mv_s M} C_{Y_{\delta_r}} \rho v_s^2 M^2 S \\ 0 & 0 & 0 \end{bmatrix} \tag{A15}$$

$$\boldsymbol{l}(\mathbf{z}) = \begin{bmatrix} \frac{I_y - I_z}{I_x} qr \\ \frac{I_z - I_x}{I_y} pr \\ \frac{I_x - I_y}{I_z} pq \end{bmatrix} \mathbf{1} \tag{A16}$$

$$\boldsymbol{l}(\mathbf{x},M) = \begin{bmatrix} \frac{1}{2I_x}\rho v_s^2 M^2 S b C_l(\beta) \\ \frac{1}{2I_y}\rho v_s^2 M^2 S c C_m(\alpha) \\ \frac{1}{2I_z}\rho v_s^2 M^2 S b C_n(\beta) \end{bmatrix} \mathbf{2} \tag{A17}$$

$$\boldsymbol{l}(\mathbf{x},M) = \begin{bmatrix} \frac{1}{2I_x}\rho v_s^2 M^2 S b C_{l_p} & 0 & \frac{1}{2I_x}\rho v_s^2 M^2 S b C_{l_r} \\ 0 & \frac{1}{2I_y}\rho v_s^2 M^2 S c C_{m_q} & 0 \\ \frac{1}{2I_z}\rho v_s^2 M^2 S b C_{n_p} & 0 & \frac{1}{2I_z}\rho v_s^2 M^2 S b C_{n_r} \end{bmatrix} \mathbf{3} \tag{A18}$$

$$\mathbf{k}(\mathbf{x},M) = \begin{bmatrix} 0 & \frac{1}{2I_x}\rho v_s^2 M^2 S b C_{l_{\delta_a}} & \frac{1}{2I_x}\rho v_s^2 M^2 S b C_{l_{\delta_a}} \\ \frac{1}{2I_y}\rho v_s^2 M^2 S c C_{m_{\delta_e}} & 0 & 0 \\ 0 & \frac{1}{2I_z}\rho v_s^2 M^2 S b C_{n_{\delta_a}} & \frac{1}{2I_z}\rho v_s^2 M^2 S b C_{n_{\delta_r}} \end{bmatrix} \tag{A19}$$

Acknowledgments

This material is based upon work supported in part by the U.S. Air Force Office of Scientific Research under contract FA9550-08-1-0038, with technical monitor Fariba Fahroo. This support is gratefully acknowledged by the authors. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Air Force.

References

[1] Prandtl, L., "Uber Ussigkeitsbewegung bei Kleiner Reibung," *Verhandlungen, III International Mathematical Kongresses*, Tuebner, Leipzig, 1905, pp. 484–491.

[2] Tikhonov, A. N., "On the Dependence of the Solutions of Differential Equations on a Small Parameter," *Matematicheskii Sbornik*, Vol. 22, No. 64, 1948, pp. 193–204.

[3] Vasileva, A. B., "Asymptotic Behaviour of Solutions to Certain Problems Involving Nonlinear Differential Equations Containing a Small Parameter Multiplying the Highest Derivative," *Russian Mathematical Surveys*, Vol. 18, No. 3, 1963, pp. 13–81. doi:10.1070/RM1963v018n03ABEH001137

[4] Wasow, W., *Asymptotic Expansions for Ordinary Differential Equations*, Wiley-Interscience, New York, 1965, pp. 249–268.

[5] Kokotovic, P. V., and Sannuti, P., "Singular Perturbation Method for Reducing Model Order in Optimal Control Design," *IEEE Transactions on Automatic Control*, Vol. 13, No. 4, 1968, pp. 377–384. doi:10.1109/TAC.1968.1098927

[6] Grujic, L. T., "On Tracking Domains of Continuous-Time Non-Linear Control Systems," *RAIRO Automatique: Systems Analysis and Control*, Vol. 16, No. 4, 1982, pp. 311–327.

[7] Grujic, L. T., "On the Theory and Synthesis of Nonlinear Non-Stationary Tracking Singularly Perturbed Systems," *Control Theory and Advanced Technology*, Vol. 4, No. 4, 1988, pp. 395–409.

[8] Christofides, P. D., Teel, A. R., and Daoutidis, P., "Robust Semi-Global Output Tracking for Nonlinear Singularly Perturbed Systems," *International Journal of Control*, Vol. 65, No. 4, 1996, pp. 639–666. doi:10.1080/00207179608921714

[9] Heck, B. S., "Sliding-Model Control for Singularly Perturbed Systems," *International Journal of Control*, Vol. 53, No. 4, 1991, pp. 985–1001. doi:10.1080/00207179108953660

[10] Suzuki, M., and Miura, M., "Stabilizing Feedback Controllers for Singularly Perturbed Linear Constant Systems," *IEEE Transactions on Automatic Control*, Vol. 21, No. 1, 1976, pp. 123–124. doi:10.1109/TAC.1976.1101151

[11] Vidyasagar, M., "Robust Stabilization of Singularly Perturbed Systems," *Systems and Control Letters*, Vol. 5, No. 6, 1985, pp. 413–418. doi:10.1016/0167-6911(85)90066-0

[12] Khalil, H. K., "Feedback Control of Nonstandard Singularly Perturbed Systems," *IEEE Transactions on Automatic Control*, Vol. 34, No. 10, 1989, pp. 1052–1060. doi:10.1109/9.35275

[13] Phillips, R. G., "A Two-Stage Design of Linear Feedback Controls," *IEEE Transactions on Automatic Control*, Vol. 25, No. 6, 1980, pp. 1220–1223. doi:10.1109/TAC.1980.1102548

[14] Menon, P., Badgett, M. E., and Walker, R., "Nonlinear Flight Test Trajectory Controllers for Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 10, No. 1, 1987, pp. 67–72. doi:10.2514/3.20182

[15] Snell, S. A., Enns, D. F., and Garrard, W. L., Jr., "Nonlinear Inversion Flight Control for a Supermaneuverable Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 4, 1992, pp. 976–984. doi:10.2514/3.20932

[16] Mulgund, S. S., and Stengel, R. F., "Aircraft Flight Control in Wind Shear Using Sequential Dynamic Inversion," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 5, 1995, pp. 1084–1091. doi:10.2514/3.21508

[17] Georgie, J., and Valasek, J., "Evaluation of Longitudinal Desired Dynamics for Dynamic-Inversion Controlled Generic Reentry Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 5, 2003, pp. 811–819. doi:10.2514/2.5116

[18] Fenichel, N., "Geometric Singular Perturbation Theory for Ordinary Differential Equations," *Journal of Differential Equations*, Vol. 31, No. 1, 1979, pp. 53–98. doi:10.1016/0022-0396(79)90152-9

[19] Marino, R., and Kokotovic, P., "A Geometric Approach to Nonlinear Singularly Perturbed Control Problems," *Automatica*, Vol. 24, No. 1, 1988, pp. 31–41. doi:10.1016/0005-1098(88)90005-2

[20] Sobolev, V., "Integral Manifolds and Decomposition of Singularly Perturbed Systems," *Systems and Control Letters*, Vol. 5, No. 3, 1984, pp. 169–179. doi:10.1016/S0167-6911(84)80099-7

[21] Sedighizadeh, M., and Rezazadeh, A., "A Wind Farm Reduced Order Model Using Integral Manifold Theory," Vol. 37, World Academy of Science, Engineering and Technology, U. K., 2008, pp. 263–268.

[22] Sharkey, P., and O'Reilly, J., "Exact Design Manifold Control of a Class of Nonlinear Singularly Perturbed Systems," *IEEE Transactions on Automatic Control*, Vol. 32, No. 10, 1987, pp. 933–935. doi:10.1109/TAC.1987.1104469

[23] Chen, C., "Global Exponential Stabilisation for Nonlinear Singularly Perturbed Systems," *IEE Proceedings: Control Theory and Applications*, Vol. 145, No. 4, 1998, pp. 377–382. doi:10.1049/ip-cta:19981984

[24] Nayfeh, A. H., *Introduction to Perturbation Techniques*, Wiley-Interscience, New York, 1993, pp. 270–279.

[25] Carr, J., *Applications of Centre Manifold Theory*, Vol. 35, Applied Mathematical Sciences, Springer, New York, 1981, pp. 5–8.

[26] Cronin, J., and Robert E. O'Malley, J. (eds.), *Analyzing Multiscale Phenomena Using Singular Perturbation Methods: Proceedings of Symposia in Applied Mathematics*, Vol. 56, American Mathematical Soc., Providence, RI, 1986, pp. 85–131.

[27] Kokotovic, P., Khalil, H. K., and Reilly, J. O., *Singular Perturbation Methods in Control: Analysis and Design*, Academic Press, New York, 1986, pp. 315–318.

[28] Saberi, A., and Khalil, H., "Stabilization and Regulation of Nonlinear Singularly Perturbed Systems-Composite Control," *IEEE Transactions on Automatic Control*, Vol. 30, No. 8, 1985, pp. 739–747. doi:10.1109/TAC.1985.1104064

[29] Choi, H.-L., and Lim, J.-T., "Gain Scheduling Control of Nonlinear Singularly Perturbed Time-Varying Systems with Derivative Information," *International Journal of Systems Science*, Vol. 36, No. 6, 2005, pp. 357–364. doi:10.1080/00207720500111707

[30] Fan, Y., Lutze, F. H., and M. Cliff, E., "Time-Optimal Lateral Maneuvers of an Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 5, 1995, pp. 1106–1112. doi:10.2514/3.21511

[31] Ito, D., Georgie, J., and Valasek, J., "Re-Entry Vehicle Flight Control Design Guidelines: Dynamic Inversion," NASA TP 2002-210771, 2002.

Proceedings of the CEAS EuroGNC 2011, 13-15 April, Munich, Germany

Global Tracking Control Structures for Nonlinear Singularly Perturbed Aircraft Systems

Anshu Siddarth and John Valasek

Abstract The problem of simultaneous tracking of both fast and slow states for a general class of nonlinear singularly perturbed systems is addressed. A motivating example is an aircraft tracking a prescribed fast moving target, while simultaneously regulating speed and/or one or more kinematic angles. Previous results in the literature have focused on tracking outputs that are a function of the slow states alone. Moreover, global asymptotic tracking has been demonstrated only for a class of nonlinear systems that possess a unique real root for the fast states. For a more general class of nonlinear systems only local tracking results have been proven. In this paper, control laws that accomplish global tracking of both fast and slow states is developed using geometric singular perturbation methods. Global exponential stability is proven using the composite Lyapunov function approach and an upper bound necessary condition for the scalar perturbation parameter is derived. Controller performance is demonstrated through simulation of a combined longitudinal lateral/directional maneuver for a nonlinear, coupled, six degree-of-freedom model of the F/A-18A Hornet. Results presented in the paper show that the controller accomplishes global asymptotic tracking even though the desired reference trajectory requires the aircraft to switch between linear and nonlinear regimes. Asymptotic tracking while keeping all the closed-loop signals bounded and well behaved is also demonstrated. Additionally the controller is independent of the scalar perturbation parameter nor does it require knowledge of it.

1 Introduction

This paper addresses systems that possess both slow and fast dynamics. This multiple time-scale behaviour is either a system characteristic (for example, aircraft

Anshu Siddarth, John Valasek
Vehicle Systems & Control Laboratory, Department of Aerospace Engineering, Texas A&M University, College Station, TX 77843-3141 e-mail: anshun1@tamu.edu,valasek@tamu.edu

and flexible beam structures) or arises due to implementation of a fast controller (for example, systems with fast actuators and/or fast sensors). The control objective is to develop a stable tracker for these two time-scale systems that would regulate both slow and fast states simultaneously. The singular perturbation approach[13] has been the foremost technique employed in the literature to examine the behaviour of these two time-scale systems. In this approach, the system dynamics are approximated by two lower-order subsystems. The slow subsystem captures the dominant phenomena assuming that the fast variables evolve infinitely many times faster, and have settled down onto a manifold. The fast subsystem addresses the neglected phenomena, and assumes that the slow variables remain constant. It has been shown that the complete system behaviour can be approximated by the dynamics of the slow subsystem provided the fast subsystem is uniformly asymptotically stable about the manifold [6, 10]. These results of singular perturbation methods have made it the most favourable model-reduction technique in the control literature[14].

The design of nonlinear tracking control laws for the slow variables using singular perturbation methods has received a lot of attention in the past. The typical methodology is to design two separate controllers for each of the two subsystems, and then apply their composite or sum to the full-order system. A tracking control law is designed for the slow subsystem whereas a stabilizing controller is designed for the fast subsystem. This is done to restrict the fast variables onto a manifold. Global asymptotic tracking of the composite control structure is guaranteed only if the manifold is unique. This manifold is the set of fixed points of the fast dynamics expressed as a smooth function of the slow variables and the control inputs; hence it is not always unique. To enforce the uniqueness condition, previous studies in the literature have:

1. Assumed that the system has a unique manifold[4, 8]
2. Considered nonlinear systems that have a unique manifold. This is satisfied by two time-scale systems that are nonlinear in the slow states and linear in the fast states[11]

For a general class of nonlinear systems such as aircraft, approximate approaches that guarantee local stability have been proposed. One approach is to consider the fast variables as control inputs for the slow subsystem. Reference[12] used this approach to design nonlinear flight test trajectories for velocity, angle-of-attack, sideslip angle and altitude by using the fast angular rates as the control variables. This control was augmented with an outer-loop controller that determines the control surface deflections needed to ensure that the angular rates track the output of the inner-loop. More recently the same concept has been employed for the control of generic reentry vehicles[7]. Another approach proposed in Reference[16] considered the general class of nonlinear singularly perturbed systems and computed local approximations of the manifold that helped conclude local stability for the complete system.

All of the approaches discussed above demonstrate slow state tracking either locally or globally by restricting the fast states, and, they address the output tracking problem for two time-scale systems with fast actuators. But for systems whose dy-

namics inherently possess different time-scales, both the slow and the fast states constitute the output vector. For example, during air combat maneuvering an aircraft is typically required to track a fast moving target while regulating speed (slow variable) and/or one or more kinematic and aerodynamic angles. In this case the fast states cannot be restricted to simply stabilize onto a manifold. The reduced-order approach therefore appears to be inadequate for a general class of output tracking problem. Reference[1] formulated optimal control laws to accomplish fast state tracking using invariant measures for systems with oscillatory fast dynamics.

In this paper, state feedback control laws are developed for a general class of nonlinear singularly perturbed systems to accomplish slow and fast state tracking simultaneously. The paper makes two major contributions. First, the approach developed here employs the reduced-order technique without imposing any assumptions about the fast manifold. This is done by extending the previous work of the authors[16] so as to not require computation of the manifold. Second, global exponential tracking is proved using the composite Lyapunov approach[10]. From the stability analysis it is shown that this approach applies to all classes of singularly perturbed systems, with the global exponential stabilization results of a class of singularly perturbed systems being a special case[3]. Additionally, the technique is independent of the scalar perturbation parameter and an upper bound on this parameter is derived as a necessary condition for stability results to hold. These results are demonstrated by simulation for a nonlinear, coupled, six degree-of-freedom model of the F/A-18A Hornet.

The paper is organized as follows. Section 2 mathematically formulates the control problem and briefly reviews the necessary concepts for model reduction from geometric singular perturbation theory. Control laws and the main results of the paper are detailed in Section 3. Section 4 presents simulation results, and conclusions are presented in Section 5.

2 Problem Formulation and Model Reduction

The following nonlinear singularly perturbed model represents the class of two time-scale dynamical systems addressed in this paper

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{z}) + \mathbf{g}(\mathbf{x},\mathbf{z})\mathbf{u} \tag{1}$$

$$\varepsilon\dot{\mathbf{z}} = \mathbf{l}(\mathbf{x},\mathbf{z}) + \mathbf{k}(\mathbf{x},\mathbf{z})\mathbf{u} \tag{2}$$

$$\mathbf{y} = \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix} \tag{3}$$

where $\mathbf{x} \in \mathbb{R}^m$ is the vector of slow variables, $\mathbf{z} \in \mathbb{R}^n$ is the vector of fast variables, $\mathbf{u} \in \mathbb{R}^p$ is the input vector and $\mathbf{y} \in \mathbb{R}^{m+n}$ is the output vector. $\varepsilon \in \mathbb{R}^+$ is the singular perturbation parameter that satisfies $0 < \varepsilon << 1$. The vector fields $\mathbf{f}(.), \mathbf{g}(.), \mathbf{l}(.)$ and $\mathbf{k}(.)$ are assumed to be sufficiently smooth and $p \geq (m+n)$. The control objective is to drive the output so as to track sufficiently smooth, bounded, time-varying trajectories, such that $\mathbf{x}(t) \to \mathbf{x}_r(t)$ and $\mathbf{z}(t) \to \mathbf{z}_r(t)$ as $t \to \infty$.

2.1 Reduced-order Modeling

The singularly perturbed model considered in Eqs.1,2 is expressed in the *slow time scale* t. In this time-scale the slow states evolve at an ordinary rate whereas the fast states move at a rate of $O\left(\frac{1}{\varepsilon}\right)$. This system can be equivalently expressed in the *fast time-scale* τ such that the fast states evolve at an ordinary rate and the slow variables move slowly at a rate of $O(\varepsilon)$

$$\mathbf{x}' = \varepsilon\left[\mathbf{f}(\mathbf{x},\mathbf{z}) + \mathbf{g}(\mathbf{x},\mathbf{z})\mathbf{u}\right] \tag{4}$$

$$\mathbf{z}' = \mathbf{l}(\mathbf{x},\mathbf{z}) + \mathbf{k}(\mathbf{x},\mathbf{z})\mathbf{u} \tag{5}$$

where $'$ represents a derivative with respect to $\tau = \frac{t-t_0}{\varepsilon}$ and t_0 is the initial time. Geometric singular perturbation theory[6] examines the behaviour of these singularly perturbed systems by studying the geometric constructs of reduced-order models obtained by substituting $\varepsilon = 0$ in Eqs.1,2 and Eqs.4,5. This results in the *Reduced Slow Subsystem*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{z}) + \mathbf{g}(\mathbf{x},\mathbf{z})\mathbf{u} \tag{6}$$

$$\mathbf{0} = \mathbf{l}(\mathbf{x},\mathbf{z}) + \mathbf{k}(\mathbf{x},\mathbf{z})\mathbf{u} \tag{7}$$

and the *Reduced Fast Subsystem*

$$\mathbf{x}' = \mathbf{0} \tag{8}$$

$$\mathbf{z}' = \mathbf{l}(\mathbf{x},\mathbf{z}) + \mathbf{k}(\mathbf{x},\mathbf{z})\mathbf{u} \tag{9}$$

The reduced slow subsystem is a locally flattened space of the complete system (Eqs.1,2). Since the vector fields were assumed to be sufficiently smooth there exists a smooth diffeomorphism that maps the complete system onto this locally flattened space. The set of points $(\mathbf{x},\mathbf{z},\mathbf{u}) \in \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^p$ is a smooth manifold $\mathscr{M}_0$ of dimension $m+p$ that satisfies the algebraic Eq.7:

$$\mathscr{M}_0 : \mathbf{z} = \mathbf{Z}_0(\mathbf{x},\mathbf{u}) \tag{10}$$

This set of points is identically the fixed points of the reduced fast subsystem (Eq.9). Thus the manifold $\mathscr{M}_0$ is invariant. The flow on this manifold is described by the differential equation

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x},\mathbf{Z}_0(\mathbf{x},\mathbf{u})\right) + \mathbf{g}\left(\mathbf{x},\mathbf{Z}_0(\mathbf{x},\mathbf{u})\right)\mathbf{u} \tag{11}$$

Fenichel[6] proved that the dynamics of a singularly perturbed system of the form represented in Eqs.1,2 is constrained within $O(\varepsilon)$ of Eq.11 if the reduced fast subsystem is stable about $\mathscr{M}_0$. If the dynamics of Eq.11 are locally asymptotically stable about the manifold, then it can be concluded that the complete system is also locally asymptotically stable. Global asymptotic stability conclusions about the complete system can only be made if the manifold is unique, which is a result from differential topology and center manifold theory [6].

3 Control Formulation and Stability Analysis

The central idea in the development is the following. If the manifold is unique and an asymptotically stable fixed point of the reduced fast subsystem, the complete system follows the dynamics of the reduced slow subsystem globally. Therefore, for a tracking problem addressed in this paper it is desired that this manifold lie exactly on the desired fast state reference for all time. *This condition can be enforced if the nonlinear algebraic set of equations is augmented with a controller that enforces the reference to be the unique manifold*. Additionally, this controller should also be capable of simultaneously driving the slow states to their specified reference. These ideas are mathematically formulated and analyzed in the following subsections.

3.1 Control Law Development

The objective is to augment the two time-scale system with controllers such that the system follows smooth, bounded, time-varying trajectories $[\mathbf{x}_r(t), \mathbf{z}_r(t)]^T$. The first step is to transform the problem into a non-autonomous stabilization control problem. Define the tracking error signals as

$$\mathbf{e}(t) = \mathbf{x}(t) - \mathbf{x}_r(t) \tag{12}$$

$$\xi(t) = \mathbf{z}(t) - \mathbf{z}_r(t) \tag{13}$$

Substituting Eqs.1,2, the tracking error dynamics are expressed as

$$\dot{\mathbf{e}} = \mathbf{f}(\mathbf{x},\mathbf{z}) + \mathbf{g}(\mathbf{x},\mathbf{z})\mathbf{u} - \dot{\mathbf{x}}_r \triangleq \mathbf{F}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\dot{\mathbf{x}}_r) + \mathbf{G}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r)\mathbf{u} \tag{14}$$

$$\varepsilon\dot{\xi} = \mathbf{l}(\mathbf{x},\mathbf{z}) + \mathbf{k}(\mathbf{x},\mathbf{z})\mathbf{u} - \varepsilon\dot{\mathbf{z}}_r \triangleq \mathbf{L}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\varepsilon\dot{\mathbf{z}}_r) + \mathbf{K}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r)\mathbf{u} \tag{15}$$

The control law is formulated using the reduced-order models for the complete stabilization problem, which is obtained using the procedure developed in Section 2.
Reduced Slow Subsystem

$$\dot{\mathbf{e}} = \mathbf{F}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\dot{\mathbf{x}}_r) + \mathbf{G}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r)\mathbf{u}_0 \tag{16}$$

$$\mathbf{0} = \mathbf{L}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\mathbf{0}) + \mathbf{K}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r)\mathbf{u}_0 \tag{17}$$

Reduced Fast Subsystem

$$\mathbf{e}' = \mathbf{0} \tag{18}$$

$$\xi' = \mathbf{L}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\mathbf{z}_r') + \mathbf{K}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r)(\mathbf{u}_0 + \mathbf{u}_f) \tag{19}$$

It is known that the fast tracking error ξ will settle onto the manifold that is a function of the error $\mathbf{e}$ and control input $\mathbf{u}$, which may not necessarily be the origin. To steer both errors to the origin, the control input must be designed such that the origin becomes the unique manifold of the reduced slow system (Eqs.16,17). There-

fore, the slow controller $\mathbf{u}_0$ is designed to take the form

$$\begin{bmatrix} \mathbf{G}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r) \\ \mathbf{K}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r) \end{bmatrix} \mathbf{u}_0 = -\begin{bmatrix} \mathbf{F}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\dot{\mathbf{x}}_r) \\ \mathbf{L}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\mathbf{0}) \end{bmatrix} + \begin{bmatrix} A_{\mathbf{e}}\mathbf{e} \\ A_\xi \xi \end{bmatrix} \tag{20}$$

where $A_{\mathbf{e}}$ and A_ξ specify the desired closed-loop characteristics. With this choice of slow control, the reduced fast subsystem becomes

$$\mathbf{e}' = \mathbf{0} \tag{21}$$
$$\xi' = \mathbf{L}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\mathbf{z}_r') - \mathbf{L}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\mathbf{0}) + A_\xi \xi + \mathbf{K}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r)\mathbf{u}_f \tag{22}$$

To stabilize the fast subsystem, the fast control $\mathbf{u}_f$ is designed as

$$\begin{bmatrix} \mathbf{G}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r) \\ \mathbf{K}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r) \end{bmatrix} \mathbf{u}_f = \begin{bmatrix} \mathbf{0} \\ \mathbf{L}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\mathbf{0}) - \mathbf{L}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\mathbf{z}_r') \end{bmatrix} \tag{23}$$

Thus, the composite control $\mathbf{u} = \mathbf{u}_0 + \mathbf{u}_f$ satisfies

$$\begin{bmatrix} \mathbf{G}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r) \\ \mathbf{K}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r) \end{bmatrix} \mathbf{u} = -\begin{bmatrix} \mathbf{F}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\dot{\mathbf{x}}_r) \\ \mathbf{L}(\mathbf{e},\xi,\mathbf{x}_r,\mathbf{z}_r,\mathbf{z}_r') \end{bmatrix} + \begin{bmatrix} A_e\mathbf{e} \\ A_\xi \xi \end{bmatrix} \tag{24}$$

assuming that the rank of $\begin{bmatrix} \mathbf{G}(.) \\ \mathbf{K}(.) \end{bmatrix} \geq (m+n)$.

The complete closed-loop and reduced slow subsystem for this control law are given as

$$\dot{\mathbf{e}} = A_e\mathbf{e} \tag{25}$$
$$\varepsilon\dot{\xi} = A_\xi \xi. \tag{26}$$

and

$$\dot{\mathbf{e}} = A_e\mathbf{e} \tag{27}$$
$$\mathbf{0} = A_\xi \xi. \tag{28}$$

respectively. Observe that with the proposed control law the nonlinear algebraic set of equations (Eq.17) have been transformed to a linear set of equations (Eq.28). With the proper choice of A_ξ, it is guaranteed that $\xi = \mathbf{0}$ is the unique manifold for both the complete and the reduced slow subsystems. Furthermore, this manifold is exponentially stable as can be deduced from the reduced fast subsystem

$$\mathbf{e}' = \mathbf{0} \tag{29}$$
$$\xi' = A_\xi \xi \tag{30}$$

Remark 1 The control law proposed in Eq.24 is independent of the perturbation parameter ε. Furthermore it is a function of $\mathbf{z}_r'$ that implies that the reference trajectory chosen for the fast states must be faster when compared to the reference of the slow states. Additionally, as for all singular perturbation techniques to work the closed-loop eigenvalues A_e and A_ξ must be chosen so as to maintain the time-scale separation.

3.2 Stability Analysis

Complete system stability is analyzed using the composite Lyapunov function approach[10]. Suppose that there exist positive definite Lyapunov functions $V(t,\mathbf{e}) = \mathbf{e}^T\mathbf{e}$ and $W(t,\xi) = \xi^T\xi$ for the reduced subsystems, with continuous first-order derivatives satisfying the following properties:

1. $V(t,\mathbf{0}) = 0$ and $\gamma_1||\mathbf{e}||^2 \leq V(t,\mathbf{e}) \leq \gamma_2||\mathbf{e}||^2\ \forall t \in \mathbb{R}^+, \mathbf{e} \in \mathbb{R}^m, \gamma_1 = \gamma_2 = 1$,
2. $(\nabla_\mathbf{e} V(t,\mathbf{e}))^T A_\mathbf{e}\mathbf{e} \leq -\alpha_1\mathbf{e}^T\mathbf{e}, \quad \alpha_1 = 2|\lambda_{\min}(A_\mathbf{e})|$,
3. $W(t,\mathbf{0}) = 0$ and $\gamma_3||\xi||^2 \leq W(t,\xi) \leq \gamma_4||\xi||^2\ \forall t \in \mathbb{R}^+, \xi \in \mathbb{R}^n, \gamma_3 = \gamma_4 = 1$,
4. $(\nabla_\xi W(t,\xi))^T A_\xi\xi \leq -\alpha_2\xi^T\xi, \quad \alpha_2 = 2|\lambda_{\min}(A_\xi)|$.

Next, consider the composite Lyapunov function $v(t,\mathbf{e},\xi) : \mathbb{R}^+ \times \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^+$ defined by the weighted sum of $V(t,\mathbf{e})$ and $W(t,\xi)$ for the complete closed-loop system

$$v(t,\mathbf{e},\xi) = (1-d)V(t,\mathbf{e}) + dW(t,\xi), \quad 0 < d < 1 \tag{31}$$

The derivative of $v(t,\mathbf{e},\xi)$ along the closed-loop trajectories Eqs.25,26 is given by

$$\dot{v} = (1-d)(\nabla_\mathbf{e} V)^T\dot{\mathbf{e}} + d(\nabla_\xi W)^T\dot{\xi} \tag{32}$$

$$\dot{v} = (1-d)(\nabla_\mathbf{e} V)^T A_\mathbf{e}\mathbf{e} + \frac{d}{\varepsilon}(\nabla_\xi W)^T A_\xi\xi \tag{33}$$

$$\dot{v} \leq -(1-d)\alpha_1\mathbf{e}^T\mathbf{e} - \frac{d}{\varepsilon}\alpha_2\xi^T\xi \tag{34}$$

$$\dot{v} \leq -\begin{bmatrix}\mathbf{e}\\ \xi\end{bmatrix}^T\begin{bmatrix}(1-d)\alpha_1 & 0\\ 0 & \frac{d}{\varepsilon}\alpha_2\end{bmatrix}\begin{bmatrix}\mathbf{e}\\ \xi\end{bmatrix} \tag{35}$$

Following the approach proposed in Reference[3], add and subtract $2\alpha v(t,\mathbf{e},\xi)$ to Eq.35 to get

$$\dot{v} \leq -\begin{bmatrix}\mathbf{e}\\ \xi\end{bmatrix}^T\begin{bmatrix}(1-d)\alpha_1 & 0\\ 0 & \frac{d}{\varepsilon}\alpha_2\end{bmatrix}\begin{bmatrix}\mathbf{e}\\ \xi\end{bmatrix} + 2\alpha(1-d)V + 2\alpha dW - 2\alpha v \tag{36}$$

where $\alpha > 0$. Substitute in Eq.36 for the Lyapunov functions $V(t,\mathbf{e})$ and $W(t,\xi)$ to get

$$\dot{v} \leq -\begin{bmatrix} \mathbf{e} \\ \xi \end{bmatrix}^T \begin{bmatrix} (1-d)\alpha_1 - 2\alpha(1-d) & 0 \\ 0 & \frac{d}{\varepsilon}\alpha_2 - 2\alpha d \end{bmatrix} \begin{bmatrix} \mathbf{e} \\ \xi \end{bmatrix} - 2\alpha v \tag{37}$$

If ε satisfies

$$\varepsilon < \varepsilon^* = \frac{\alpha_2}{2\alpha} \tag{38}$$

and provided $\alpha_1 > 2\alpha$, then from the definitions of α_2, α, and d it can be concluded that the matrix in Eq.37 is positive definite. Then the derivative of the Lyapunov function is lower-bounded by

$$\dot{v} \leq -2\alpha v \tag{39}$$

Since the composite Lyapunov function lies within the following bounds

$$(1-d)\gamma_1||\mathbf{e}||^2 + d\gamma_3||\xi||^2 \leq v(t,\mathbf{e},\xi) \leq (1-d)\gamma_2||\mathbf{e}||^2 + d\gamma_4||\xi||^2 \tag{40}$$

or,

$$\gamma_{11}\left|\left|\begin{bmatrix} \mathbf{e} \\ \xi \end{bmatrix}\right|\right|^2 \leq v(t,\mathbf{e},\xi) \leq \gamma_{22}\left|\left|\begin{bmatrix} \mathbf{e} \\ \xi \end{bmatrix}\right|\right|^2 \tag{41}$$

where $\gamma_{11} = \min((1-d)\gamma_1, d\gamma_3)$ and $\gamma_{22} = \min((1-d)\gamma_2, d\gamma_4)$, the derivative of the Lyapunov function can be expressed as

$$\dot{v} \leq -2\alpha\gamma_{11}\left|\left|\begin{bmatrix} \mathbf{e} \\ \xi \end{bmatrix}\right|\right|^2 \tag{42}$$

From the definition of the constants γ_{11}, γ_{22}, and α, and invoking Lyapunov's Direct Method[9], *uniform exponential stability in the large of* $(\mathbf{e} = \mathbf{0}, \xi = \mathbf{0})$ *can be concluded*. Furthermore, since the reference trajectory $\mathbf{x}_r(t)$ and $\mathbf{z}_r(t)$ is bounded, it is concluded that the states $\mathbf{x}(t) \to \mathbf{x}_r(t)$ and $\mathbf{z}(t) \to \mathbf{z}_r(t)$ as $t \to \infty$. Since the matrix $\begin{bmatrix} \mathbf{G}(.) \\ \mathbf{K}(.) \end{bmatrix}$ is restricted to be full rank, examining the expression for $\mathbf{u}$ in Eq.24 it is concluded that $\mathbf{u} \in \mathcal{L}_\infty$.

Remark 2 Recall that for the special case of state regulation the system dynamics in Eqs.14,15 become autonomous. In such a case, the result of global exponential stability is obtained with less-restrictive conditions on the Lyapunov functions $V(\mathbf{e})$, $W(\xi)$, and consequently $v(\mathbf{e},\xi)$. Similar conclusions were made in Reference[3] for the stabilization problem of a special class of singularly perturbed systems where the control affects only the fast states. Note that for the special class of systems considered in Reference[3], the non-diagonal elements of the matrix in Eq.37 are nonzero, and the bound on the parameter ε is slightly different.

Remark 3 From Eq.37, a conservative upper bound for α is $\alpha < \frac{\alpha_1}{2}$, and consequently $\varepsilon^* \approx \frac{\alpha_2}{\alpha_1}$. Therefore, qualitatively this upper bound is indirectly dependent upon the choice of the closed-loop eigenvalues.

4 Numerical Simulation

The purpose of the example is to demonstrate the methodology and controller performance for an under-actuated, nonlinear, singularly perturbed system. The system studied is a nonlinear, coupled, six degree-of-freedom F/A-18A Hornet aircraft[5]. The combined longitudinal-lateral/directional maneuver requires tracking of the fast variables, in this case body-axis pitch and roll rates, while maintaining zero sideslip angle. Closed-loop characteristics such as stability, accuracy, speed of response and robustness are qualitatively analyzed. The maneuver consists of an aggressive vertical climb with a pitch rate of 25 deg/sec, followed by a roll at a rate of 50 deg/sec, while maintaining zero sideslip angle. The Mach number and angle-of-attack are assumed to be input-to-state stable. The initial conditions are a Mach number of 0.4 at $15,000$ feet, an angle-of-attack of 10 deg, and elevon angle of -11.85 deg. All other states are zero. The F/A-18A Hornet model is expressed in stability axes. Since it is difficult to cast the nonlinear aircraft model into the singular perturbation form of Eq.1-2 , the perturbation parameter ε is introduced in front of those state variables that have the fastest dynamics. This is done so that the results obtained for $\varepsilon = 0$ will closely approximate the complete system behaviour (with $\varepsilon = 1$). This is called the forced perturbation technique, and is commonly used in the aircraft literature [2, 12]. Motivated by experience and previous results, the six slow states are Mach number M, angle-of-attack α, sideslip angle β and the three kinematic states: bank angle ϕ, pitch-attitude angle θ, and heading angle ψ. The three body-axis angular rates (p,q,r) constitute the fast states. The control variables for this model are elevon δ_e, aileron δ_a, and rudder δ_r and are assumed to have sufficiently fast enough actuator dynamics. The convention used is that a positive deflection generates a negative moment. The throttle η is maintained constant at 80%, because slow engine dynamics require introduction of an additional time-scale in the analysis; this is a consideration which is beyond the scope of this paper. The aerodynamic stability and control derivatives are represented as nonlinear analytical functions of aerodynamic angles and control surface deflections. Quaternions are used to represent the kinematic relationships from which the Euler angles are extracted. The details of these relationships are discussed in Reference[15].

Results and Discussion

Simulation results in Figures 1-6 show that all controlled states closely track their references. At two seconds the aircraft is commanded to perform a vertical climb, and after eight seconds the pitch rate command changes direction and Mach num-

ber drops. The lateral/directional states and controls are identically zero until the roll command is introduced at time equals 15 seconds. Observe that all of the states asymptotically track the reference. Figure 2 shows that the elevon deflection remains within specified limits[5] throughout the vertical climb, and the commanded roll produces a sideslip angle which is negated by application of the rudder. The aileron and the rudder deflections remain within bounds while the aircraft rolls and comes back to level flight. The maximum pitch-attitude angle is 81 deg, maximum bank angle is 81 deg (Figure 4), and the maximum sideslip error is $\pm$4deg. The quaternions and the complete trajectory are shown in Figures 5 and 6 respectively. From Figure 6, note that after completing the combined climb and roll maneuver, the aircraft is commanded to remain at zero sideslip angle, roll rate, and pitch rate. It then enters a steady dive with all other aircraft states bounded. The controller response is judged to be essentially independent of the reference trajectory designed. The robustness properties of the controller are quantified by the upper bound ε^*. For this example, the design variables are $d = 0.5$, $\alpha_1 = 10$, $\alpha = 2$, and $\alpha_2 = 15$, so the upper bound becomes $\varepsilon^* = 7.5$. Therefore for all $\varepsilon < \varepsilon^*$ global asymptotic tracking is guaranteed and in this case $\varepsilon = 1$.

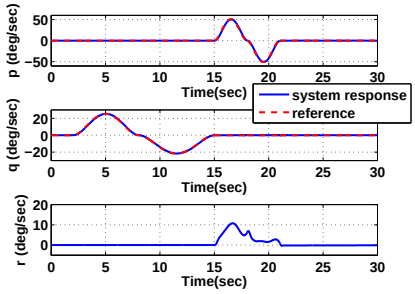


Fig. 1 Body-Axis Angular Rates

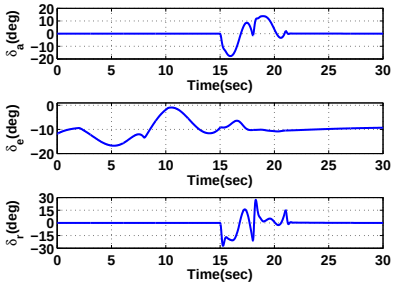


Fig. 2 Control Surface Deflections

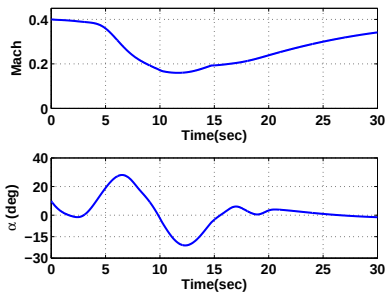


Fig. 3 Mach Number and Angle-of-Attack

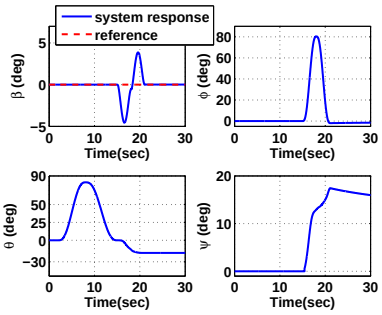


Fig. 4 Sideslip Angle and Kinematic Angles

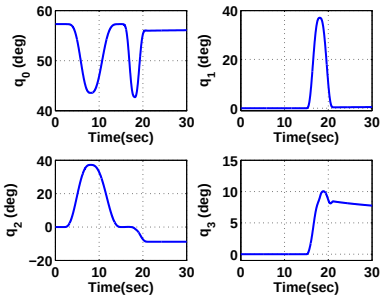


Fig. 5 Quaternion Parameters

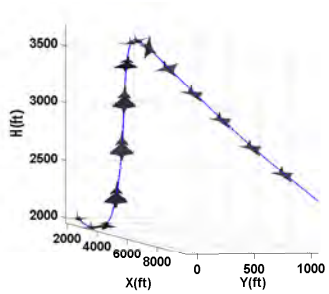


Fig. 6 Three-Dimensional Trajectory

5 Conclusions

A control law for global asymptotic tracking of both the slow and the fast states for a general class of nonlinear singularly perturbed systems was developed. A composite control approach was adopted to satisfy two objectives. First, it enforces the specified reference for the fast states to be 'the unique manifold' of the fast dynamics for all time. Second, it ensures that the slow states are tracked simultaneously as desired. Stability of the closed-loop signals was analyzed using the composite Lyapunov approach, and controller performance was demonstrated through numerical simulation of a nonlinear, coupled, six degree-of-freedom model of an F/A-18A Hornet. The control laws were implemented without making any assumptions about the nonlinearity of the six degree-of-freedom aircraft model. Based on the results presented in the paper, the following conclusions are drawn. First, both positive and negative angular rate commands were seen to be perfectly tracked by the controller and consistent tracking was guaranteed independent of the desired reference trajectory. Second, throughout the maneuver the controller demonstrated global asymptotic tracking even though the desired reference trajectory requires the aircraft to switch between linear and nonlinear regimes. This robust performance of the controller was shown to hold for all $\varepsilon < \varepsilon^* = 7.5$. Third, all closed-loop signals were bounded and the control surface deflections computed were smooth and within specified limits. Fourth, this technique does not require the knowledge of the perturbation parameter ε. This is an important consideration for systems such as aircraft, where quantifying this parameter can be difficult.

Acknowledgements This material is based upon work supported in part by the U.S. Air Force Office of Scientific Research under contract FA9550-08-1-0038, with technical monitor Dr.Fariba Fahroo. This support is gratefully acknowledged by the authors. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Air Force.

References

[1] Arstein Z, Gaitsgory V (1997) Tracking fast trajectories along a slow dynamics: a singular perturbation approach. SIAM Journal of Control and Optimization 35(4):1487–1507
[2] Calise AJ (1976) Singular perturbation methods for variational problems in aircraft flight. IEEE Transactions on Automatic Control 21:345–353
[3] Chen CC (1998) Global exponential stabilization for nonlinear singularly perturbed systems. IEEE Proceedings of Control Theory and Applications 145:377–382
[4] Choi HL, Lim JT (2005) Gain scheduling control of nonlinear singularly perturbed time-varying systems with derivative information. International Journal of Systems Science 36(6):357–364
[5] Fan Y, Lutze FH, MCliff E (1995) Time-optimal lateral maneuvers of an aircraft. Journal of Guidance,Control and Dynamics 18:1106–1112
[6] Fenichel N (1979) Geometric singular perturbation theory for ordinary differential equations. Journal of Differential Equations 31:53–98
[7] Georgie J, Valasek J (2003) Evaluation of longitudinal desired dynamics for dynamic-inversion controlled generic reentry vehicles. Journal of Guidance, Control and Dynamics 26:811–819
[8] Grujic LT (1988) On the theory and synthesis of nonlinear non-stationary tracking singularly perturbed systems. Control Theory and Advanced Technology 4(4):395–409
[9] Ioannou P, Sun J (2003) Robust Adaptive Control. Prentice Hall Inc.
[10] Kokotovic P, Khalil HK, Reilly JO (1986) Singular Perturbation Methods in Control: Analysis and Design. Academic Press
[11] Li L, Sun FC (2009) An adaptive tracking controller design for nonlinear singularly perturbed systems using fuzzy singularly perturbed model. IMA Journal of Mathematical Control and Information 26:395–415
[12] Menon P, Badgett ME, Walker R (1987) Nonlinear flight test trajectory controllers for aircraft. Journal of Guidance 10:67–72
[13] Naidu DS (1988) Singular Perturbation Methodology in Control Systems, vol 34. IEEE Control Engineering Series
[14] Naidu DS, JCalise A (2001) Singular perturbations and time scales in guidance and control of aerospace systems: A survey. Journal of Guidance, Control and Dynamics 24(6):1057–1078
[15] Schaub H, Junkins JL (2003) Analytical Mechanics of Space Systems. AIAA Education Series
[16] Siddarth A, Valasek J (2011) Kinetic state tracking of a class of singularly perturbed systems. AIAA Journal of Guidance, Navigation and Control Accepted, To appear

AIAA Guidance, Navigation, and Control Conference
08 - 11 August 2011, Portland, Oregon

AIAA 2011-6487

Output Tracking of Non-Minimum Phase Dynamics

Anshu Siddarth* and John Valasek†
Texas A & M University, College Station, TX 77843-3141.

This paper develops a general control algorithm for the exact output tracking of nonlinear systems with non-minimum phase dynamics. The control technique is causal and does not require preview or knowledge of the desired reference beforehand. Additionally, the control is independent of the operating condition and the desired reference. The main idea of the paper is to convert the output tracking problem into a slow state tracking problem for singularly perturbed systems. Previous work on singularly perturbed systems have shown asymptotic tracking of slow states only for a class of nonlinear systems that are linear in the fast states. However, this paper develops a control technique that does not have this restriction and is applicable to a general class of nonlinear singularly perturbed systems. The procedure is to compute the desired internal state trajectory and the control scheme that stabilizes the nonlinear system online, thereby guaranteeing asymptotic output tracking. Performance of this approach is demonstrated in simulation for two benchmark problems: the beam-ball example that is slightly non-minimum phase and fails to have a well-defined relative degree, and the Conventional Take-off and Landing (CTOL) non-minimum phase aircraft. Results presented in the paper show that the approach is able to accomplish perfect tracking while stabilizing the closed-loop system, while keeping all closed-loop signals bounded.

Nomenclature

n	order of the nonlinear system
p	number of control variables and number of outputs
r	relative-degree of the output
t	slow time
$\mathbf{u}$	control vector
$\mathbf{x}$	state vector
$\mathbf{y}$	output vector

Greek

(ξ, η)	normal coordinates of the system
ϵ	singular perturbation parameter
$\nu(t)$	desired output dynamics
τ	fast time

Subscripts

d	reference trajectory
0	reference quantity

Symbols

$\dot{}$	time derivative with respect to slow time
$'$	time derivate with respect to fast time
$O()$	order symbol

*Graduate Research Assistant, Vehicle Systems & Control Laboratory, Department of Aerospace Engineering, Student Member AIAA, *anshun1@tamu.edu*

†Professor and Director, Vehicle Systems & Control Laboratory, Department of Aerospace Engineering, Associate Fellow AIAA, *valasek@tamu.edu, http://jungfrau.tamu.edu/valasek* (Corresponding Author)

I. Introduction

OUTPUT tracking control structures have received considerable attention in the literature. Nonlinear control techniques such as Feedback Linearization and Sliding Mode Control are able to guarantee closed-loop stability and precise output tracking, but only for a specific class of nonlinear systems that are minimum phase.[1] Additionally, these control approaches require that the output have a well-defined relative degree. But there are a number of important flight control problems such as acceleration control of tail-controlled missiles,[2] control of planar Vertical Take-off and Landing (V/STOL) aircraft,[3] and Conventional Take-off and Landing (CTOL) aircraft[4] that are characterized by unstable zero dynamics, thereby not satisfying the conditions cited above. These restrictions and the need to develop stabilizing trackers have paved the way for control algorithms that are applicable to a more general class of nonlinear systems.

The technique presented by Benvenuti et.al[5] modified the output of a corresponding linear system so that it does not contain right half-plane zeros. A similar technique was employed by Hedrick and Gopalswamy[6] to track pilot g commands while satisfying flying quality specifications. These approaches were able to guarantee 'local' tracking that is specific to the desired flight condition and reference trajectory. Another approximate approach proposed by Doyle et.al[7] takes a sufficient number of derivatives of the output such that the control and its higher-order derivatives appear in the equation. The paper proposed to modify the sign of some of the control derivatives in order to render the modified output dynamics minimum phase. It was shown that these modified output dynamics closely approximate the actual dynamics of the system. In contrast to the former, Shklnikov and Shtessel[8] modified the sliding surface to ensure that the right half-plane zero is canceled out. The system was required to be in normal form with bounded nonlinearities, and the technique was demonstrated for an F-16 aircraft.[9] Considering the local nature of these works, Zhu et.al[10] proposed a controller which separates the internal dynamics into linear and nonlinear parts. The linear part is stabilized by linear state feedback, whereas the nonlinear part is stabilized only if the system strays away from the trajectory. In an effort to control the V/STOL slightly non-minimum phase aircraft, Hauser et.al[3] neglected some terms that are the cause of this unstable behaviour, and proved that a stable controller can be designed using this approximate technique.

Another class of the literature takes advantage of the multiple time-scale behaviour of air vehicles. Lee and Ha[2] designed an autopilot for a Skid-To-Turn (STT) missile by splitting the dynamics into slow and fast components. The slow subsystem was composed of the zero dynamics and was indirectly controlled by the controllable fast subsystem. A similar approach was proposed by Lee and Ha[11] wherein the normal form of a nonlinear I/O feedback linearizable system was transformed to a two time-scale system by a change of coordinates. But in this case the fast subsystem constituted the zero dynamics, and a modified composite control scheme was employed to stabilize the complete system.

In addition to the approximate schemes described above, low gain feedback approaches have been proposed in the literature for nonlinear systems with the upper triangular form.[12,13,14] The exact output tracking approach proposed by Devasia et.al[15] employed a combination of feed-forward and feedback control. The feed-forward control was found using inversion, given a desired output trajectory and its higher-order derivatives. This inversion is non-causal and requires the infinite time preview of the complete output trajectory. It is computed offline, and the inversion computes the desired input-state trajectory that would lead to asymptotic output tracking. The linear feedback control is employed to locally stabilize the internal dynamics. This approach was extended to require a finite time preview of the output and was applied to the benchmark VTOL landing example.[16]

Summarizing these previous results, internal-state feedback is necessary to stabilize a non-minimum phase system. Moreover, exact output tracking is achieved when the desired internal state trajectory is tracked. Motivated by this fact, this paper develops an exact output tracking control technique for non-minimum systems using singular perturbation methods. The paper makes three major contributions. First, the output dynamics are not required to have a well-defined relative degree with respect to the input. The idea is to take a sufficient number of derivatives of the output and cast the system in a singularly perturbed form. This procedure forces the internal states of the system to behave as the fast variables. It also allows the internal states to be used as 'pseudo-control variables' for output tracking. A sequential procedure is developed to compute the internal states that ensure asymptotic output tracking and the controller is designed to force the internal states to follow the computed trajectory. The second contribution is a full-state feedback controller that is designed online, and is independent of any particular operating condition and desired output trajectory. Third, the controller so designed is causal and does not require any knowledge or preview of the output trajectory beforehand.

The paper is organized as follows. Section II describes the class of systems considered and formulates the control problem. Section III develops the nonlinear control design and analyses stability of the closed-loop system. In Section IV the methodology is illustrated with application to two benchmark problems: the beam-ball example that is slightly non-minimum phase and does not have a well-defined relative degree, and the classic CTOL aircraft non-minimum phase problem. Conclusions are presented in Section V.

II. Problem Statement

The dynamical system considered is the nonlinear affine in control dynamical system expressed as

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(\mathbf{t}))\mathbf{u}(t) && (1) \\ \mathbf{y}(t) &= \mathbf{h}(\mathbf{x}(t)) && (2) \end{aligned}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the vector of state variables, $\mathbf{y}(t) \in \mathbb{R}^p$ is the output vector, and $\mathbf{u}(t) \in \mathbb{R}^p$ is the vector of control variables. The vector fields $\mathbf{f}(.)$, $\mathbf{g}(.)$, and $\mathbf{h}(.)$ are sufficiently smooth. The control objective is to ensure that the output asymptotically tracks a sufficiently smooth, time-varying, bounded trajectory, such that $\mathbf{y}(t) \to \mathbf{y}_d(t)$ as $t \to \infty$. It is assumed that the nonlinear system considered satisfies the following assumptions:

Assumption 1: The system described by Eqs.1-2 is non-minimum phase.

Assumption 2: The output of the nonlinear dynamical system considered does not have a well-defined relative degree with respect to the control variables.

Assumption 3: The output dynamics are differentially flat or the number of control inputs available is equal to the number of output variables to be controlled.

Assumption 4: The desired output trajectory y_d and its higher-order derivatives are bounded.

Henceforth the time-dependency notation is dropped for convenience.

III. Tracking Control Development

The system dynamics Eqs.1-2 are expanded and written in the form

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u} && (3a) \\ y_1 &= h_1(\mathbf{x}) \\ y_2 &= h_2(\mathbf{x}) \\ &\vdots \\ y_p &= h_p(\mathbf{x}) && (3b) \end{aligned}$$

Let the relative degree of the outputs $(y_1, y_2, \ldots, y_p)$ be $(r_1, r_2, \ldots, r_p)$ respectively, and let $r = r_1 + r_2 + \ldots + r_p$. Note that in this context the relative-degree is defined as the number of derivatives of the output required such that the control appears linearly. The control influence may be singular. The system is cast into normal form using the procedure shown in Reference 1. Define $\xi_j^i = y_i^{j-1}$ for $i = 1, \ldots, p$ and $j = 1, \ldots, r_i$ and denote

$$\begin{aligned} \xi(t) &= (y_1, y_1^{(1)}, \ldots, y_1^{r_1}, y_2, \ldots, y_2^{r_2}, \ldots, y_p^{r_p})^T \\ &= (\xi_1^1, \xi_2^1, \ldots, \xi_{r_1}^1, \xi_1^2, \ldots, \xi_{r_2}^2, \ldots, \xi_{r_p}^p)^T && (4) \end{aligned}$$

Let $\eta \in \mathbb{R}^{n-r}$ denote the set of internal states such that (ξ, η) form a new set of coordinates for the n^{th} order system Eqs.3. In these new coordinates the system dynamics become

$$\begin{aligned} \dot{\xi}_1^i &= \xi_2^i \\ &\vdots \\ \dot{\xi}_{r_i-1}^i &= \xi_{r_i}^i \qquad (5a) \\ \dot{\xi}_{r_i}^i &= f_{y_i}(\xi,\eta) + g_{y_i}(\xi,\eta)\mathbf{u} \\ \dot{\eta} &= \mathbf{f}_\eta(\xi,\eta) + \mathbf{g}_\eta(\xi,\eta)\mathbf{u} \qquad (5b) \end{aligned}$$

for $i = 1, \ldots, p$. It has been shown in Reference.11 that the normal form of Eqs.5 can be cast in the singularly perturbed form. This procedure shows that the internal dynamics constitute the fast subsystem. The singular perturbation parameter ϵ is introduced in the system of Eqs. 5 to emphasize that the internal states evolve faster than the other states.

$$\begin{aligned} \dot{\xi}_1^i &= \xi_2^i \\ &\vdots \\ \dot{\xi}_{r_i-1}^i &= \xi_{r_i}^i \qquad (6a) \\ \dot{\xi}_{r_i}^i &= f_{y_i}(\xi,\eta) + g_{y_i}(\xi,\eta)\mathbf{u} \\ \epsilon\dot{\eta} &= \mathbf{f}_\eta(\xi,\eta) + \mathbf{g}_\eta(\xi,\eta)\mathbf{u} \qquad (6b) \end{aligned}$$

It is desired that the slow states $\xi(t)$ follow the desired output trajectory while the fast states η remain bounded for all time. This problem has been studied in the literature as a control problem of asymptotic tracking of the slow states for a special class of systems in which the fast dynamics are linear in the fast states.[17] In the present work a control algorithm that leads to global asymptotic tracking is developed and demonstrated.

A. Control Design

In geometric singular perturbation theory,[18] the behaviour of singularly perturbed systems is determined using geometric constructs of the reduced-order models, which are obtained by substituting $\epsilon = 0$ in Eqs.6. This results in two subsystems

Reduced Slow Subsystem:

$$\begin{aligned} \dot{\xi}_1^i &= \xi_2^i \\ &\vdots \\ \dot{\xi}_{r_i-1}^i &= \xi_{r_i}^i \qquad (7a) \\ \dot{\xi}_{r_i}^i &= f_{y_i}(\xi,\eta) + g_{y_i}(\xi,\eta)\mathbf{u} \\ 0 &= \mathbf{f}_\eta(\xi,\eta) + \mathbf{g}_\eta(\xi,\eta)\mathbf{u} \qquad (7b) \end{aligned}$$

Reduced Fast Subsystem:

$$\begin{aligned} {\xi'}_1^i &= 0 \\ &\vdots \\ {\xi'}_{r_i-1}^i &= 0 \qquad (8a) \\ {\xi'}_{r_i}^i &= 0 \\ \eta' &= \mathbf{f}_\eta(\xi,\eta) + \mathbf{g}_\eta(\xi,\eta)\mathbf{u} \qquad (8b) \end{aligned}$$

where $'$ represents derivatives with respect to the fast time scale: $\tau = \frac{t}{\epsilon}$. The dynamics of the resulting reduced slow subsystem is restricted to r dimensions and constrained to lie upon an $\mathcal{M}_0$: $n-r$ dimensional

smooth surface defined by the nonlinear algebraic set of equations Eq.7b. This surface is identically the fixed points of the reduced fast subsystem Eq.8b. If the the reduced fast subsystem of Eqs.8 is stable about this smooth surface, then conclusions about the stability of the complete system Eqs.6 can be made by studying the reduced slow system Eqs.7.

Note that the smooth surface cannot be analytically computed, and in addition there maybe several such surfaces that satisfy the algebraic set of equations. In order to obtain the unique surface that the fast variables must be stable about it is assumed that the fast states are the pseudo control variables for the reduced slow subsystem. The control variables may then be computed by ensuring that the fast states follow the computed surface. For convenience rewrite the complete system Eqs.6 in compact form

$$\begin{aligned} \xi^r &= \mathbf{f_y}(\xi,\eta) + \mathbf{g_y}(\xi,\eta)\mathbf{u} && \text{(9a)} \\ \epsilon\dot{\eta} &= \mathbf{f}_\eta(\xi,\eta) + \mathbf{g}_\eta(\xi,\eta)\mathbf{u} && \text{(9b)} \end{aligned}$$

with the reduced slow subsystem written as

$$\begin{aligned} \xi^r &= \mathbf{f_y}(\xi,\eta) + \mathbf{g_y}(\xi,\eta)\mathbf{u} && \text{(10a)} \\ 0 &= \mathbf{f}_\eta(\xi,\eta) + \mathbf{g}_\eta(\xi,\eta)\mathbf{u} && \text{(10b)} \end{aligned}$$

Let $(\mathbf{y}_d, \mathbf{y}_d^1, \ldots, \mathbf{y}_d^r)^T$ denote the vector of the desired output trajectory and its r order derivatives. To ensure that $\mathbf{y}_d$ is an asymptotically stable equilibrium of the reduced slow system, define a positive-definite and decrescent Lyapunov function that satisfies the following:

Condition 1. $V(t, \xi - y_d) : [0,\infty) \times D_y \to \mathbb{R}$ is continuously differentiable and $D_y \in \mathbb{R}^r$ contains the origin, such that $0 < \psi_1(||\xi - y_d||) \leq V(t, \xi - y_d) \leq \psi_2(||\xi - y_d||)$ for some **class** $\mathcal{K}$ functions $\psi_1(.)$ and $\psi_2(.)$.

Design a manifold $\eta = \eta_{\mathbf{d}}(\xi, \mathbf{y_d}, \mathbf{u})$ such that the closed-loop reduced slow system Eq.10a satisfies

Condition 2.

$$\frac{\partial V}{\partial t} + \frac{\partial V}{\partial \xi}\left[\mathbf{f_y}(\xi,\eta_d) + \mathbf{g_y}(\xi,\eta_d)\mathbf{u}\right] \leq -\alpha_1\psi_3^2(\xi - \mathbf{y_d}), \quad \alpha_1 > 0$$

where $\psi_3(.)$ is a continuous positive-definite scalar function that satisfies $\psi_3(\mathbf{0}) = 0$.

Conditions 1-2 complete the design of the controller for the reduced slow subsystem. Note that the manifold $\mathcal{M}_0 : \eta_{\mathbf{d}}$ computed above is a function of the control $\mathbf{u}$, which is unknown. It is known that the complete system will have the properties of the reduced slow subsystem if the fast state asymptotically stabilizes about the fast state trajectory $\eta_{\mathbf{d}}$. This condition is enforced by designing the control $\mathbf{u}$. Define the error $\mathbf{e}_\eta = \eta - \eta_{\mathbf{d}}$ and rewrite Eq.9b as

$$e_\eta{}' = \mathbf{f}_\eta(\xi,\eta) + \mathbf{g}_\eta(\xi,\eta)\mathbf{u} \tag{11}$$

Define a positive-definite and decrescent Lyapunov function that satisfies

Condition 4. $W(t, \xi - \mathbf{y_d}, \mathbf{e}_\eta) : [0,\infty) \times D_{\mathbf{y}} \times D_\eta \to \mathbb{R}$ is continuously differentiable and $D_\eta \subset \mathbb{R}^{n-r}$ contains the origin, such that

$$0 < \phi_1(||\mathbf{e}_\eta||) \leq W(t, \xi - \mathbf{y_d}, \mathbf{e}_\eta) \leq \phi_2(||\mathbf{e}_\eta||)$$

for some **class** $\mathcal{K}$ functions $\phi_1(.)$ and $\phi_2(.)$.

and design $\mathbf{u}$ such that the closed-loop reduced fast system Eq. 11 satisfies

Condition 5.

$$\frac{\partial W}{\partial \mathbf{e}_\eta}\left[\mathbf{f}_\eta(\xi,\eta) + \mathbf{g}_\eta(\xi,\eta)\mathbf{u}\right] \leq -\alpha_3\phi_3^2(\mathbf{e}_\eta), \quad \alpha_3 > 0$$

where $\phi_3(.)$ is a continuous positive-definite scalar function that satisfies $\phi_3(\mathbf{0}) = 0$.

B. Stability Analysis

The following theorem summarizes the main result of the paper.

Theorem 1: Suppose the control $\mathbf{u}$ is designed according to the Conditions $1-6$ and the nonlinear system Eqs.1-2 satisfies Assumptions 1-4. Then for all initial conditions $(\xi - \mathbf{y_d}, \mathbf{e}_\eta) \in D_\mathbf{y} \times D_\eta$ the control uniformly asymptotically stabilizes the non-minimum phase system and equivalently drives the output $\mathbf{y}(t) \to \mathbf{y}_d(t)$ for all positive constants $\epsilon < \epsilon^*$, where ϵ^* satisfies the inequality Eq.17.

Proof. The closed-loop complete system in the error coordinates $(e = \xi - y_d)$ is given as

$$\begin{aligned} \mathbf{e^r} &= \mathbf{F}(\mathbf{e}, \mathbf{e}_\eta + \eta_d, \mathbf{y}_d, \mathbf{u}) && \text{(12a)} \\ \epsilon\dot{\mathbf{e}}_\eta &= \mathbf{G}(\mathbf{e}, \mathbf{e}_\eta + \eta_d, \mathbf{y}_d, \mathbf{u}) - \epsilon\dot{\boldsymbol{\eta}}_\mathbf{d} && \text{(12b)} \end{aligned}$$

Rearrange the closed-loop system to form

$$\begin{aligned} \mathbf{e^r} &= \mathbf{F}(\mathbf{e}, \eta_\mathbf{d}, \mathbf{y}_d, \mathbf{u}) && \text{(13a)} \\ &+ [\mathbf{F}(\mathbf{e}, \mathbf{e}_\eta + \eta_d, \mathbf{y}_d, \mathbf{u}) - \mathbf{F}(\mathbf{e}, \eta_\mathbf{d}, \mathbf{y}_d, \mathbf{u})] \\ \epsilon\dot{\mathbf{e}}_\eta &= \mathbf{G}(\mathbf{e}, \mathbf{e}_\eta + \eta_d, \mathbf{y}_d, \delta_\mathbf{u}) - \epsilon\dot{\boldsymbol{\eta}}_\mathbf{d} && \text{(13b)} \end{aligned}$$

Next, closed-loop system stability of the states is analyzed using the composite Lyapunov function approach.[19] Consider a Lyapunov function candidate for the complete closed-loop system

$$\nu(t, \mathbf{e}, \mathbf{e}_\eta) = V(t, \mathbf{e}) + W(t, \mathbf{e}, \mathbf{e}_\eta) \tag{14}$$

From the properties of V and W it follows that $\nu(t, \mathbf{e}, \mathbf{e}_\eta)$ is positive-definite and decrescent. The derivative of ν along the trajectories of Eq.13 is given by

$$\begin{aligned} \dot{\nu} &= \frac{\partial V}{\partial t} + \frac{\partial V}{\partial \mathbf{e}}\mathbf{e^r} \\ &+ \frac{\partial W}{\partial t} + \frac{\partial W}{\partial \mathbf{e}}\mathbf{e^r} + \frac{1}{\epsilon}\frac{\partial W}{\partial \mathbf{e}_\eta}\mathbf{e}_\eta{}' \end{aligned} \tag{15}$$

Suppose that Lyapunov functions V and W also satisfy both conditions

Condition 5.

$$\frac{\partial V}{\partial \mathbf{e}} [\mathbf{F}(\mathbf{e}, \mathbf{e}_\eta + \eta_d, \mathbf{y}_d, \mathbf{u}) - \mathbf{F}(\mathbf{e}, \eta_\mathbf{d}, \mathbf{y}_d, \mathbf{u})] \leq \beta_1 \psi_3(\mathbf{e}) \phi_3(\mathbf{e}_\eta); \beta_1 \geq 0$$

Condition 6.

$$\frac{\partial W}{\partial t} + \left[\frac{\partial W}{\partial \mathbf{e}} - \frac{\partial W}{\partial \mathbf{e}_\eta}\frac{\partial \eta_\mathbf{d}}{\partial \mathbf{e}}\right] \mathbf{e^r} \leq \gamma_1 \phi_3^2(\mathbf{e}_\eta) + \beta_2 \psi_3(\mathbf{e}) \phi_3(\mathbf{e}_\eta); \gamma_1 \geq 0, \beta_2 \geq 0$$

Using Conditions 1-6 Eq.15 now becomes

$$\dot{\nu} = -\alpha_1 \psi_3(\mathbf{e})^2 + \beta_1 \psi_3(\mathbf{e}) \phi_3(\mathbf{e}_\eta) - \frac{\alpha_2}{\epsilon} \phi_3(\mathbf{e}_\eta)^2 + \gamma_1 \phi_3^2(\mathbf{e}_\eta) + \beta_2 \psi_3(\mathbf{e}) \phi_3(\mathbf{e}_\eta) \tag{16}$$

Rearrange (16) to get

$$\dot{\nu} \leq -\Psi^T \mathbb{K} \Psi \tag{17}$$

where $\Psi = \begin{bmatrix} \psi_3 \\ \phi_3 \end{bmatrix}$ and

$$\mathbb{K} = \begin{bmatrix} \alpha_1 & -\frac{1}{2}[\beta_1 + \beta_2] \\ -\frac{1}{2}[\beta_1 + \beta_2] & \frac{\alpha_2}{\epsilon} - \gamma_1 \end{bmatrix}$$

and K is positive-definite for $\epsilon < \epsilon^*$. By definition of the continuous scalar functions ψ_3 and ϕ_3 it follows that $\dot{\nu}$ is negative definite. Using the composite Lyapunov approach[20] it is concluded that $\mathbf{y}(t) \to \mathbf{y}_d(t)$ asymptotically. Since the desired trajectory is assumed to be smooth and bounded with bounded first-order derivatives, the control commands $\mathbf{u}$ remain bounded for all time. □

IV. Numerical Examples

A. Purpose and Scope

The preceding theoretical developments are demonstrated with simulation for two benchmark problems. The first example is the beam-ball example which fails to have a well-defined relative degree. The objective is to ensure that the ball remains in contact with the beam and tracks any trajectory from a class of admissible trajectories. A step-by-step procedure of controller development is detailed for the system and the closed-loop results are studied for a time-varying trajectory. The second example develops control laws for a nonlinear CTOL aircraft problem. The objective of this example is to test the performance of the controller for a nonlinear, non-minimum phase benchmark problem.

B. Low-Order Nonlinear System Tracking: The Beam-Ball Example

The setup consists of a beam that can only rotate in a vertical plane by applying a torque at the center of the beam, and a ball that is free to roll along the beam. It is desired that the ball always remains in contact with the beam and that the rolling occurs without slipping. The goal is to track any trajectory from a class of admissible trajectories. The dynamical system is taken from Reference 3:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ B(x_1 x_4^2 - Gsinx_3) \\ x_4 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u \tag{18}$$

$$y = x_1 \tag{19}$$

where $\mathbf{x} = (x_1, x_2, x_3, x_4)^T \equiv (r, \dot{r}, \theta, \dot{\theta})$, $y = x_1$ where r is the distance of the ball from the center of the beam, and θ is the roll angle of the beam. The constants M and J_b are the mass and moment of inertia of the ball, J is the moment of inertia of the beam, R is the radius of the ball, G is the acceleration due to gravity, and B is defined as $M/(J_b/R^2 + M)$. The torque of the system is related to the control u by

$$\tau = (Mr^2 + J + J_b)u + 2Mr\dot{r}\dot{\theta} + MGr\cos\theta \tag{20}$$

The system output is required to track a desired trajectory $y_d(t)$ asymptotically.

1. *Control Design*

Following the procedure detailed in Section III the system Eq.18 is cast in the normal form

$$\begin{aligned} y &= x_1 && \text{(21a)} \\ \dot{y} &= x_2 && \text{(21b)} \\ \ddot{y} &= B(x_1 x_4^2 - Gsinx_3) && \text{(21c)} \\ \dddot{y} &= B(x_4^2 x_2 - Gx_4 cosx_3 + 2x_1 x_4 u) && \text{(21d)} \\ \dot{x}_4 &= u && \text{(21e)} \end{aligned}$$

Eqs.21 are clearly non-minimum phase since a feedback linearizable control cannot stabilize the internal state x_4. Additionally, if the angular velocity of the beam is zero and/or the ball is at the center of the beam, the control influence in Eq.21d is zero.

The system is nondimensionalized to determine whether or not it exhibits multiple time-scale behaviour. Let $(t_0, x_{10}, x_{20}, x_{30}, x_{40}, u_0)$ indicate the reference quantities for time and states of the system Eqs.18. Then the non-dimensional quantities can be represented as the ratio of the actual quantities over their respective

reference; for example $\hat{t} = \frac{t}{t_0}$, etc. Thus, the non-dimensionalized equations are given as

$$\dot{\hat{x}}_1 = \frac{x_{20}t_0}{x_{10}}\hat{x}_2 \tag{22a}$$

$$\dot{\hat{x}}_2 = \frac{t_0}{x_{20}}(Bx_{10}x_{40}^2\hat{x}_1\hat{x}_4^2 - BGsin(x_{30}\hat{x}_3)) \tag{22b}$$

$$\dot{\hat{x}}_3 = \frac{t_0 x_{40}}{x_{30}}\hat{x}_4 \tag{22c}$$

$$\dot{\hat{x}}_4 = \frac{t_0}{x_{40}}u_0\hat{u} \tag{22d}$$

Note that the reference quantities $(t_0, x_{10}, x_{20}, u_0)$ are all of $O(1)$, whereas (x_{30}, x_{40}) are of $O(0.1)$ or even less as they represent angular quantities in radians. So it can be seen that the evolution of the states $(\hat{x}_1, \hat{x}_2, \hat{x}_3)$ is of $O(1)$ whereas the evolution of the state x_4 is of $O(1/0.1)$ which is much faster. Therefore it is concluded that the state x_4 evolves at a much faster rate when compared to the other states of the system.

As a result of the above analysis, a desired internal state trajectory x_{4d} is computed such that the output asymptotically tracks the desired trajectory. The reduced slow system for this example is given by the following equations

$$y = x_1 \tag{23a}$$

$$\dot{y} = x_2 \tag{23b}$$

$$\ddot{y} = B(x_1x_4^2 - Gsinx_3) \tag{23c}$$

$$\dddot{y} = B(x_4^2x_2 - Gx_4cosx_3 + 2x_1x_4u) \tag{23d}$$

$$0 = u \tag{23e}$$

which simplifies to

$$y = x_1 \tag{24a}$$

$$\dot{y} = x_2 \tag{24b}$$

$$\ddot{y} = B(x_1x_{4d}^2 - Gsinx_3) \tag{24c}$$

$$\dddot{y} = B(x_{4d}^2x_2 - Gx_{4d}cosx_3) \tag{24d}$$

Let the desired output dynamics $\bar{\nu} = \dddot{y}_d + \alpha_2\ddot{y}_d + \alpha_1\dot{y}_d + \alpha_0 y_d - \alpha_2\ddot{y} - \alpha_1\dot{y} - \alpha_0 y$ where α_i are positive constants. Note that this choice of $\bar{\nu}$ ensures that the output exponentially converges to the desired trajectory. Rearranging $\bar{\nu}$ using Eqs.24b-24d gives

$$\nu = B(\alpha_2x_1 + x_2)x_{4d}^2 - BGx_{4d}cosx_3 \tag{25}$$

where $\nu = y_d^{(3)} + \alpha_2\ddot{y}_d + \alpha_1\dot{y}_d + \alpha_0 y_d + \alpha_2 BGsinx_3 - \alpha_1 x_2 - \alpha_0 y$, which is quadratic in the internal state x_{4d}. Using the procedure proposed in Reference. 21 the desired internal state is computed as

$$x_{4d} = \frac{BGcosx_3 - \sqrt{(BG\cos x_3)^2 + 4B(\alpha_2x_1 + x_2)\nu}}{2B(\alpha_2x_1 + x_2)} \tag{26}$$

In order to enforce the condition that the internal state follows the desired x_{4d}, the control variable u is designed such that the fast subsystem

$$x_4' = u \tag{27}$$

is asymptotically stable about x_{4d}. Since the control appears linearly, proportional control is chosen

$$u = -\beta(x_4 - x_{4d})$$

2. *Results and Discussion*

The desired trajectory is $y_d(t) = A\cos(\frac{\pi t}{5})$ with $A = 1, 2$. The constants are chosen as $\alpha_2 = 6, \alpha_1 = 12, \alpha_0 = 8, \beta = 8$. Note that these constants are chosen such that the time-scale behaviour is preserved in the closed-loop system. The control torque *tau* is assumed to have a time constant of $0.05s$ and position limits of ± 1. Figures 1-5 present the simulation results. The position output and the tracking error is shown in Figures 1-2. Notice that after the transient settles out perfect position tracking is achieved. This perfect output tracking indicates that the internal states are bounded and follow their desired values closely, as seen in Figures 3-4. The error between the desired internal state x_{4d} and the actual system response for both the cases is within ± 0.001. The control input required to accomplish the exact output tracking is shown in Figure 5. Notice that the torque computed is bounded and within constrained limits. The peaks around the first few seconds are due to the arbitrarily chosen initial conditions, and not the equilibrium solution for the system.

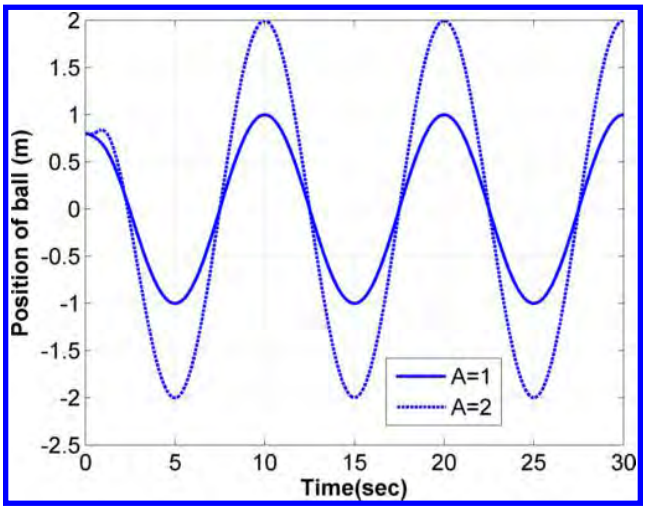


Figure 1. Position of Ball, Beam-Ball Example

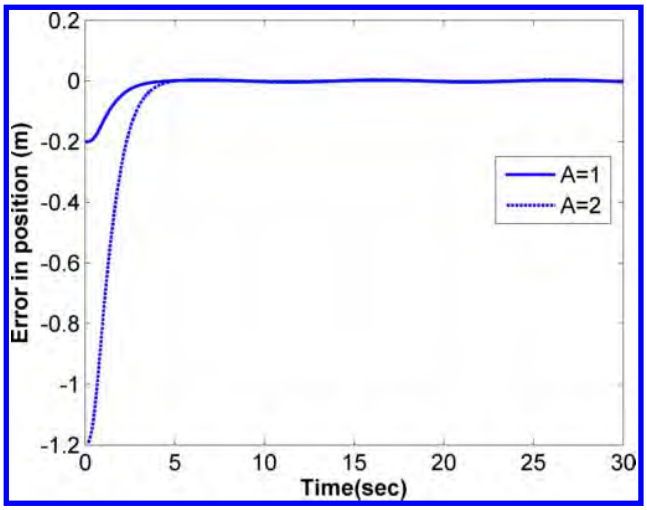


Figure 2. Tracking Error, Beam-Ball Example

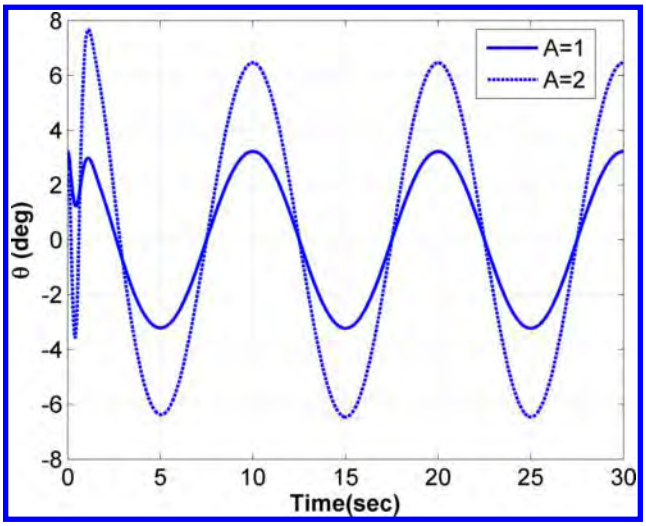


Figure 3. Roll Angle,Beam-Ball Example

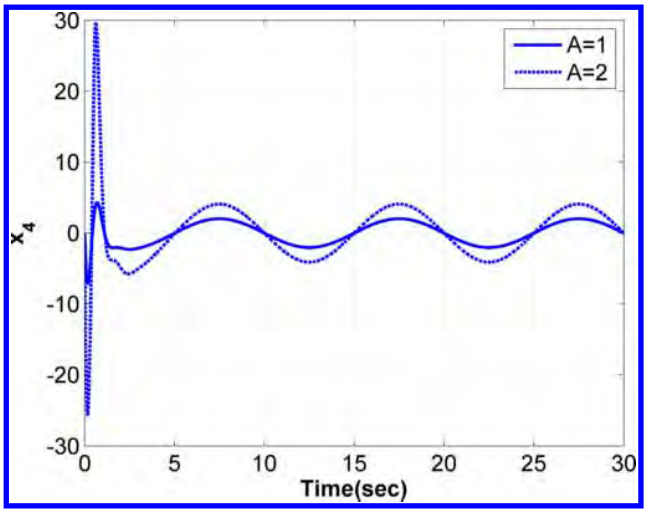


Figure 4. Internal State, Beam-Ball Example

C. Nonlinear Aircraft Tracking

The purpose of this example is to test the performance of the proposed controller for the longitudinal axis Conventional Take-off and Landing (CTOL) Aircraft, which is a Douglas DC-8.[4] The aircraft model has

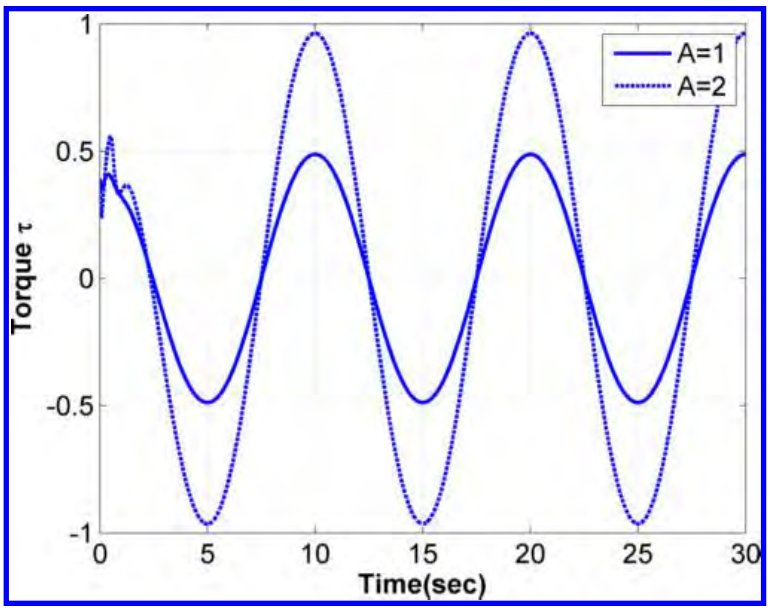


Figure 5. Torque, Beam-Ball Example

three degrees-of-freedom: horizontal and vertical position (x, z), and pitch attitude angle θ. The objective is to control the translational kinematics while stabilizing the unstable rotational dynamics. The two available controls are thrust u_1, and pitching moment u_2. The aircraft model is described by the following first-order differential equations:

$$\dot{x} = u \tag{28a}$$

$$\dot{u} = \frac{\cos\theta\,(u_1 - D\cos\alpha + L\sin\alpha) - \sin\theta\,(-F_y + D\sin\alpha + L\cos\alpha)}{m} \tag{28b}$$

$$\dot{z} = w \tag{28c}$$

$$\dot{w} = \frac{\sin\theta\,(u_1 - D\cos\alpha + L\sin\alpha) + \cos\theta\,(-F_y + D\sin\alpha + L\cos\alpha)}{m} - g \tag{28d}$$

$$\dot{\theta} = q \tag{28e}$$

$$\dot{q} = \frac{u_2}{J} \tag{28f}$$

where u,w are the forward and vertical velocities, q is the pitch rate and $\alpha = \theta - \tan^{-1}\frac{w}{u}$ is the angle-of-attack. The aerodynamic forces and physical constants are chosen[22] and given as $m = 85000kg$, $J = 4 * 10^6 kgm^2$, $g = 9.81ms^{-2}$, $L = a_L(u^2 + w^2)(1 + c\alpha)$, $D = a_D(u^2 + w^2)(1 + b(1 + c\alpha)^2)$ with $F_y = \frac{0.3mg}{J}u_2$, $a_L = \frac{30m}{g}$, $a_D = \frac{2m}{g}$, $b = 0.01$, and $c = 6$. The system of Eqs.28 is given in the desired compact normal form. The non-minimum phase characteristic is due to the pitching moment inducing a parasitic downward force onto the system.

1. *Controller Design*

The first step in the control design is to write Eqs.28 in singularly perturbed form. To determine whether the rotation θ can be employed as a control, a time-scale analysis similar to the beam-ball example is carried out. Let the reference quantities be denoted as $(t_0, x_0, z_0, u_0, w_0, \theta_0, q_0)$, (u_{10}, u_{20}), and $(D_0 = L_0 = u_{10} = F_{y0})$. The non-dimensionalized equations are given as

$$\dot{\hat{x}} = \left[\frac{t_0 u_0}{x_0}\right]\hat{u} \tag{29a}$$

$$\dot{\hat{u}} = \left[\frac{L_0 t_0}{m u_0}\right]\Big(\cos(\theta)(\hat{u}_1 - \hat{D}\cos(\alpha) + \hat{L}\sin(\alpha)) - \sin(\theta)(-\hat{F}_y + \hat{D}\sin(\alpha) + \hat{L}\cos(\alpha))\Big) \tag{29b}$$

$$\dot{\hat{z}} = \left[\frac{t_0 w_0}{z_0}\right]\hat{w} \tag{30a}$$

$$\dot{\hat{w}} = \left[\frac{t_0 L_0}{m w_0}\right]\left(\sin(\theta)(\hat{u}_1 - \hat{D}\cos(\alpha) + \hat{L}\sin(\alpha)) + \cos(\theta)(-\hat{F}_y + \hat{D}\sin(\alpha) + \hat{L}\cos(\alpha))\right) - \frac{t_0}{w_0}g \tag{30b}$$

$$\dot{\hat{\theta}} = \left[\frac{t_0}{q_0}\theta_0\right]\hat{q} \tag{30c}$$

$$\dot{\hat{q}} = \left[\frac{t_0 u_{20}}{J q_0}\right]\hat{u}_2 \tag{30d}$$

Let $\left[\frac{L_0 t_0}{m u_0}\right] = \left[\frac{t_0 L_0}{m w_0}\right] = 1$. This is a valid assumption as $\left[\frac{m u_0}{t_0}\right]$ also has units of force. Similarly, $\left[\frac{t_0 u_0}{x_0}\right] = \left[\frac{t_0 w_0}{z_0}\right] = 1$. Next, since the angular quantities are small, $\left[\frac{t_0}{q_0}\theta_0\right] = 1$ and $\left[\frac{t_0 u_{20}}{J q_0}\right] = \frac{1}{\epsilon}$ is a very large quantity. Thus, it can be concluded that the rotational dynamics evolve faster and the pitch rate evolves faster than the translational velocities, where (x, y, u, w, θ) evolve at a rate of $O(1)$. This conclusion permits the assumption of pitch attitude angle as the 'pseudo-control'. Thus pitch rate is the control input for the desired pitch attitude angle.

Let $\mathbf{e_u} = u - u_d$ and $e_w = w - w_d$ denote the errors between the actual and the desired output and rewrite the system of Eqs.28 as

$$\dot{x} = u \tag{31a}$$

$$\dot{e}_u = \frac{\cos\theta\,(u_1 - D\cos\alpha + L\sin\alpha) - \sin\theta\,(-F_y + D\sin\alpha + L\cos\alpha)}{m} - \dot{u}_d \tag{31b}$$

$$\dot{z} = w \tag{31c}$$

$$\dot{e}_w = \frac{\sin\theta\,(u_1 - D\cos\alpha + L\sin\alpha) + \cos\theta\,(-F_y + D\sin\alpha + L\cos\alpha)}{m} - g - \dot{w}_d \tag{31d}$$

$$\dot{\theta} = q \tag{31e}$$

$$\epsilon\dot{q} = \frac{u_2}{J} \tag{31f}$$

where ϵ is introduced to signify the time difference. Let θ_d and q_d indicate the desired internal states. Thus, the resulting reduced slow subsystem becomes

$$\dot{x} = u \tag{32a}$$

$$\dot{e}_u = \frac{\cos\theta\,(u_1 - D\cos\alpha + L\sin\alpha) - \sin\theta\,(-F_y + D\sin\alpha + L\cos\alpha)}{m} - \dot{u}_d \tag{32b}$$

$$\dot{z} = w \tag{32c}$$

$$\dot{e}_w = \frac{\sin\theta\,(u_1 - D\cos\alpha + L\sin\alpha) + \cos\theta\,(-F_y + D\sin\alpha + L\cos\alpha)}{m} - g - \dot{w}_d \tag{32d}$$

$$\dot{\theta} = q_d \tag{32e}$$

$$0 = \frac{u_2}{J} \tag{32f}$$

that further simplifies to

$$\dot{x} = u \tag{33a}$$

$$\dot{e}_u = \frac{\cos\theta_d\,(u_1 - D\cos\alpha + L\sin\alpha) - \sin\theta_d\,(D\sin\alpha + L\cos\alpha)}{m} - \dot{u}_d \tag{33b}$$

$$\dot{z} = w \tag{33c}$$

$$\dot{e}_w = \frac{\sin\theta_d\,(u_1 - D\cos\alpha + L\sin\alpha) + \cos\theta_d\,(D\sin\alpha + L\cos\alpha)}{m} - g - \dot{w}_d \tag{33d}$$

$$\dot{\theta} = q_d \tag{33e}$$

Using trigonometric identities and rearranging

$$\dot{x} = u \tag{34a}$$
$$\dot{z} = w \tag{34b}$$
$$\dot{e}_u = \frac{u_1 \cos\theta_d - D\cos(\theta_d - \alpha) - L\sin(\theta - \alpha)}{m} - \dot{u}_d \tag{34c}$$
$$\dot{e}_w = \frac{u_1 \sin\theta_d - D\sin(\theta_d - \alpha) + L\cos(\theta_d - \alpha)}{m} - g - \dot{w}_d \tag{34d}$$
$$\dot{\theta} = q_d \tag{34e}$$

In order to force the errors to asymptotically approach the origin, design the desired θ_d and thrust u_1 such that

$$-\lambda_1 e_u = \frac{u_1 \cos\theta_d - D\cos(\theta_d - \alpha) - L\sin(\theta - \alpha)}{m} - \dot{u}_d \tag{35}$$
$$-\lambda_2 e_w = \frac{u_1 \sin\theta_d - D\sin(\theta_d - \alpha) + L\cos(\theta_d - \alpha)}{m} - g - \dot{w}_d \tag{36}$$

Notice that Eqs.35-36 are independent of q_d and hence it can be computed as

$$q_d = -\lambda_3(\theta - \theta_d) \tag{37}$$

This procedure completes the design of the controller for the slow subsystem. The fast subsystem controller now needs to be designed such that the fast state q follows q_d asymptotically. This can be achieved by computing the required moment as

$$u_2 = -\lambda_4 J(q - q_d) \tag{38}$$

In the equations above λ denotes the desired closed-loop characteristics.

2. *Results and Discussion*

The control objective is to perform a climbing maneuver that tracks a constant velocity.[22] The forward velocity is commanded to be constant at $145ms^{-1}$ and the vertical velocity is chosen as $w_d = \frac{125\pi}{60}\sin(\frac{\pi t}{60})$. The closed-loop characteristics are chosen such that the time-scale properties are preserved: $\lambda_1 = 4$, $\lambda_2 = 4$, $\lambda_3 = 4$ and $\lambda_4 = 6$. The actuators are assumed to have first-order dynamics with a $0.05s$ time constant. The nonlinear equations Eqs.35-36 were solved using the constrained optimizer `fsolve` in MATLAB with arbitrarily chosen initial conditions of $(100, 0.05)$ for thrust and θ_d respectively. The small angle assumption was made for angle-of-attack to ease the computational burden. The results are presented in Figures 6-12. Figures 6-7 compare the forward and vertical velocities to their respective desired references. Close tracking is demonstrated with an error of $0.002ms^{-1}$ in forward velocity and $\pm 0.049ms^{-1}$ in vertical velocity. The control commands are presented in Figures 8-9. Thrust is seen to settle down to its equilibrium value of $3.694 \times 10^8 N$ while the moment varies accordingly to provide sufficient upward force. As expected the directions of the vertical velocity and the applied moment are opposite: positive moment induces a negative downward force and reduces the vertical velocity to its desired value. Therefore, for the first 60 seconds the moment is negative, after which it changes sign. Perfect output tracking indicates that the internal aircraft states are stable. This behaviour is seen in Figure 10-11. The pitch attitude angle (Figure 11) is bounded and behaves as expected. A climb produces an increase in pitch attitude angle, and a descent produces a negative value. The pitch rate behaviour seen in Figure 10 agrees with the commanded trajectory. In comparison with results published in Reference 22, this exact internal state trajectory was obtained using the offline technique proposed by Devasia et.al.[15] The complete two-dimensional trajectory is shown in Figure 12.

V. Conclusions

A control formulation for output tracking of a general class of nonlinear non-minimum phase systems was developed. The desired internal-state reference and feedback control to stabilize the unstable internal dynamics were posed as an asymptotic slow tracking problem for singularly perturbed systems. Controller performance was demonstrated through numerical simulation for two benchmark nonlinear examples.

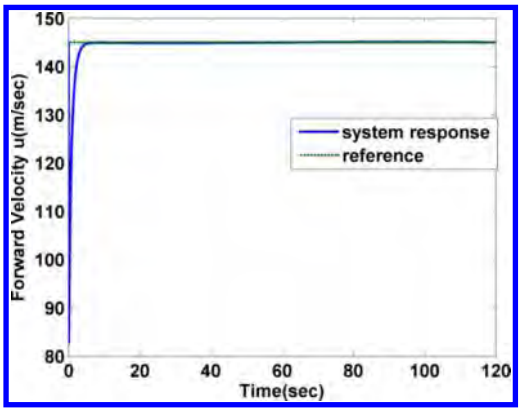


Figure 6. Forward Velocity, Aircraft Example

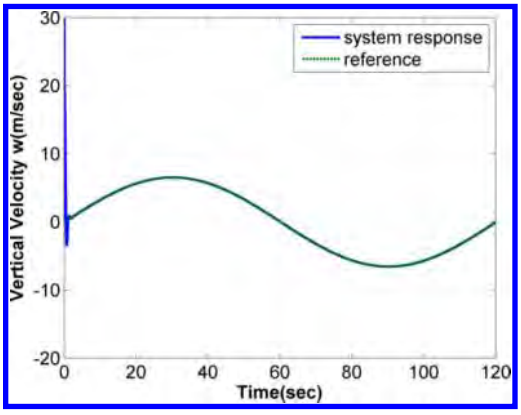


Figure 7. Vertical Velocity, Aircraft Example

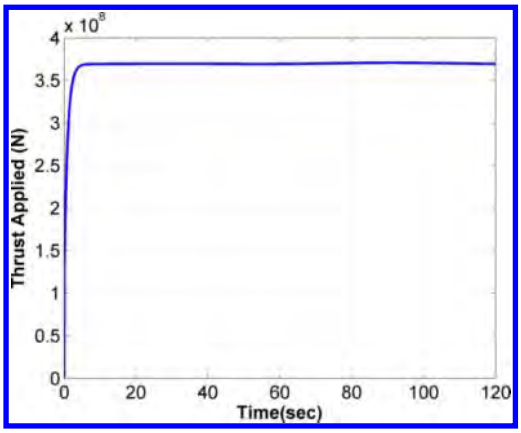


Figure 8. Applied Thrust, Aircraft Example

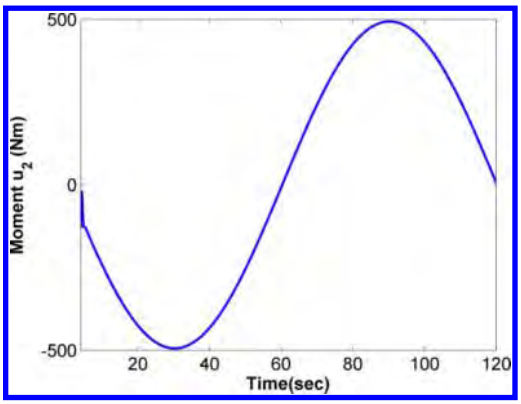


Figure 9. Applied Moment (after transient), Aircraft Example

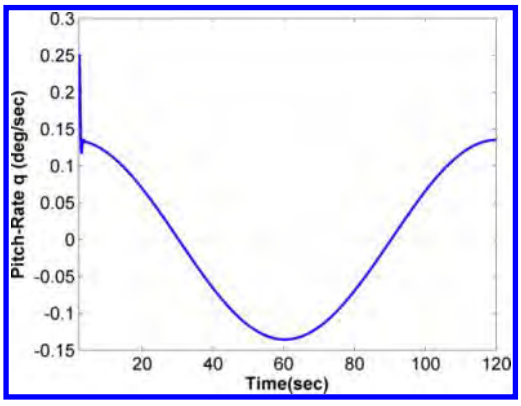


Figure 10. Pitch Rate, Aircraft Example

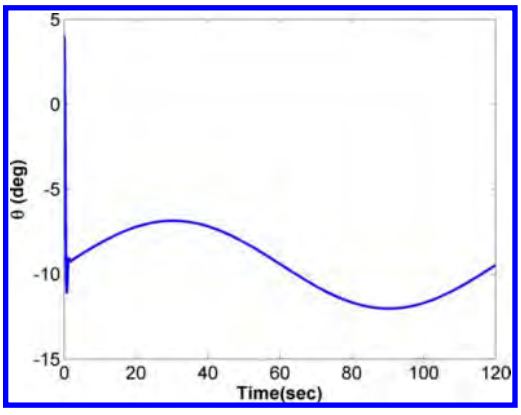


Figure 11. Pitch Attitude, Aircraft Example

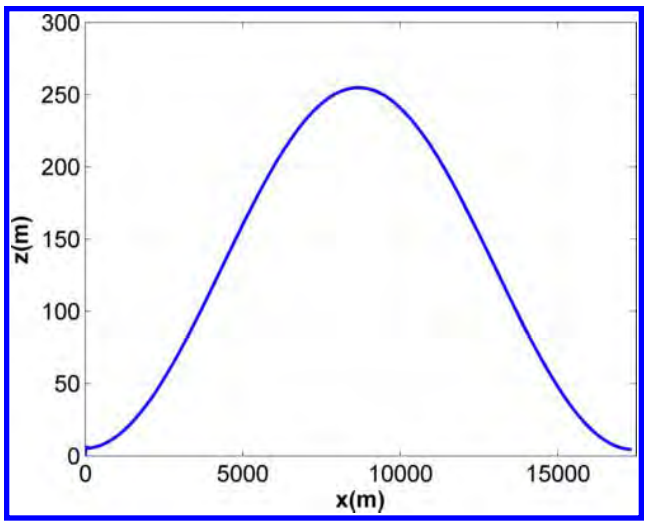


Figure 12. Two-Dimensional Trajectory, Aircraft Example

Based on the results presented in the paper, the following conclusions are drawn. The tracking error for the beam-ball example was demonstrated to remain within |0.03| at all times, and perfect output tracking was demonstrated. This perfect output tracking was a result of perfect internal state tracking that was achieved by the nonlinear feedback law. This same behaviour was also seen for the aircraft example, where the tracking error was within |0.002| for the forward velocity and |0.049| for the vertical velocity. For both of these benchmark problems, the controller demonstrated asymptotic tracking irrespective of the desired reference trajectory. The controller was causal and did not require any preview of the desired reference.

Acknowledgements

This material is based upon work supported by the U.S. Air Force Office of Scientific Research under contract FA9550-08-1-0038, with technical monitor Dr. Fariba Fahroo. This support is gratefully acknowledged by the authors. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Air Force.

References

[1]Slotine, J.-J. E. and Li, W., *Applied Nonlinear Control*, Prentice Hall, 1991.

[2]Lee, J. and Ha, I., "Autopilot design for highly maneuvering STT missiles via singular perturbation-like technique," *IEEE Transactions on Control System Technology*, Vol. 7, 1999, pp. 527–541.

[3]Hauser, J., Sastry, S. S., and Meyer, G., "Nonlinear control design for slightly nonminimum phase systems: Application to V/STOL aircraft," *Automatica*, Vol. 28, 1992, pp. 665–679.

[4]Tomlin, C., Lygeors, J., Benvenuti, L., and Sastry, S., "Output Tracking for a Non-Minimum Phase Dyanmic CTOL Aircraft Model," *Proceedings of the 34th Conference on Decision and Control*, 1995, pp. 1867–1872, New Orleans, LA.

[5]Benvenuti, L., Benedetto, M. D. D., and Grizzle, J. W., "Approximate output tracking for nonlinear nonminimum phase systems with an application to flight control," *Journal of Nonlinear Robust Control*, Vol. 4, 1994, pp. 397–414.

[6]Hedrick, J. and Gopalswamy, S., "Nonlinear Flight Control Design via Sliding Manifolds," *Journal of Guidance*, Vol. 13, 1990, pp. 850–858.

[7]Doyle, F. J., Allgöwer, F., and Morari, M., "A normal form approach to approximate input-output linearization for maximum phase nonlinear SISO systems," *IEEE Transactions on Automatic Control*, Vol. 41, 1996, pp. 305–309.

[8]Shkolnikov, I. A. and Shtessel, Y. B., "Nonminimum phase tracking in MIMO systems with square input- output dynamics via dynamic sliding manifolds," *Journal of Franklin Institute*, 2000, pp. 43–56.

[9]Shkolnikov, I. A. and Shtessel, Y. B., "Aircraft Nonminimum Phase Control in Dynamic Sliding Manifolds," *Journal of Guidance, Control and Dynamics*, Vol. 24, 2001, pp. 566–572.

[10]Zhu, B., Wang, X., and Cai, K.-Y., "Approximate trajectory tracking of input-disturbed PVTOL aircraft with delayed attitude measurements," *International Journal Robust Nonlinear Control*, Vol. 20, 2010, pp. 1610–1621.

[11]Lee, J.-I. and Ha, I.-J., "A Novel Approach to Control of Nonminimum-Phase Nonlinear Systems," *IEEE Transactions on Automatic Control*, Vol. 47, No. 9, 2002, pp. 1480–1486.

[12]Sepulchre, R., "Slow peaking and low-gain designs for global stabilization of nonlinear systems," *IEEE Transactions on Automatic Control*, Vol. 45, 2000, pp. 453–461.

[13]Teel, A. R., "A nonlinear small gain theorem for the analysis of control systems with saturation," *IEEE Transactions on Automatic Control*, Vol. 41, 1996, pp. 1256–1270.

[14]Lin, W. and Li, X., "Synthesis of upper-triangular nonlinear systems with marginally unstable free dynamics using state-dependent saturation," *International Journal of Control*, Vol. 72, 1999, pp. 1078–1086.

[15]Devasia, S., Chen, D., and Paden, B., "Nonlinear inversion-based output tracking," *IEEE Transactions on Automatic Control*, Vol. 41, 1996, pp. 930–942.

[16]Zoua, Q. and Devasia, S., "Precision preview-based stable-inversion for nonlinear nonminimum-phase systems: TheVTOL example," *Automatica*, Vol. 43, 2007, pp. 117–127.

[17]Choi, H.-L. and Lim, J.-T., "Gain scheduling control of nonlinear singularly perturbed time-varying systems with derivative information," *International Journal of Systems Science*, Vol. 36, No. 6, 2005, pp. 357–364.

[18]Fenichel, N., "Geometric Singular Perturbation Theory for Ordinary Differential Equations," *Journal of Differential Equations*, Vol. 31, 1979, pp. 53–98.

[19]Kokotovic, P., Khalil, H. K., and Reilly, J. O., *Singular Perturbation Methods in Control: Analysis and Design*, Academic Press, 1986.

[20]Khalil, H. K., *Nonlinear Systems*, Prentice Hall, Upper Saddle River, NJ, 3rd ed., December 2001.

[21]Moulay, E. and Perruquetti, W., "Stabilization of Nonaffine Systems: A Constructive Method for Polynomial Systems," *IEEE Transactions on Automatic Control*, Vol. 50, No. 4, 2005, pp. 520–526.

[22]Al-Hiddabi, S. A., "Trajectory Tracking Control and Maneuver Regulation Control for the CTOL Aircraft Model," *Proceedings of the 38th Conference on Decision & Control*, 1999, pp. 1958–1963, Phoenix, Arizona.

2012 American Control Conference
Fairmont Queen Elizabeth, Montréal, Canada
June 27-June 29, 2012

Tracking Control Design for Non-Standard Nonlinear Singularly Perturbed Systems

Anshu Siddarth and John Valasek

Abstract— **Tracking control laws for a general class of nonlinear singularly perturbed systems are developed. No assumptions concerning the nonlinearity of the system is made. The effect of the different speeds of controllers and nonlinear actuator dynamics is studied and asymptotic stabilization is shown using Lyapunov methods. Design procedure and performance of the proposed technique is evaluated against composite control method. Results indicate that the proposed technique applies both to standard and non-standard forms of singularly perturbed systems.**

I. INTRODUCTION

Analysis and control of singularly perturbed systems has received considerable attention in literature [1]. The common approach is to design two separate controllers for each of the two lower-order subsystems and then apply their composite or sum to the full-order system. The composite control technique[2] guarantees asymptotic stability for standard singularly perturbed systems or for systems wherein the algebraic problem has a unique root for the fast variables in the region of interest. In literature this assumption is satisfied by either assuming that a unique root for the fast states exists [3] or assuming that the system dynamics is nonlinear only in the slow states [4]. However, this root is a set of fixed points of the fast dynamics expressed as a smooth function of the slow variables and the control inputs, and hence is not always unique nor guaranteed to exist. Consequently one is required to choose an isolated manifold in order to design a stabilizing control structure for the slow subsystem. This not only requires substantial system knowledge but also restricts the results to a local domain. Furthermore, analytical determination of this manifold is restricted by the nonlinearity of the system. In such cases, it has been shown that only ultimate boundedness of the signals maybe concluded [5].

This paper proposes an alternate approach for control design of non-standard forms of singularly perturbed systems. The proposed approach avoids analytical computation of the manifold by considering it as an additional control variable. This idea is motivated by singularly perturbed systems such as aircraft wherein the fast states appear linearly in the slow dynamics. Reference [6] successfully designed nonlinear flight trajectories using angular rates as control variables, although, the effect of control variables on the slow variables was neglected. More recently ultimate uniformly bounded results were concluded [7] using similar ideas while assuming that the set of nonlinear algebraic equations can be solved for the control variables and the fast controller was designed using gain-scheduling.

This paper makes three major contributions. First, the approach developed here employs the reduced-order technique without imposing any assumptions about the solutions of the transcendental equations or the effect of the control variables. By computing the slow manifold upon which the fast states must be restricted for asymptotic tracking and ensuring that this manifold is the equilibrium of the system uniformly, control objective is accomplished. Second, controllers with different speeds are addressed in comparison to composite control technique that requires all control variables to be sufficiently fast. Third, the control laws are computed using Lyapunov-based designs that are able to capture the nonlinear behaviour that is lost in the linearization of the system. Owing to this, the global or local nature of results are relaxed from the complexities of analytic construction of the manifold and are entirely a consequence of the choice of underlying controllers for the reduced-order models. Additionally, the control laws developed in this paper are independent of the singular perturbation parameter and an upper bound for the scalar perturbation parameter is derived as a sufficiency condition for asymptotic stability.

The paper is organized as follows. Section II mathematically formulates the control problem and briefly reviews the necessary concepts from geometric singular perturbation theory. Control laws and the main results of the paper are detailed in Section III. Section IV studies several numerical examples and qualitatively analyses the performance and design procedure of the proposed technique. Conclusions are discussed in Section V.

II. PROBLEM DESCRIPTION AND MODEL REDUCTION

A. System Description

The class of nonlinear singularly perturbed dynamical systems addressed in this paper are

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{z}, \delta); \quad \mathbf{x} \in \mathbb{R}^m, \delta \in \mathbb{R}^p \tag{1a}$$

$$\dot{\delta_1} = \mathbf{f}_{\delta_1}(\delta_1, \mathbf{u_1}); \quad \delta_1 \in \mathbb{R}^l, \mathbf{u_1} \in \mathbb{R}^l \tag{1b}$$

$$\epsilon\dot{\mathbf{z}} = \mathbf{g}(\mathbf{x}, \mathbf{z}, \delta, \epsilon); \quad \mathbf{z} \in \mathbb{R}^n \tag{1c}$$

$$\epsilon\dot{\delta_2} = \mathbf{f}_{\delta_2}(\delta_2, \mathbf{u_2}); \quad \delta_2 \in \mathbb{R}^{p-l}, \mathbf{u_2} \in \mathbb{R}^{p-l} \tag{1d}$$

This work was supported in part by the U.S. Air Force Office of Scientific Research under contract FA9550-08-1-0038 with technical monitor Dr.Fariba Fahroo.

A. Siddarth is a Graduate Research Assistant in the Vehicle Systems and Controls Laboratory, Aerospace Engineering Department, Texas A&M University, College Station, TX 77843-3141, USA anshun1@tamu.edu

J. Valasek is a Professor and Director of the Vehicle Systems and Controls Laboratory, Aerospace Engineering Department, Texas A&M University, College Station, TX 77843-3141, USA valasek@tamu.edu

where $\mathbf{x}$ is the vector of slow variables, $\mathbf{z}$ is the vector of fast variables, $\delta = [\delta_\mathbf{1}, \delta_\mathbf{2}]^T$ is the vector of actuator commands input to the system, $\mathbf{u} = [\mathbf{u_1}, \mathbf{u_2}]^T \in \mathbb{R}^p$ is the input vector that is to be computed and $\epsilon \in \mathbb{R}$ is the singular perturbation parameter that satisfies $0 < \epsilon << 1$ and is unknown. All the vector fields are assumed to be sufficiently smooth and $p \geq m$. The control objective is to drive the slow state so as to track sufficiently smooth, bounded, time-varying trajectories or, $\mathbf{x}(t) \rightarrow \mathbf{x}_r(t)$ as $t \rightarrow \infty$.

The controls have been separated into vectors δ_1 and δ_2 to consider the different speeds of the control variables, with δ_1 representing the actuators with slow dynamics and δ_2 representing actuators with relatively fast actuator dynamics. The vector fields $\mathbf{f}_{\delta_1}(.)$ and $\mathbf{f}_{\delta_2}(.)$ represent their actuator dynamics respectively. The model given in (1) represents the special case of two time-scale dynamical systems. The design procedure developed here also applies to multiple time-scale systems of the following form

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{z}, \delta) \\ \epsilon_1 \dot{\delta}_\mathbf{1} &= \mathbf{f}_{\delta_\mathbf{1}}(\delta_\mathbf{1}, \mathbf{u_1}) \\ \epsilon_2 \dot{\mathbf{z}} &= \mathbf{g}(\mathbf{x}, \mathbf{z}, \delta, \epsilon_2) \\ \epsilon_3 \dot{\delta}_\mathbf{2} &= \mathbf{f}_{\delta_\mathbf{2}}(\delta_\mathbf{2}, \mathbf{u_2}) \end{aligned}$$

where ϵ_1, ϵ_2 and ϵ_3 are singular perturbation parameters of different orders that satisfy $\epsilon_3 < \epsilon_2 < \epsilon_1$.

B. Reduced-Order Models

The system considered in (1) is labeled the *Slow System* and the independent variable t is called the *slow time-scale*. This system is equivalently written as

$$\begin{aligned} \mathbf{x}' &= \epsilon \mathbf{f}(\mathbf{x}, \mathbf{z}, \delta) && \text{(2a)} \\ \delta_\mathbf{1}' &= \epsilon \mathbf{f}_{\delta_1}(\delta_\mathbf{1}, \mathbf{u_1}) && \text{(2b)} \\ \mathbf{z}' &= \mathbf{g}(\mathbf{x}, \mathbf{z}, \delta, \epsilon) && \text{(2c)} \\ \delta_\mathbf{2}' &= \mathbf{f}_{\delta_\mathbf{2}}(\delta_\mathbf{2}, \mathbf{u_2}) && \text{(2d)} \end{aligned}$$

where $'$ represents derivative with respect to $\tau = \frac{t - t_0}{\epsilon}$ and t_0 is the initial time. Equation (2) are labeled the *Fast System* and the independent variable τ is called the *fast time-scale*. Geometric singular perturbation theory[8] examines the behaviour of these singularly perturbed systems by studying the geometric constructs of the reduced-order models which are obtained by substituting $\epsilon = 0$ in (1) and (2). This results in:

Reduced Slow Subsystem:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{z}, \delta) && \text{(3a)} \\ \dot{\delta}_\mathbf{1} &= \mathbf{f}_{\delta_1}(\delta_\mathbf{1}, \mathbf{u_1}) && \text{(3b)} \\ \mathbf{0} &= \mathbf{g}(\mathbf{x}, \mathbf{z}, \delta, 0) && \text{(3c)} \\ \mathbf{0} &= \mathbf{f}_{\delta_\mathbf{2}}(\delta_\mathbf{2}, \mathbf{u_2}) && \text{(3d)} \end{aligned}$$

Reduced Fast Subsystem:

$$\begin{aligned} \mathbf{x}' &= \mathbf{0}; \quad \delta_\mathbf{1}' = \mathbf{0} && \text{(4a)} \\ \mathbf{z}' &= \mathbf{g}(\mathbf{x}, \mathbf{z}, \delta, 0) && \text{(4b)} \\ \delta_\mathbf{2}' &= \mathbf{f}_{\delta_\mathbf{2}}(\delta_\mathbf{2}, \mathbf{u_2}) && \text{(4c)} \end{aligned}$$

The dynamics of the resulting reduced slow subsystem are restricted to $m + l$ dimensions, constrained to lie upon an $n + p - l$ dimensional smooth manifold defined by the set of points $(\mathbf{x}, \mathbf{z}, \delta) \in \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^p$ that satisfy the algebraic equations (3c),(3d):

$$\mathcal{M}_0 : \mathbf{z} = \mathbf{z}(\mathbf{x}, \delta_\mathbf{1}, \delta_\mathbf{2}); \delta_\mathbf{2} = \delta_\mathbf{2}(\mathbf{u}_2) \qquad (5)$$

This set of points is identically the fixed points of the reduced fast subsystem (4b)-(4c). Thus the manifold $\mathcal{M}_0$ is invariant [9]. Furthermore, the flow on this manifold is described by the differential equations

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{z}(\mathbf{x}, \delta_\mathbf{1}, \delta_\mathbf{2}), \delta_\mathbf{1}, \delta_\mathbf{2}(\mathbf{u}_2)) && \text{(6a)} \\ \dot{\delta}_\mathbf{1} &= \mathbf{f}_{\delta_1}(\delta_\mathbf{1}, \mathbf{u_1}) && \text{(6b)} \end{aligned}$$

if the reduced fast subsystem is stable about the manifold $\mathcal{M}_0$. If the dynamics of (6) are locally asymptotically stable about the manifold, then it can be concluded that the complete system (1) is also locally asymptotically stable [9].

III. CONTROL FORMULATION AND STABILITY ANALYSIS

Stability properties of the slow system depend upon the identification of the manifold $\mathcal{M}_0$. In general, the nonlinear set of algebraic equations (3c),(3d) possess multiple roots and the manifold $\mathcal{M}_0$ may take any of these values; hence it is not unique. One approach to ensure uniqueness is to consider the fast state as another control variable. These ideas are mathematically formulated and analyzed in this section.

A. Control Design

The first step is to transform the problem into a non-autonomous stabilization control problem. Define the tracking error signal as $\mathbf{e}(t) = \mathbf{x}(t) - \mathbf{x}_r(t)$ and express the slow system as

$$\begin{aligned} \dot{\mathbf{e}} &= \mathbf{F}(\mathbf{e}, \mathbf{z}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \delta) && \text{(7a)} \\ \dot{\delta}_\mathbf{1} &= \mathbf{f}_{\delta_1}(\delta_\mathbf{1}, \mathbf{u_1}) && \text{(7b)} \\ \epsilon \dot{\mathbf{z}} &= \mathbf{G}(\mathbf{e}, \mathbf{z}, \mathbf{x}_r, \delta, \epsilon) && \text{(7c)} \\ \epsilon \dot{\delta}_\mathbf{2} &= \mathbf{f}_{\delta_\mathbf{2}}(\delta_\mathbf{2}, \mathbf{u_2}) && \text{(7d)} \end{aligned}$$

where $\mathbf{F}(\mathbf{e}, \mathbf{z}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \delta) = \mathbf{f}(\mathbf{e} + \mathbf{x}_r, \mathbf{z}, \delta) - \dot{\mathbf{x}}_r$ and $\mathbf{G}(\mathbf{e}, \mathbf{z}, \mathbf{x}_r, \delta) = \mathbf{g}(\mathbf{e} + \mathbf{x}_r, \mathbf{z}, \delta, \epsilon)$ are Lipschitz on a domain of the state-space. Using the procedure described in Section II, the reduced slow subsystem for set of equations in (7) is obtained as

$$\begin{aligned} \dot{\mathbf{e}} &= \mathbf{F}(\mathbf{e}, \mathbf{z}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \delta) && \text{(8a)} \\ \dot{\delta}_\mathbf{1} &= \mathbf{f}_{\delta_1}(\delta_\mathbf{1}, \mathbf{u_1}) && \text{(8b)} \\ \mathbf{0} &= \mathbf{G}(\mathbf{e}, \mathbf{z}, \mathbf{x}_r, \delta, 0) && \text{(8c)} \\ \mathbf{0} &= \mathbf{f}_{\delta_\mathbf{2}}(\delta_\mathbf{2}, \mathbf{u_2}) && \text{(8d)} \end{aligned}$$

In order to ensure $\mathbf{e} = \mathbf{0}$ is an asymptotically stable equilibrium of the reduced slow system (8) define a positive-definite and decrescent Lyapunov function that satisfies

Condition 1. $V(t, \mathbf{e}) : [0, \infty) \times D_\mathbf{x} \rightarrow \mathbb{R}$ is continuously differentiable and $D_\mathbf{x} \subset \mathbb{R}^m$ contains the origin, such that

$$0 < \psi_1(||\mathbf{e}||) \leq V(t, \mathbf{e}) \leq \psi_2(||\mathbf{e}||)$$

for some **class** $\mathcal{K}$ functions $\psi_1(.)$ and $\psi_2(.)$.

Let $\delta_{\mathbf{2r}}$ represent the manifold of the equation (8d) that is defined shortly. Design a manifold $\mathbf{z} = \mathbf{z_r}(\mathbf{e}, \mathbf{x_r}, \dot{\mathbf{x}}_\mathbf{r}, \delta_{\mathbf{2r}})$ and control $\delta_\mathbf{1} = \delta_{\mathbf{1r}}(\mathbf{e}, \mathbf{x_r}, \dot{\mathbf{x}}_\mathbf{r}, \delta_{\mathbf{2r}})$ such that the slow state error system (8a) satisfies

Condition 2.

$$\frac{\partial V}{\partial t} + \frac{\partial V}{\partial \mathbf{e}}\mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \delta_{\mathbf{1r}}, \delta_{\mathbf{2r}}) \leq -\alpha_1 \psi_3^2(\mathbf{e}), \quad \alpha_1 > 0$$

where $\psi_3(.)$ is a continuous positive-definite scalar function that satisfies $\psi_3(\mathbf{0}) = 0$.

The next step is to design control $\mathbf{u_1}$ that ensures the actuator state asymptotically approaches $\delta_{\mathbf{1r}}$. Define the error in actuator state as $\mathbf{e}_{\delta_\mathbf{1}} := \delta_\mathbf{1} - \delta_{\mathbf{1r}}$ and rewrite the reduced slow error subsystem (8a),(8b) as

$$\begin{aligned} \dot{\mathbf{e}} &= \mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \delta_{\mathbf{1r}}, \delta_{\mathbf{2r}}) + \\ &\quad \mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \delta_\mathbf{1}, \delta_{\mathbf{2r}}) - \mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \delta_{\mathbf{1r}}, \delta_{\mathbf{2r}}) \quad (9a) \\ \dot{\mathbf{e}}_{\delta_\mathbf{1}} &= \mathbf{f}_{\delta_\mathbf{1}}(\delta_\mathbf{1}, \mathbf{u_1}) - \dot{\delta}_{\mathbf{1r}} \quad (9b) \end{aligned}$$

where $\dot{\delta}_{\mathbf{1r}}$, the derivative of the manifold is given as

$$\begin{aligned} \dot{\delta}_{\mathbf{1r}} &= \frac{\partial \delta_{\mathbf{1r}}}{\partial \mathbf{e}}\dot{\mathbf{e}} + \frac{\partial \delta_{\mathbf{1r}}}{\partial \mathbf{x_r}}\dot{\mathbf{x}}_\mathbf{r} + \frac{\partial \delta_{\mathbf{1r}}}{\partial \dot{\mathbf{x}}_\mathbf{r}}\ddot{\mathbf{x}}_\mathbf{r} + \frac{\partial \delta_{\mathbf{1r}}}{\partial \delta_{\mathbf{2r}}}\dot{\delta}_{\mathbf{2r}} \quad (10) \\ &= \frac{\partial \delta_{\mathbf{1r}}}{\partial \mathbf{e}}\mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \delta_\mathbf{1}, \delta_{\mathbf{2r}}) + \frac{\partial \delta_{\mathbf{1r}}}{\partial \mathbf{x_r}}\dot{\mathbf{x}}_\mathbf{r} + \frac{\partial \delta_{\mathbf{1r}}}{\partial \dot{\mathbf{x}}_\mathbf{r}}\ddot{\mathbf{x}}_\mathbf{r} \end{aligned}$$

using (9a) and the fact that $\delta_{\mathbf{2r}}$ is a fixed point of the reduced slow subsystem as it satisfies equation (8d). Conditions $1-2$ ensure that the slow error is asymptotically stabilized by the slow actuator variable $\delta_{\mathbf{1r}}$. In order to ensure the system remains asymptotically stable when $\mathbf{e}_{\delta_\mathbf{1}} \neq 0$, define a combined positive-definite decrescent Lyapunov function for equations (9a),(9b) such that $V_s(t, \mathbf{e}, \mathbf{e}_{\delta_\mathbf{1}}) : [0, \infty) \times D_\mathbf{x} \times D_{\delta_\mathbf{1}} \to \mathbb{R}$ is continuously differentiable and $D_{\delta_\mathbf{1}} \subset \mathbb{R}^l$ contains the origin

$$V_s(t, \mathbf{e}, \mathbf{e}_{\delta_\mathbf{1}}) = V(t, \mathbf{e}) + \frac{1}{2}{\mathbf{e}_{\delta_\mathbf{1}}}^T\mathbf{e}_{\delta_\mathbf{1}} \quad (11)$$

and design $\mathbf{u_1}$ such that the closed-loop reduced slow system (9) satisfies

Condition 3.

$$\frac{\partial V_s}{\partial t} + \frac{\partial V_s}{\partial \mathbf{e}}\dot{\mathbf{e}} + \frac{\partial V_s}{\partial \mathbf{e}_{\delta_\mathbf{1}}}\dot{\mathbf{e}}_{\delta_\mathbf{1}} \leq -\alpha_1\psi_3^2(\mathbf{e}) - \alpha_2\psi_4^2(\mathbf{e}_{\delta_\mathbf{1}}), \quad \alpha_2 > 0$$

where $\psi_4(.)$ is a continuous positive-definite scalar function that satisfies $\psi_4(\mathbf{0}) = 0$.

Conditions $1-3$ complete the design of control for the reduced slow subsystem. Notice that the manifold $\mathbf{z_r}(\mathbf{e}, \mathbf{x_r}, \dot{\mathbf{x}}_\mathbf{r}, \delta_{\mathbf{2r}})$ computed in the above control design is a function of the manifold $\delta_{\mathbf{2r}}$ which is unknown. From the discussion detailed in Section II, it is known that this manifold is a fixed point of the reduced fast subsystem,

$$\begin{aligned} \mathbf{e}' &= \mathbf{0}; \quad {\delta_\mathbf{1}}' = \mathbf{0} \quad (12a) \\ \mathbf{z}' &= \mathbf{G}(\mathbf{e}, \mathbf{z}, \mathbf{x}_r, \delta, 0) \quad (12b) \\ {\delta_\mathbf{2}}' &= \mathbf{f}_{\delta_\mathbf{2}}(\delta_\mathbf{2}, \mathbf{u_2}) \quad (12c) \end{aligned}$$

The complete system will have the properties of the reduced slow subsystem if the fast state asymptotically stabilizes about $\mathbf{z_r}$. This condition is enforced by designing the manifold $\delta_{\mathbf{2r}}$. Define the error in the fast state vector $\mathbf{e_z} := \mathbf{z} - \mathbf{z_r}$ and rewrite (12b) as

$${\mathbf{e_z}}' = \mathbf{G}(\mathbf{e}, \mathbf{e_z}, \mathbf{x}_r, \delta_\mathbf{1}, \delta_{\mathbf{2r}}, 0) \quad (13)$$

while noting that ${\mathbf{z_r}}' = \epsilon\dot{\mathbf{z}}_\mathbf{r} = \mathbf{0}$ for the reduced fast subsystem. Define a positive-definite and decrescent Lyapunov function that satisfies

Condition 4. $W(t, \mathbf{e}, \mathbf{e}_{\delta_\mathbf{1}}, \mathbf{e_z}) : [0, \infty) \times D_\mathbf{x} \times D_{\delta_\mathbf{1}} \times D_\mathbf{z} \to \mathbb{R}$ is continuously differentiable and $D_\mathbf{z} \subset \mathbb{R}^n$ contains the origin, such that

$$0 < \phi_1(||\mathbf{e_z}||) \leq W(t, \mathbf{e}, \mathbf{e}_{\delta_\mathbf{1}}, \mathbf{e_z}) \leq \phi_2(||\mathbf{e_z}||)$$

for some **class** $\mathcal{K}$ functions $\phi_1(.)$ and $\phi_2(.)$.
and design $\delta_{\mathbf{2r}}$ such that the closed-loop reduced fast system (13) satisfies

Condition 5.

$$\frac{\partial W}{\partial \mathbf{e_z}}\mathbf{G}(\mathbf{e}, \mathbf{e_z}, \mathbf{x}_r, \delta_\mathbf{1}, \delta_{\mathbf{2r}}, 0) \leq -\alpha_3\phi_3^2(\mathbf{e_z}), \quad \alpha_3 > 0$$

where $\phi_3(.)$ is a continuous positive-definite scalar function that satisfies $\phi_3(\mathbf{0}) = 0$.

Thus, the last step in the design procedure is to enforce that the fast actuators asymptotically stabilize about $\delta_{\mathbf{2r}}$ and the closed-loop reduced fast subsystem is uniformly stable. Define the error in the fast actuator states $\mathbf{e}_{\delta_\mathbf{2}} := \delta_\mathbf{2} - \delta_{\mathbf{2r}}$ and rewrite the closed-loop reduced fast subsystem in the error coordinates

$$\begin{aligned} {\mathbf{e_z}}' &= \mathbf{G}(\mathbf{e}, \mathbf{e_z}, \mathbf{x}_r, \delta_\mathbf{1}, \delta_{\mathbf{2r}}, 0) + \\ &\quad \mathbf{G}(\mathbf{e}, \mathbf{e_z}, \mathbf{x}_r, \delta_\mathbf{1}, \delta_\mathbf{2}, 0) - \mathbf{G}(\mathbf{e}, \mathbf{e_z}, \mathbf{x}_r, \delta_\mathbf{1}, \delta_{\mathbf{2r}}, 0) \quad (14a) \\ {\mathbf{e}_{\delta_\mathbf{2}}}' &= \mathbf{f}_{\delta_\mathbf{2}}(\delta_\mathbf{2}, \mathbf{u_2}) - {\delta_{\mathbf{2r}}}' \quad (14b) \end{aligned}$$

using the fact that the slow variables remain constant in the fast time scale and

$${\delta_{\mathbf{2r}}}' = \frac{\partial \delta_{\mathbf{2r}}}{\partial \mathbf{e_z}}{\mathbf{e_z}}' \quad (15)$$

Define a positive-definite decrescent combined Lyapunov function $W_f(t, \mathbf{e}, \mathbf{e}_{\delta_\mathbf{1}}, \mathbf{e_z}, \mathbf{e}_{\delta_\mathbf{2}}) : [0, \infty) \times D_\mathbf{x} \times D_{\delta_\mathbf{1}} \times D_\mathbf{z} \times D_{\delta_\mathbf{2}} \to \mathbb{R}$ for the reduced fast subsystem (14) that is continuously differentiable and $D_{\delta_\mathbf{2}} \subset \mathbb{R}^{p-l}$ contains the origin

$$W_f(t, \mathbf{e}, \mathbf{e}_{\delta_\mathbf{1}}, \mathbf{e_z}, \mathbf{e}_{\delta_\mathbf{2}}) = W(t, \mathbf{e}, \mathbf{e}_{\delta_\mathbf{1}}, \mathbf{e_z}) + \frac{1}{2}{\mathbf{e}_{\delta_\mathbf{2}}}^T\mathbf{e}_{\delta_\mathbf{2}} \quad (16)$$

Design $\mathbf{u_2}$ such that the closed-loop reduced fast system (14) satisfies

Condition 6.

$$\frac{\partial W_f}{\partial \mathbf{e_z}}{\mathbf{e_z}}' + \frac{\partial W_f}{\partial \mathbf{e}_{\delta_\mathbf{2}}}{\mathbf{e}_{\delta_\mathbf{2}}}' \leq -\alpha_3\phi_3^2(\mathbf{e_z}) - \alpha_4\phi_4^2(\mathbf{e}_{\delta_\mathbf{2}}), \quad \alpha_4 > 0$$

B. Stability Analysis

The following theorem summarizes the main result of the paper.

Theorem 1. *Suppose the control* $\mathbf{u}$ *of the system (1) is designed according to the Conditions* $1-14$*. Then for all initial conditions,* $(\mathbf{e}, \mathbf{e}_{\delta_1}, \mathbf{e_z}, \mathbf{e}_{\delta_2}) \in D_{\mathbf{x}} \times D_{\delta_1} \times D_{\mathbf{z}} \times D_{\delta_2}$*, the control uniformly asymptotically stabilizes the nonlinear singularly perturbed system (1) and equivalently drives the slow state* $\mathbf{x}(t) \to \mathbf{x}_r(t)$ *for all* $\epsilon < \epsilon^*$*, that is defined by the inequality given in (24).*

Proof: The closed-loop complete system in the error coordinates is given as

$$\begin{aligned}
\dot{\mathbf{e}} &= \mathbf{F}(\mathbf{e}, \mathbf{e_z} + \mathbf{z}_r, \mathbf{x}_r, \dot{\mathbf{x}}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}) && \text{(17a)}\\
\dot{\mathbf{e}}_{\delta_1} &= \mathbf{f}_{\delta_1}(\mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{u_1}) - \dot{\delta}_{\mathbf{1rC}} && \text{(17b)}\\
\epsilon\dot{\mathbf{e}}_{\mathbf{z}} &= \mathbf{G}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}, \epsilon) \\
&\quad - \epsilon\dot{\mathbf{z}}_{\mathbf{r}} && \text{(17c)}\\
\epsilon\dot{\mathbf{e}}_{\delta_2} &= \mathbf{f}_{\delta_2}(\mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}, \mathbf{u_2}) - \epsilon\dot{\delta}_{\mathbf{2rC}} && \text{(17d)}
\end{aligned}$$

where the subscript 'C' is added to note the difference between (17b)-(17d) and (10) and (15). These expressions for the complete system are noted as

$$\begin{aligned}
\dot{\delta}_{\mathbf{1rC}} &= \frac{\partial\delta_{\mathbf{1r}}}{\partial t} + \frac{\partial\delta_{\mathbf{1r}}}{\partial\mathbf{e}}\dot{\mathbf{e}} + \\
&\quad \frac{\partial\delta_{\mathbf{1r}}}{\partial\mathbf{x_r}}\dot{\mathbf{x}}_{\mathbf{r}} + \frac{\partial\delta_{\mathbf{1r}}}{\partial\dot{\mathbf{x}}_{\mathbf{r}}}\ddot{\mathbf{x}}_{\mathbf{r}} + \frac{\partial\delta_{\mathbf{1r}}}{\partial\mathbf{e_z}}\dot{\mathbf{e}}_{\mathbf{z}} && \text{(18)}\\
\dot{\delta}_{\mathbf{2rC}} &= \frac{\partial\delta_{\mathbf{2r}}}{\partial t} + \frac{\partial\delta_{\mathbf{2r}}}{\partial\mathbf{e}}\dot{\mathbf{e}} + \frac{\partial\delta_{\mathbf{2r}}}{\partial\mathbf{x_r}}\dot{\mathbf{x}}_{\mathbf{r}} + \frac{\partial\delta_{\mathbf{2r}}}{\partial\dot{\mathbf{x}}_{\mathbf{r}}}\ddot{\mathbf{x}}_{\mathbf{r}} + \\
&\quad \frac{\partial\delta_{\mathbf{2r}}}{\partial\mathbf{e}_{\delta_1}}\dot{\mathbf{e}}_{\delta_1} + \frac{\partial\delta_{\mathbf{2r}}}{\partial\mathbf{e_z}}\dot{\mathbf{e}}_{\mathbf{z}} && \text{(19)}
\end{aligned}$$

Closed-loop system stability of the system states is analyzed using the composite Lyapunov function approach[10]. Consider a Lyapunov function candidate

$$\nu(t, \mathbf{e}, \mathbf{e}_{\delta_1}, \mathbf{e_z}, \mathbf{e}_{\delta_2}) = V_s(t, \mathbf{e}, \mathbf{e}_{\delta_1}) + W_f(t, \mathbf{e}, \mathbf{e}_{\delta_1}, \mathbf{e_z}, \mathbf{e}_{\delta_2}) \quad (20)$$

for the complete closed-loop system. From the properties of V_s and W_f it follows that $\nu(t, \mathbf{e}, \mathbf{e}_{\delta_1}, \mathbf{e_z}, \mathbf{e}_{\delta_2})$ is positive-definite and decrescent. The derivative of ν along the trajectories of (17) is given by

$$\begin{aligned}
\dot{\nu} &= \frac{\partial V_s}{\partial t} + \frac{\partial V_s}{\partial\mathbf{e}}\dot{\mathbf{e}} + \frac{\partial V_s}{\partial\mathbf{e}_{\delta_1}}\dot{\mathbf{e}}_{\delta_1} + \frac{\partial W_f}{\partial t} + \frac{\partial W_f}{\partial\mathbf{e}}\dot{\mathbf{e}} \\
&\quad + \frac{\partial W_f}{\partial\mathbf{e}_{\delta_1}}\dot{\mathbf{e}}_{\delta_1} + \frac{1}{\epsilon}\frac{\partial W_f}{\partial\mathbf{e_z}}\mathbf{e_z}' + \frac{1}{\epsilon}\frac{\partial W_f}{\partial\mathbf{e}_{\delta_2}}\mathbf{e}_{\delta_2}' && \text{(21)}
\end{aligned}$$

Note that the vector fields in (17a) and (17c) can also be expressed as

$$\begin{aligned}
&\mathbf{F}(\mathbf{e}, \mathbf{e_z} + \mathbf{z}_r, \mathbf{x}_r, \dot{\mathbf{x}}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}) = \\
&\mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \delta_{\mathbf{1r}}, \delta_{\mathbf{2r}}) + \mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \delta_{\mathbf{2r}}) \\
&- \mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \delta_{\mathbf{1r}}, \delta_{\mathbf{2r}}) - \mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \delta_{\mathbf{2r}}) \\
&+ \mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}) && \text{(22)}\\
&+ \mathbf{F}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}) \\
&- \mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}})
\end{aligned}$$

$$\begin{aligned}
&\mathbf{G}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}, \epsilon) = \\
&\mathbf{G}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \delta_{\mathbf{2r}}, 0) \\
&+ \mathbf{G}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}, 0) \\
&- \mathbf{G}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \delta_{\mathbf{2r}}, 0) \\
&+ \mathbf{G}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}, \epsilon) \\
&- \mathbf{G}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}, 0) \qquad \text{(23)}
\end{aligned}$$

Suppose that Lyapunov functions V_s and W_f also satisfy the following conditions with $\beta_i \geq 0$ and $\gamma_i \geq 0$

Condition 7.

$$\begin{aligned}
&\frac{\partial V_s}{\partial\mathbf{e}}\mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}) - \\
&\frac{\partial V_s}{\partial\mathbf{e}}\mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_{\mathbf{r}}, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \delta_{\mathbf{2r}}) \leq \beta_1\psi_3(\mathbf{e})\phi_4(\mathbf{e}_{\delta_2})
\end{aligned}$$

Condition 8.

$$\begin{aligned}
&\frac{\partial V_s}{\partial\mathbf{e}}\mathbf{F}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}) - \\
&\frac{\partial V_s}{\partial\mathbf{e}}\mathbf{F}(\mathbf{e}, \mathbf{z_r}, \mathbf{x}_r, \dot{\mathbf{x}}_r, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}) \leq \beta_2\psi_3(\mathbf{e})\phi_3(\mathbf{e_z})
\end{aligned}$$

Condition 9.

$$\begin{aligned}
&\frac{\partial V_s}{\partial\mathbf{e}_{\delta_1}}\frac{\partial\delta_{\mathbf{1r}}}{\partial\mathbf{e_z}}\dot{\mathbf{e}}_{\mathbf{z}} \leq \beta_3\psi_3(\mathbf{e})\psi_4(\mathbf{e}_{\delta_1}) + \beta_4\psi_4(\mathbf{e}_{\delta_1})\phi_3(\mathbf{e_z}) \\
&+ \gamma_1\psi_4^2(\mathbf{e}_{\delta_1})
\end{aligned}$$

Condition 10.

$$\begin{aligned}
&\frac{\partial W_f}{\partial\mathbf{e_z}}\mathbf{G}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x_r}, \dot{\mathbf{x}}_{\mathbf{r}}, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}, \epsilon) - \\
&\frac{\partial W_f}{\partial\mathbf{e_z}}\mathbf{G}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x_r}, \dot{\mathbf{x}}_{\mathbf{r}}, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}, \mathbf{0}) \\
&\leq \epsilon\gamma_2\phi_3^2(\mathbf{e_z}) + \epsilon\beta_5\psi_3(\mathbf{e})\phi_3(\mathbf{e_z}) + \epsilon\beta_6\psi_4(\mathbf{e}_{\delta_1})\phi_3(\mathbf{e_z}) \\
&+ \epsilon\beta_7\phi_3(\mathbf{e_z})\phi_4(\mathbf{e}_{\delta_2})
\end{aligned}$$

Condition 11.

$$\begin{aligned}
&\frac{\partial W_f}{\partial t} + \left[\frac{\partial W_f}{\partial\mathbf{e}} - \frac{\partial W_f}{\partial\mathbf{e_z}}\frac{\partial\mathbf{z_r}}{\partial\mathbf{e}}\right]\dot{\mathbf{e}} - \left[\frac{\partial W_f}{\partial\mathbf{e}_{\delta_1}} + \frac{\partial W_f}{\partial\mathbf{e_z}}\frac{\partial\mathbf{z_r}}{\partial\mathbf{e}_{\delta_1}}\right]\dot{\mathbf{e}}_{\delta_1} \\
&- \frac{\partial W_f}{\partial\mathbf{e_z}}\frac{\partial\mathbf{z_r}}{\partial\mathbf{x}_r}\dot{\mathbf{x}}_{\mathbf{r}} - \frac{\partial W_f}{\partial\mathbf{e_z}}\frac{\partial\mathbf{z_r}}{\partial\dot{\mathbf{x}}_{\mathbf{r}}}\ddot{\mathbf{x}}_{\mathbf{r}} \leq \gamma_3\phi_3^2(\mathbf{e_z}) + \beta_8\psi_3(\mathbf{e})\phi_3(\mathbf{e_z}) \\
&+ \beta_9\psi_4(\mathbf{e}_{\delta_1})\phi_3(\mathbf{e_z})
\end{aligned}$$

Condition 12.

$$\begin{aligned}
&\frac{\partial W_f}{\partial\mathbf{e}_{\delta_2}}\left[\frac{\partial\delta_{\mathbf{2r}}}{\partial t} + \frac{\partial\delta_{\mathbf{2r}}}{\partial\mathbf{e}}\dot{\mathbf{e}} + \frac{\partial\delta_{\mathbf{2r}}}{\partial\mathbf{x_r}}\dot{\mathbf{x}}_{\mathbf{r}} + \frac{\partial\delta_{\mathbf{2r}}}{\partial\dot{\mathbf{x}}_{\mathbf{r}}}\ddot{\mathbf{x}}_{\mathbf{r}} + \frac{\partial\delta_{\mathbf{2r}}}{\partial\mathbf{e}_{\delta_1}}\dot{\mathbf{e}}_{\delta_1}\right] \leq \\
&\gamma_4\phi_4^2(\mathbf{e}_{\delta_2}) + \beta_{10}\psi_3(\mathbf{e})\phi_4(\mathbf{e}_{\delta_2}) + \beta_{11}\psi_4(\mathbf{e}_{\delta_1})\phi_4(\mathbf{e}_{\delta_2})
\end{aligned}$$

Condition 13.

$$\begin{aligned}
&\frac{\partial W_f}{\partial\mathbf{e}_{\delta_2}}\frac{\partial\delta_{\mathbf{2r}}}{\partial\mathbf{e_z}}\mathbf{G}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x_r}, \dot{\mathbf{x}}_{\mathbf{r}}, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}, \epsilon) \\
&- \frac{\partial W_f}{\partial\mathbf{e}_{\delta_2}}\frac{\partial\delta_{\mathbf{2r}}}{\partial\mathbf{e_z}}\mathbf{G}(\mathbf{e}, \mathbf{e_z} + \mathbf{z_r}, \mathbf{x_r}, \dot{\mathbf{x}}_{\mathbf{r}}, \mathbf{e}_{\delta_1} + \delta_{\mathbf{1r}}, \mathbf{e}_{\delta_2} + \delta_{\mathbf{2r}}, \mathbf{0}) \\
&\leq \epsilon\gamma_5\phi_4^2(\mathbf{e}_{\delta_2}) + \epsilon\beta_{12}\psi_3(\mathbf{e})\phi_4(\mathbf{e}_{\delta_2}) + \epsilon\beta_{13}\psi_4(\mathbf{e}_{\delta_1})\phi_4(\mathbf{e}_{\delta_2}) \\
&+ \epsilon\beta_{14}\phi_3(\mathbf{e_z})\phi_4(\mathbf{e}_{\delta_2})
\end{aligned}$$

Condition 14.

$$\frac{\partial W_f}{\partial\mathbf{e}_{\delta_2}}\frac{\partial\delta_{\mathbf{2r}}}{\partial\mathbf{e_z}}[\epsilon\dot{\mathbf{z}}_{\mathbf{r}}] \leq \epsilon\beta_{15}\psi_3(\mathbf{e})\phi_4(\mathbf{e}_{\delta_2}) + \epsilon\beta_{16}\psi_4(\mathbf{e}_{\delta_1})\phi_4(\mathbf{e}_{\delta_2})$$

Conditions $9-14$ enforce restrictions upon the difference between the complete system and the reduced subsystems. Use Conditions $1-14$ into (21) and rearrange to get,

$$\dot{\nu} \leq -\Psi^T \mathbb{K} \Psi \tag{24}$$

where $\Psi = \begin{bmatrix} \psi_3 \\ \psi_4 \\ \phi_3 \\ \phi_4 \end{bmatrix}$ and matrix $\mathbb{K}$ given in (24) is positive-definite for $\epsilon < \epsilon^*$. By definition of the continuous scalar functions ψ_3,ψ_4, ϕ_3 and ϕ_4, it follows that $\dot{\nu}$ is negative definite. By Lyapunov theorem it is concluded that $(\mathbf{e}, \delta_{\mathbf{1}}, \mathbf{z}, \delta_{\mathbf{2}}) = (\mathbf{0}, \delta_{\mathbf{1r}}, \mathbf{z_r}(\mathbf{0}, \mathbf{x}_r, \dot{\mathbf{x}}_r), \delta_{\mathbf{2r}})$ is uniformly asymptotic stable equilibrium of the closed-loop system (17). Further, from definition of the tracking error, it is concluded that $\mathbf{x}(t) \to \mathbf{x}_r(t)$ asymptotically. Since the desired trajectory is assumed to be smooth and bounded with bounded first-order derivatives all the other signals remain bounded for all time. ■

IV. NUMERICAL EXAMPLES

A. Purpose and Scope

The purpose of this section is to illustrate the preceding theoretical developments and demonstrate the controller performance for both standard and non-standard forms of singularly perturbed systems. The first example is taken from Reference [2] and the purpose is to see how the proposed approach compares with composite control technique for standard singularly perturbed systems. The objective of the second example is to analyze the performance and robustness characteristics of the controller for non-standard forms of singularly perturbed systems.

Example 1: Standard Singularly Perturbed Model

The following example is taken from Reference [2]. The objective is to design a regulator to stabilize both the slow and the fast state in the domain $D_x \in [-1, 1]$ and $D_z = [-1/2, 1/2]$.

$$\dot{x} = xz^3; \quad \epsilon \dot{z} = z + u \tag{25}$$

The reduced-order models for the system under study are *Reduced Slow Subsystem*

$$\dot{x} = xz^3; \quad 0 = z + u \tag{26}$$

Reduced Fast Subsystem

$$x' = 0; \quad z' = z + u \tag{27}$$

Notice that the algebraic equation in the reduced slow subsystem has an isolated root for the fast state; thus the system given is in standard form.

The controller is designed using the same Lyapunov functions and closed-loop characteristics as in [2]. Using $V(x) = \frac{1}{6}x^6$ as Lyapunov function for the slow subsystem, the desired manifold $z_r = -x^{\frac{4}{3}}$ satisfies Condition 2 with $\alpha_1 = 1$ and $\psi_3(x) = |x|^5$. The control is designed as $u = -3z - 2x^{\frac{4}{3}}$ to satisfy Condition 5 with Lyapunov function $W = \frac{1}{2}(z - z_r)^2$, $\alpha_3 = 2$ and $\phi_3(x, z) = |z - z_r|$. The closed-loop system with $e_z = z - z_r$ becomes

$$\dot{x} = x(e_z + z_r)^3 \tag{28a}$$

$$\epsilon \dot{e}_z = -2e_z + \frac{4}{3}\epsilon x^{\frac{4}{3}}(e_z + z_r)^3 \tag{28b}$$

The inequality in (24) is satisfied for all $\epsilon < 0.4246$.

Notice that the control law designed is exactly same as that obtained using composite control. However, the upper-bound is conservative when compared to the upper-bound obtained using composite control (0.4286). This variation appears because the coefficients of the composite Lyapunov function were chosen as unity in (20) instead of optimal values as in composite control.

Example 2: Non-Standard Singularly Perturbed Model

Consider the following unstable linear system

$$\dot{x} = z - u; \quad \epsilon \dot{z} = x + u \tag{29}$$

The objective is to stabilize the system about $x = 0$ and $z = 0$. Notice that the algebraic equation obtained by setting $\epsilon = 0$ has infinitely many solutions and composite control cannot be applied.

Control Design: With $V(x) = \frac{1}{2}x^2$ as Lyapunov function for the reduced slow subsystem, manifold $z_r = u - \alpha_1 x$ satisfies Condition 2 for $\psi_3(x) = |x|$. Lyapunov function for the fast subsystem is $W(x, z) = \frac{1}{2}(z - z_r)^2$. Condition 5 is satisfied with control $u = -x - \alpha_2(z - z_r)$ and $\phi_3(x, z) = |z - z_r|$. The applied control in original system coordinates is given as

$$u = \frac{-1 - \alpha_1\alpha_2}{1 - \alpha_2}x - \frac{\alpha_2}{1 - \alpha_2}z. \tag{30}$$

Substituting (30) in (29) and with a change of coordinates gives the closed-loop system

$$\dot{x} = -\alpha_1 x + \frac{1}{1 - \alpha_2}e_z \tag{31a}$$

$$\epsilon \dot{e}_z = -\frac{\alpha_2}{1 - \alpha_2}e_z + \epsilon\left[\frac{1 + \alpha_1}{1 - \alpha_2}e_z - \alpha_1(1 + \alpha_1)x\right] \tag{31b}$$

where $e_z = z - z_r$. The constants satisfying Conditions 7-14 are $\beta_2 = \frac{1}{1-\alpha_2}$, $\beta_5 = -\alpha_1(1 + \alpha_1)$ and $\gamma_2 = \frac{1+\alpha_1}{1-\alpha_2}$, rest all being zeros.

Results and Discussion: The closed-loop gains chosen are $\alpha_1 = 0.5$ and $\alpha_2 = 0.5$. With these values, global asymptotic stabilization is satisfied for all $\epsilon < 0.2645$. Table.I documents the closed-loop eigenvalues for different values of ϵ. The closed-loop system loses its time-scale property for $\epsilon > 0.2645$ but remains stable with complex conjugate eigenvalues indicating that the upper bound ϵ^* satisfying condition (24) is conservative. The system becomes unstable for all $\epsilon > 0.4$.

The system given in (29) is the linearized model of the nonlinear non-standard form [11]

$$\dot{x} = \tan z - u; \quad \epsilon \dot{z} = x + u \tag{32}$$

Notice that the fast state appears nonlinearly in the slow dynamics and hence determining a manifold z_r to meet the

$$\mathbb{K} = \begin{bmatrix} \alpha_1 & -\frac{\beta_3}{2} & -\frac{1}{2}\left[\beta_2+\beta_5+\beta_6+\beta_8\right] & -\frac{1}{2}\left[\beta_1-\beta_{10}-\beta_{12}-\beta_{15}\right] \\ -\frac{\beta_3}{2} & \alpha_2-\gamma_1 & -\frac{1}{2}\left[\beta_4+\beta_7+\beta_9\right] & \frac{1}{2}\left[\beta_{11}+\beta_{13}+\beta_{16}\right] \\ -\frac{1}{2}\left[\beta_2+\beta_5+\beta_6+\beta_8\right] & -\frac{1}{2}\left[\beta_4+\beta_7+\beta_9\right] & \frac{\alpha_3}{\epsilon}-\gamma_2-\gamma_3 & \frac{\beta_{14}}{2} \\ -\frac{1}{2}\left[\beta_1-\beta_{10}-\beta_{12}-\beta_{15}\right] & \frac{1}{2}\left[\beta_{11}+\beta_{13}+\beta_{16}\right] & \frac{\beta_{14}}{2} & \frac{\alpha_4}{\epsilon}+\gamma_4+\gamma_5 \end{bmatrix} \quad (24)$$

TABLE I
EXAMPLE 2: CLOSED-LOOP EIGENVALUES

ϵ	Eigenvalues λ
0.05	$\lambda_1 = -0.5914,\ \lambda_2 = -16.9086$
0.1	$\lambda_1 = -0.7396,\ \lambda_2 = -6.7604$
0.2645	$\lambda_{1,2} = -0.6404 \pm 1.167j$
0.35	$\lambda_{1,2} = -0.1786 \pm 1.1818j$
0.4	$\lambda_{1,2} = 0.000 \pm 1.1180j$
0.405	$\lambda_{1,2} = 0.0154 \pm 1.1110j$

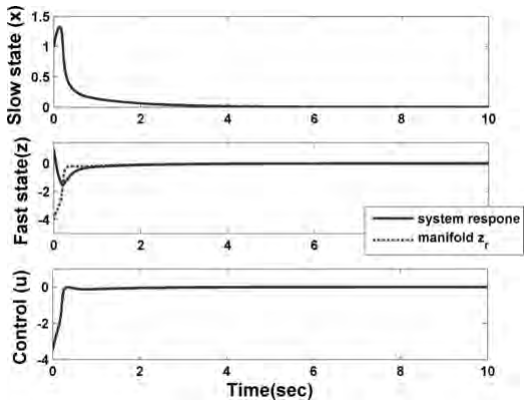


Fig. 1. Example 2: Nonlinear System (32) Closed-loop Response ($\epsilon = 0.1$)

control objective is difficult. Instead, use the controller (30) that was developed for the linear counterpart. The resulting closed-loop system with $\alpha_1 = \alpha_2 = 0.5$ is

$$\dot{x} = 2.5x + \tan z + z; \quad \epsilon\dot{z} = -1.5x - z \qquad (33)$$

The controller converts the non-standard form into standard form which uniquely restricts the system onto the desired manifold, which in this case is $z_r = -1.5x$. It is clear that due to the nonlinear nature of the problem the domain of attraction is now restricted to a subspace of the two-dimensional Euclidean space. Using the previously outlined procedure, constants satisfying Conditions $7-14$ are $\beta_2 = 2$ and $\gamma_2 = 3$ with all others being zero in the domain $D_x \in [0, -1)$ and $D_z \in [-1, 2]$. The upper-bound on singular perturbation parameter is computed as $\epsilon^* = 0.2$. Simulation study in this case indicates that stability is maintained for all $\epsilon < 0.4$ and the nonlinear system is asymptotically stabilized in the domain $D_x \in [-2, 2]$ and $D_z \in [-1.5, 2]$. Simulation results for the case of $\epsilon = 0.1$ are shown in Fig.1. Notice that non-zero control is applied until the fast state falls onto the desired manifold.

V. CONCLUSIONS

In this paper, design procedure for tracking the slow states and stabilization of a general class of nonlinear singularly perturbed systems was developed. Based on the stability proof and simulation results presented in the paper, the following conclusions are drawn. First, the control laws computed for standard singularly perturbed systems using composite control[2] are seen to be a special case of the proposed technique. It was also shown that the upper-bound is conservative with comparison to composite control technique as fixed unity gains were used in the formulation of the composite Lyapunov function. These gains can be chosen optimally as done in composite control technique to provide a less conservative upper-bound. Second, simulations for non-standard singularly perturbed systems shows that for all values of $\epsilon < \epsilon^*$ asymptotic convergence is guaranteed and all closed-loop signals remain bounded. The domain of convergence of the proposed technique was seen to be dependent upon the underlying controllers developed for the reduced-order systems. It is possible to guarantee global results by identifying controllers that satisfy Conditions $1-14$ for the complete space spanned by the system states.

REFERENCES

[1] D. S. Naidu, "Singular perturbations and time scales in control theory and applications: An overview," *Dynamics of Continuous, Discrete, and Impulsive Systems*, vol. 9, pp. 233–278, June 2002.

[2] A. Saberi and H. Khalil, "Stabilization and regulation of nonlinear singularly perturbed systems-composite control," *IEEE Transactions on Automatic Control*, vol. 30, no. 8, pp. 739–747, August 1985.

[3] L. T. Grujic, "On the theory and synthesis of nonlinear non-stationary tracking singularly perturbed systems," *Control Theory and Advanced Technology*, vol. 4, no. 4, pp. 395–409, 1988.

[4] L. Li and F. C. Sun, "An adaptive tracking controller design for nonlinear singularly perturbed systems using fuzzy singularly perturbed model," *IMA Journal of Mathematical Control and Information*, vol. 26, pp. 395–415, 2009.

[5] A. Siddarth and J. Valasek, "Kinetic state tracking of a class of singularly perturbed systems," *Journal of Guidance, Control and Dynamics*, vol. 34, no. 3, pp. 734–749, 2011.

[6] P. Menon, M. E. Badgett, and R. Walker, "Nonlinear flight test trajectory controllers for aircraft," *Journal of Guidance*, vol. 10, pp. 67–72, 1987.

[7] H.-L. Choi and J.-T. Lim, "Gain scheduling control of nonlinear singularly perturbed time-varying systems with derivative information," *International Journal of Systems Science*, vol. 36, no. 6, pp. 357–364, 2005.

[8] N. Fenichel, "Geometric singular perturbation theory for ordinary differential equations," *Journal of Differential Equations*, vol. 31, pp. 53–98, 1979.

[9] J. Cronin and J. Robert E.O'Malley, *Analyzing Multiscale Phenomena Using Singular Perturbation Methods: Proceedings of Symposia in Applied Mathematics*. American Mathematical Society, 1986, vol. 56.

[10] P. Kokotovic, H. K. Khalil, and J. O. Reilly, *Singular Perturbation Methods in Control: Analysis and Design*. Academic Press, 1986.

[11] E. Fridman, "A descriptor system approach to nonlinear singularly perturbed optimal control problem," *Automatica*, vol. 37, no. 4, pp. 543–549, April 2001.

Infotech@Aerospace 2012
19 - 21 June 2012, Garden Grove, California

AIAA 2012-2587

Intelligent Motion Video Guidance for Unmanned Air System Ground Target Surveillance

John Valasek*, Kenton Kirkpatrick†, and James May‡

Advances in unmanned flight have led to the development of Unmanned Air Systems that are capable of carrying state-of-the-art video capturing systems for the intended purpose of surveillance and tracking. These vehicles have the capability to fly through a target area with a mounted camera and allow humans to operate both the UAS and the camera to attempt to survey any objects that are deemed targets. These systems have worked well when controlled by humans, but having them operate autonomously to reduce operator workload and manpower is even more challenging when the camera is fixed to the airframe instead of being mounted on a gimbal, so that the aircraft must be steered in order to steer the camera. The presence of winds must also be accounted for. This paper develops an algorithm for surveillance of ground targets by UAS with fixed pan and tilt cameras, in the presence of winds. This paper develops an algorithm for surveillance of ground targets by UAS. The specific RL algorithm used is Q-learning, and the objective of the approach is to bring any target located in an image captured by a camera into the center of the image using the learned control policy. The learning agent determines offline (initially) how to control the UAS and camera to get a target from any point in the image to the center and hold it there. A feature of this approach is that the learning agent will continue to learn and refine and update the previously offline learned control policy, during actual operation. Results presented in the paper demonstrate that the approach has merit for autonomous surveillance of ground targets.

I. Introduction

One way to introduce the concept of autonomy to the Unmanned Air System (UAS) motion video tracking problem is to determine a control policy that is capable of controlling the UAS autonomously along a certain trajectory, while having the camera controlled by a human. Another way is to do the opposite, and have the UAS flown manually while the camera gimbals to capture and track identified targets. Both of these methods have been explored before and have merit, but having both the UAS and the camera operated autonomously could provide greater flight and tracking efficiency. Having a system that is capable of controlling a UAS and camera system to keep a selected target visible in the camera screen would free the human supervisor to focus on selecting viable targets and analyzing the images received.

The biggest challenge stems from the need to determine an optimal control policy for keeping the target in the middle of the image, using a fixed pan and tilt camera in the presence of winds. Conventional control techniques require determining an appropriate cost function and then finding the weights that make the control optimal. Although finding the optimal control is often straightforward, determining the cost function that best describes the problem is not straightforward. For this research, Reinforcement Learning (RL) is utilized for the determination of the optimal control policy that will both gimbal the camera and steer the UAS to provide target tracking.

Reinforcement Learning (RL) is a subset of machine learning techniques that have been implemented in similar control policy learning scenarios with success. It does not require the declaration of a cost function

*Professor and Director, Vehicle Systems & Control Laboratory, Aerospace Engineering Department. Associate Fellow AIAA, valasek@tamu.edu

†Graduate Research Assistant, Vehicle Systems & Control Laboratory, Aerospace Engineering Department. Student Member AIAA, kentonkirk@gmail.com

‡Graduate Research Assistant, Vehicle Systems & Control Laboratory, Aerospace Engineering Department. Student Member AIAA, jim.may.jr@gmail.com

because it learns the optimal control policy based on physical interaction with the environment rather than mathematical approximations of the problem. RL algorithms break the problem down into a set of states and actions for attempting to reach a goal, and receive rewards from the environment for meeting those goals. Given a particular state of the system, feasible actions are chosen, and the governing control policy is updated based on what kind of reward was given for the results of the action. Over time, the learning agent is able to converge to the optimal control policy for achieving the desired goal from each state.

The specific RL algorithm to be used is Q-learning[1] modified with an Adaptive Action Grid (AAG). The AAG method was developed by Lampton and Valasek[2,3] as a means to provide greater accuracy in reaching the goal state (i.e., the target), while also decreasing the size (dimensions) of the state-space to be considered. This dramatically decreases the total number of states in the system, so that the learning time becomes more feasible and the storage requirements more tractable. Consider a typical optical or infrared (IR) image which contains background features plus a target located in a Region of Interest (ROI). It is assumed here that a human operator would identify the ROI and the target in the image, although the algorithm could conceivably identify the ROI and target by itself, based upon ROI and target characteristics supplied by the human operator. The AAG software agent then discretizes the entire state-space (i.e. image) with course grid spacing, followed by a finer discretization of just the ROI, which is the area of the image in the immediate vicinity of the target. By not discretizing the entire state-space with a fine resolution, there are fewer state-action pairs to both learn and store in memory. However, more precision is needed to reach the goal (target), so multiple levels of finer discretization are used in the ROI as the learning agent gets closer to the goal.

This paper develops an algorithm for surveillance of ground targets by UAS. The specific RL algorithm used is Q-learning, and the objective of the approach is to bring any target located in an image captured by a camera into the center of the image using the learned control policy described above. The learning agent will determine offline (initially) how to control the UAS and camera to get a target from any point in the image to the center and hold it there. A feature of this approach is that the learning agent will continue to learn and refine and update the previously offline learned control policy during actual operation.

II. Algorithm Development

Reinforcement Learning is a process of learning through interaction in which a program uses previous knowledge of the results of its actions in each situation to make an informed decision when it later returns to the same situation. It is a method that has been used for many diverse problems ranging from board games to behavior-based robotics. The purpose of the learning agent used in RL is to maximize the long-term cumulative reward, not just the immediate reward.[5] In this work the goal is to remain in the image, with a preference for being far from the edges. The reward structure must be set up to effect this desire. The agent uses the knowledge gained by reward maximization to update a control policy that is a function of the states and actions. This control policy is essentially a large matrix that is composed of every possible state for the rows, and every possible action for the columns. The three most commonly used classes of RL algorithms are Dynamic Programming, Monte Carlo, and Temporal Difference.[5] The majority of Dynamic Programming methods require an environmental model, making the use of them impractical in problems with complex models. Monte Carlo only allows learning to occur at the end of each episode, causing problems that have long episodes to have a slow learning rate. Temporal Difference methods have the advantage of being able to learn at every time step without requiring the input of an environmental model. The most commonly used method of Temporal Difference is known as Q-learning, with the most common variation being Watkins Q-learning.[6] Q-learning is an on-policy form of Temporal Difference that utilizes an action-value function update rule based on the equation:[5]

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{1}$$

where s_t is the current state, a_t is the current action, s_{t+1} is the next state, a_{t+1} is the next action, Q is the action-value function (used in the control policy), and the k subscript signifies the current policy. The parameter α is a parameter that is used to "penalize" the RL algorithm when it repeats itself within each episode. The parameter γ is the future policy discount factor. α and γ are design parameters that are kept constant for this work. To utilize RL for this problem the proper representation of the environment as the parameters of the Watkins Q-learning algorithm must be made. States, goals, rewards, and actions must be designed. The states of the RL agent, s, are defined for this problem to be those states of the system

that either give information regarding the target's position or those of the UAS that can be controlled for tracking. This yields a state-space consisting of 3 variables: target x-position in the image, target y-position in the image, and UAS bank angle.

$$\underline{s} = \begin{bmatrix} X & Y & \phi \end{bmatrix}^{\mathrm{T}} \tag{2}$$

The goal for an RL agent is defined using the reward structure. The overall goal is to have the target remain in the image frame once commanded, so a proper set of reward requirements must be constructed. Since leaving the image frame is considered the worst result, it retains the worst reward. In this case, a value of $r = -20$ is given for a target leaving the image frame. It is also desired to remain away from the edges of the image, so a reward of $r = -5$ is given for hitting the image boundary, while a positive reward of $r = +20$ is awarded for reaching the center of the image. This encourages the RL agent to move the UAS such that the target stays as far from any edge of the image frame as is possible. All other possible states yield a neutral reward of $r = 0$. Since the goal is generally defined for the RL agent as the state that yields the highest reward, the goal is defined as all states where the target image position is at the center.

$$\underline{g} = \begin{bmatrix} 0 & 0 & \phi \end{bmatrix}^{\mathrm{T}} \tag{3}$$

The agent for this problem is limited in its control of the states because the target global position is independent of any action the UAS takes. The only part of the environment that the UAS can control is itself. Based on the assumptions for this problem, the only way the UAS can control the position of the target in the image frame is to change its bank angle. Therefore, the action-space for this problem is defined to be increments in the UAS bank angle, where for this problem $\Delta\phi = 2$ degrees.

$$\underline{a} = \begin{bmatrix} -2 & 0 & 2 \end{bmatrix}^{\mathrm{T}} \tag{4}$$

This formulation of the RL problem is therefore

$$\underline{s} = \begin{bmatrix} X_i & Y_i & \phi \end{bmatrix}^{\mathrm{T}} \tag{5}$$

$$\underline{a} = \begin{bmatrix} -\Delta\phi & 0\Delta\phi & +\Delta\phi \end{bmatrix}^{\mathrm{T}} \tag{6}$$

$$\underline{g} = \begin{bmatrix} X_{ic} & Y_{ic} & \phi \end{bmatrix}^{\mathrm{T}} \tag{7}$$

III. Simulation Results

III.A. Disturbance Free

All learning takes place offline, and after a sufficient number of learning episodes have been completed the control policy performance and robustness is evaluated via Monte Carlo simulation. Test cases consisting of a fixed target tracking scenario and a moving target tracking scenario are presented. For each case, target position in the image frame and time histories of the UAS states are displayed. Monte Carlo simulations are presented for a chosen timespan of 100 seconds. This represents the typical amount of time for the controller to position the aircraft in a stable tracking configuration. The RL agent was allowed to run for 1,000,000 episodes, with Monte Carlo snapshots taken at a few places beginning at 500,000 episodes. The Monte Carlo randomization places the initial position of the target in one of the four quadrants of the image frame, and at a random position within each quadrant. The controller must then steer the UAS so that ideally, the target is driven to the center of the image frame. One representative case is provided for each of the four quadrants. Images positions are given in pixels and aircraft bank angles are given in degrees. Aircraft inertial positions are in meters.

The results are taken from one single test case of the Monte Carlo runs, in which the target initial conditions place it in the image frame in quadrant 1. Figure 1 shows a 3-dimensional view of the aircraft moving in inertial airspace tracking the target. As can be seen in Figure 1, the aircraft approaches a circular orbit to track the stationary target on the ground. In Figure 2, the position of the target at each timestep

is displayed. The target begins in quadrant 1 of the image and moves toward the goal of the center of the image. However, it is unable to remain there given the aircraft's current state and it is lost from the center. As the aircraft banks left, the target moves up in the image frame, while a right bank moves the target down. The forward motion of the aircraft causes difficulty in tracking the target in the x-direction, and this is reflected in the target almost being lost off the left side of the image. The reinforcement that the agent has received leads it to settle in a state of keeping the target in quadrant 2. Time histories of the target position in the image frame, aircraft bank angle, and commanded change in aircraft bank angle are shown in Figure 3. It is seen that the controller keeps the target in the image frame throughout the simulation timespan.

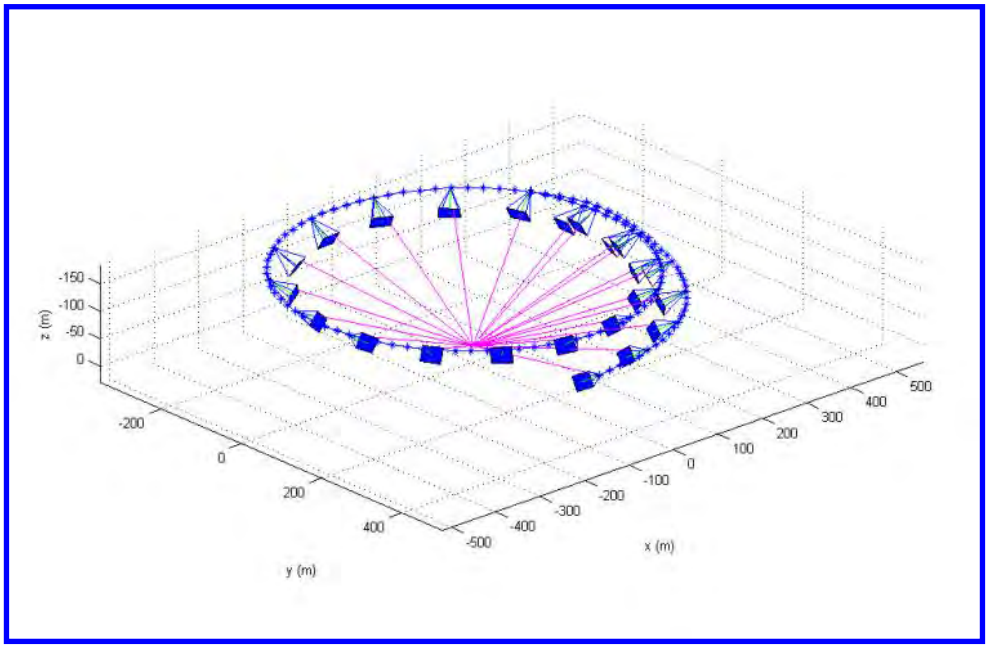


Figure 1. Simulation 3-D View: Stationary Target

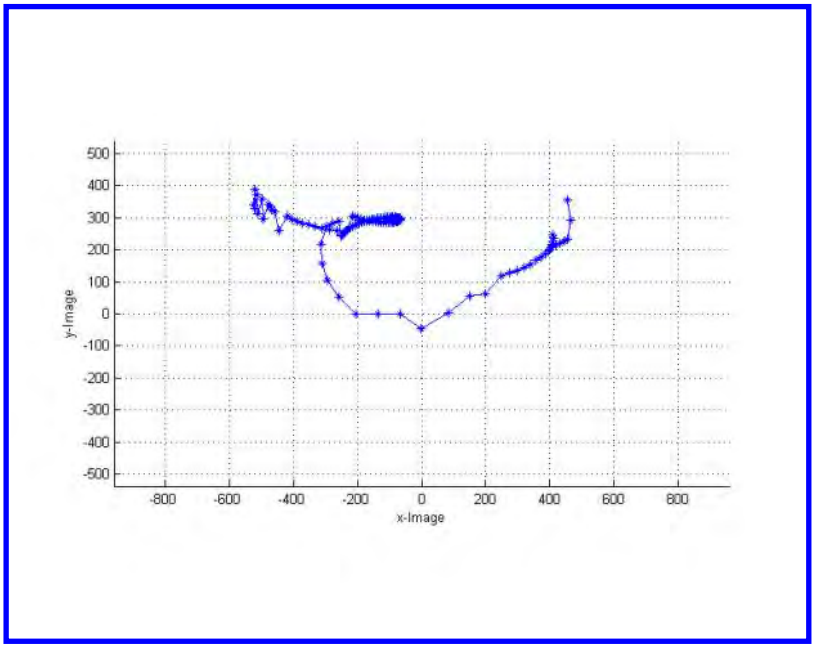


Figure 2. Image Time History: Stationary Target

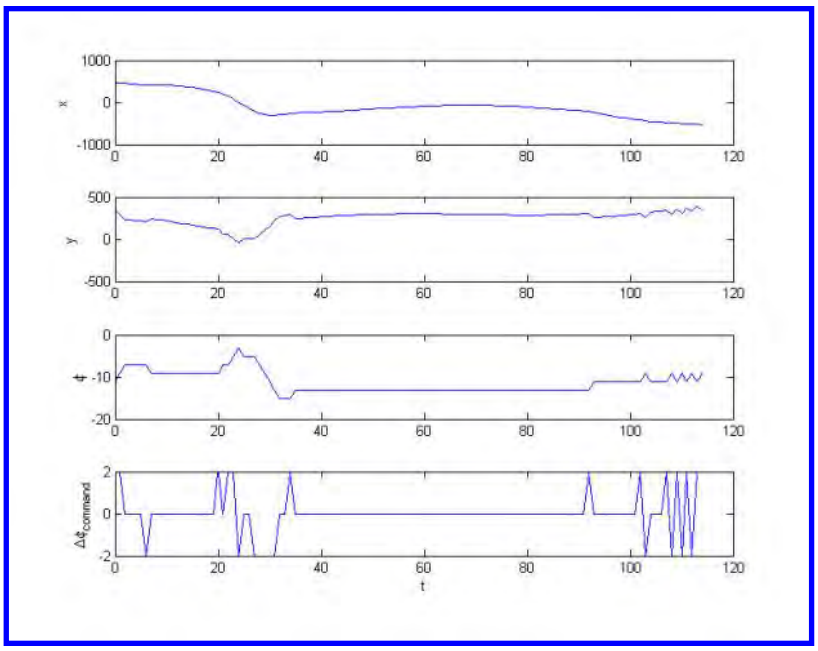


Figure 3. State Time History: Stationary Target

With a successful orbit of a moving target learned, the RL agent was then presented the task of learning to follow a target that moves. This target moves in a straight line at 60 mph, the same speed as the cruise of the aircraft. Under this condition, the aircraft attempts to follow alongside the target as it travels forward. In Figure 4, it can be seen that the aircraft begins following alongside the target well, and begins to deviate away from it as time moves forward. This is due to the stationary camera requiring tracking to be handled by banking the aircraft. It can be seen in Figure 5 that the target is maintained in the image frame throughout the duration of this simulation. Like the stationary case, the target passes the prescribed goal of centering in the image and settles in quadrant 2. The time histories shown in Figure 6 reveal that to maintain this tracking requires frequent bank angle commands, unlike the stationary case.

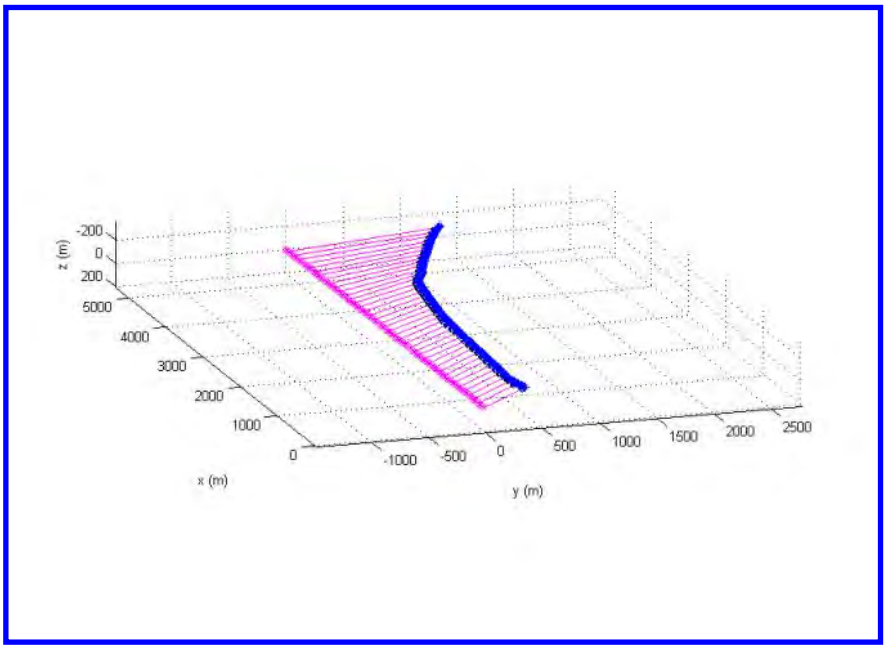


Figure 4. Simulation 3-D View: Moving Target

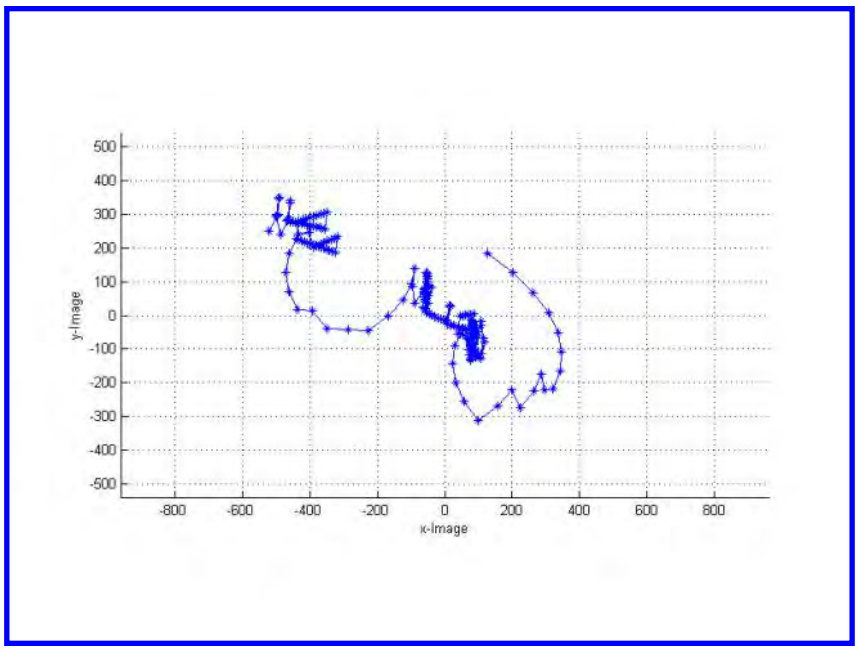


Figure 5. Image Time History: Moving Target

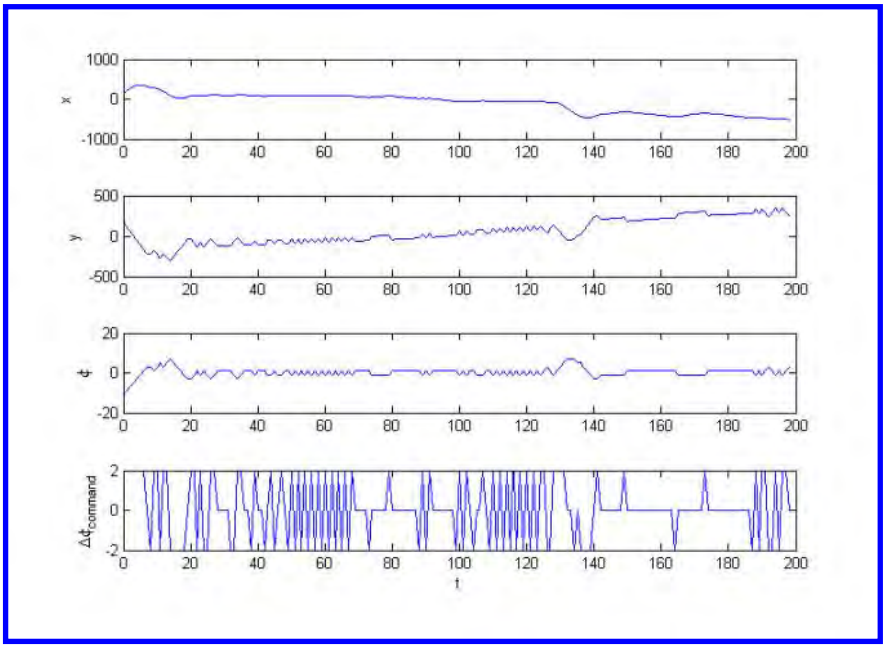


Figure 6. State Time History: Moving Target

III.B. Wind

Accounting for wind in this problem has been done using a variety of methods,[7,8] and accounting for wind disturbances using the present method requires altering the learning process in the state-space to handle the additional state information. Wind can be handled by the learning agent, but it requires knowledge of the wind speed and direction. This modification can be done by adding two new states to the learning state-space. This modified state-space is shown in Equation 8, where v_w and ψ_w are the wind speed and wind heading angle, respectively.

$$\underline{s} = \begin{bmatrix} X & Y & \phi & v_w & \psi_w \end{bmatrix}^{\mathrm{T}} \tag{8}$$

With wind added to the simulation, new learning was experienced with random wind speeds and directions initialized at the beginning of each episode. After 1,000,000 learning episodes, Monte Carlo results were used with the learned Q-matrix. The following figures show an example from these Monte Carlo results for a stationary target with wind disturbances. The wind vector for this particular simulation is 13 mph at a heading angle of 45 degrees. As can be seen in Figure 7, the aircraft approaches the circular orbit as from before, but due to the wind disturbance it is not nearly as smooth of a circular orbit as in Figure 1. Figure 8 shows that throughout the duration of the simulation, the target does remain in the image. By comparing Figure 9 to Figure 3, it can be seen that many more bank angle commands are required to maintain tracking when there is wind, as is expected.

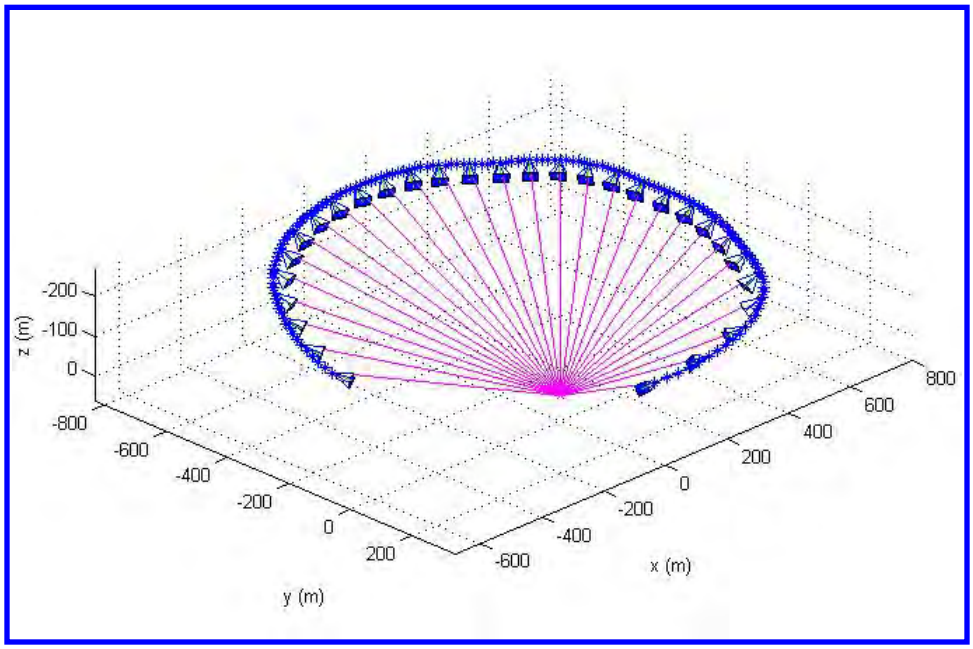


Figure 7. Simulation 3-D View: Stationary Target with Wind

IV. Conclusions and Future Work

Based on the results presented in this paper, it is concluded that:

1. Algorithm learning convergence for the static target case with fixed camera is rapid, due to the low number of state-action pairs. The number of learning episodes required to converge to a "good" solution varies with the particular case/scenario being learned, but all results have shown a clear point of diminishing returns about which running additional learning scenarios provides only a marginal improvement in performance.

2. For all of the stationary target cases evaluated, the RL controller keeps the target in the image frame throughout the simulation timespan. Although the target does not stay in the reinforcement learning positive goal area (the origin), the broader tracking goal of keeping the target in the image frame itself for a useful period of time is generally met.

3. Camera installation and orientation, and the initial position of the target relative to the initial position of the aircraft have a strong influence on the results. In each example, the controller attempts to drive the target into the second quadrant. This is due to the geometry of the scenario and the location of the camera in the aircraft. Having the target in the second quadrant allows the aircraft to turn ahead of

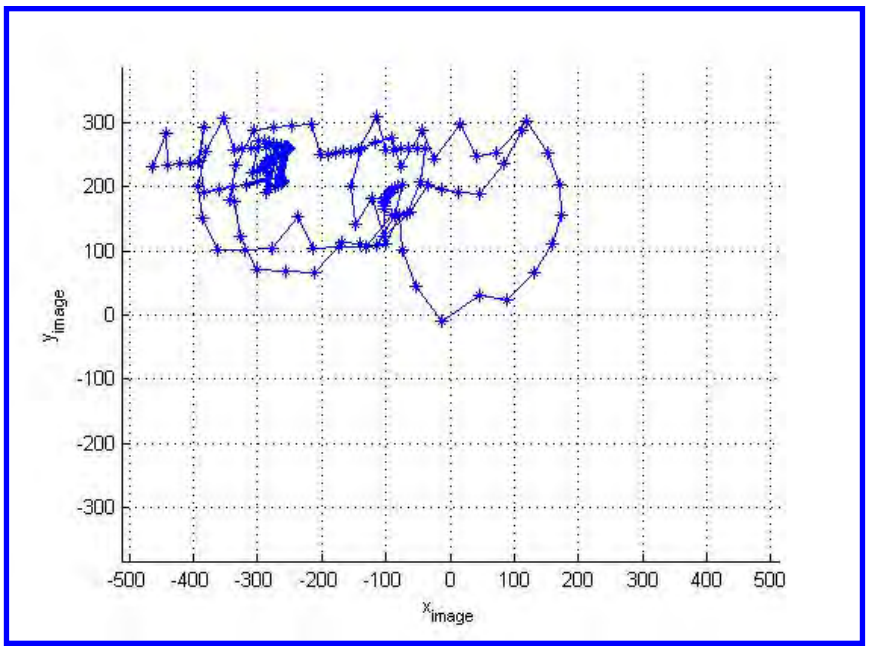


Figure 8. Image Time History: Stationary Target with Wind

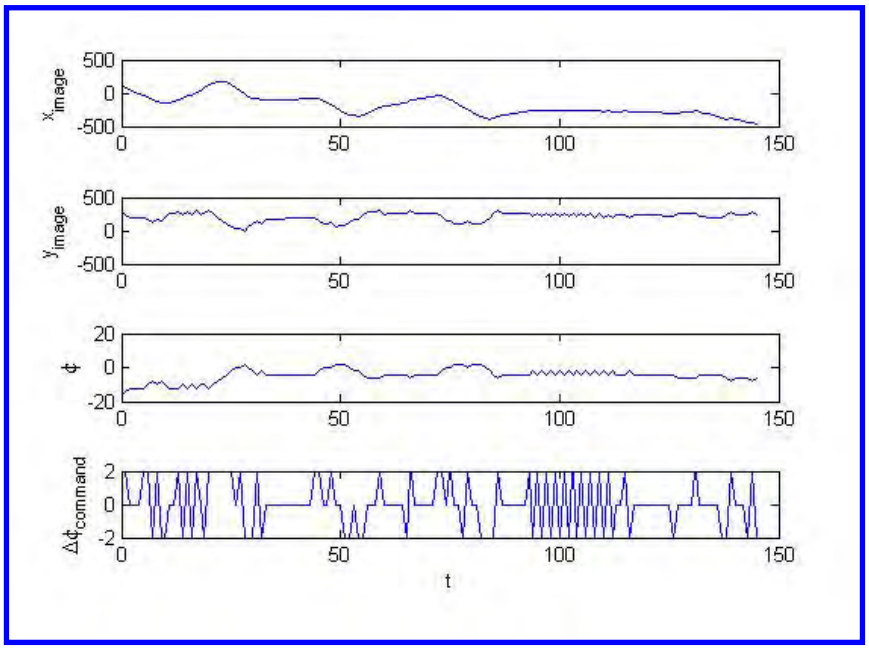


Figure 9. State Time History: Stationary Target with Wind

the target, in order to keep it in the image frame in the future. Consequently, if the camera is oriented to point out the right side of the aircraft, the aircraft is steered to keep the target in the first quadrant.

4. Preliminary results for the case of a moving target show that this method has promise for learning to track a moving target, and merits further investigation.

5. Preliminary results for wind results shows that using wind measurements in the state-space is a promising method for accounting for wind in the learning process, and merits further investigation alongside the moving target scenarios.

This research will be expanded in future work in several ways. One expansion that is to be explored is the use of a gimbaled camera rather than a fixed-base camera mounted on the UAS. This will allow for greater ease of tracking, but will require a reimagining of the Reinforcement Learning problem. Another extension that will be explored is the introduction of more action choices by varying the UAS altitude and/or cruise speed. This will allow for learning to follow a moving target that is traveling at various speeds and along a variety of trajectories, but will greatly increase the learning state-action pairs. For each expansion explored, the inclusion of wind considerations as done in this paper will allow for accurate appraisal of the UAS ability to track ground targets in actuality. With learned control policies for each research scenario, the final research to be conducted in this line will be to evaluate each policy through flight testing on an actual UAS.

Acknowledgment

This research is funded by the Raytheon Company, Intelligence and Information Systems. The Technical Monitor is Mr. Michael R. Moan. This support is gratefully acknowledged by the authors.

References

[1]Bethke, B., Valenti, M., and How, J. "Cooperative vision based estimation and tracking using multiple UAVs." *Lecture Notes in Control and Information Sciences*, 369:179-189, 2007.

[2]Egbert, J., and Beard, R. "Low-altitude road following using strap-down cameras on miniature air vehicles." *Mechatronics*, November 2010.

[3]Redding, J., McLain, T., Beard, R., and Taylor, C. "Vision-based target localization from a fixed-wing miniature air vehicle." In American Control Conference, Minneapolis, MN, 14-16 June 2006.

[4]Nelson, D., Barber, D., McLain, T., and Beard, R. "Vector field path following for miniature air vehicles." *IEEE Transactions on Science*, 23(3):519-529, June 2007.

[5]Sutton, R., and Barto, A. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA, 1998.

[6]Watkins, C. J. C. H., and Dayan, P. "Q-learning." *Machine Learning*, 8:279-292, 1992.

[7]Rysdyk, Rolf, "Unmanned Air Vehicle Path Following For Target Observation in Wind,"' *Journal of Guidance, Control, and Dynamics*, 29(5): 1092-1100, September-October 2006.

[8]Saunders, J, and Beard, R.W.,"Visual Tracking in Wind with Field of View Constraints." *International Journal of Micro Air Vehicles*, 3(2), 2012.

Infotech@Aerospace 2012
19 - 21 June 2012, Garden Grove, California

AIAA 2012-2589

Unmanned Air System Search and Localization Guidance Using Reinforcement Learning

Caroline Dunn*, John Valasek†, and Kenton Kirkpatrick‡
Texas A&M University, College Station, Texas 77843-3141

Requirments for current and future Unmanned Air Vehicles with longer flight endurances have led to a need for an autonomous soaring capability. This paper investigates aircraft flight path guidance for search and localization of Regions of Interest, consisting of atmospheric phenomena. The problem is posed as an offline agent learning problem, of localizing atmospheric thermal locations and then guiding an Unmanned Air Vehicle to soar from one to another. Q- learning is used as the learning algorithm. The computational navigation solution used here is a basic grid algorithm that assigns thermal locations and intensities, with the representation being specified states, actions, goals, and rewards that are used to accomplish the agent learning. The approach is validated with a simulation of a powered Unmanned Air Vehicle. Results presented in the paper show that the autonomous agent can learn how to navigate to a thermal quickly and efficiently by controlling bank angle, while simultaneously monitoring its inertial position and heading angle.

I. Introduction

In this paper, search and localization guidance will initially be applied to a powered UAV but will be eventually used for autonomous soaring of UAVs, a task that can theoretically permit them to stay airborne for many hours or even several days at a time. Thermal updrafts and their intensities can be located and tracked to allow a glider UAV with a nominal endurance of two hours to soar through the air for a maximum of 14 hours. Motion in a fixed-dimension spiral path can be used to explore the atmosphere for thermals, while the aircraft simultaneously keeps track of the updraft's target position.[1] An aircraft can easily make use of a thermal gust's energy to permit higher wing loadings and smaller battery size, which consequently facilitates larger payload and increased cruise speed. Search and localization guidance using thermal updrafts can additionally be accomplished by centering a vehicle about a thermal, or using inter-thermal gusts. Reward signals can be cast in terms of net lift, while the states are a thermal locator's estimate of thermal size and strength.[2] An optimum trajectory generation algorithm that promotes greater autonomy of an aircraft can improve flight range and endurance promoted by a thermal gust.[3]

Thermal updrafts' locations and intensities can be mapped using on-board sensors. Specifically, infrared cameras can be used to locate thermals, and real time trajectories for dynamic soaring applications can be produced.[4] Furthermore, memory components can map available energy based on previous sensor readings and a history of learning.[5] Other research presents maintaining a wind map based on data collected during flight, as well as using currently available maps to generate energy-gain paths. A path generator can then plan paths based on energy efficiency and field exploration.[6]

Once thermal updrafts are located, the amount of vertical velocity and the drifting motion of the center of the thermal can be considered.[7] Thermal centering controllers can be used to represent the thermal's location and size. Maximum climb rate can be achieved at the center of the updraft; bank angle can be changed based on the climb rate to move the aircraft's circular path to coincide with the center of the

*Undergraduate Research Assistant, Vehicle Systems & Control Laboratory, Aerospace Engineering Department. Student Member AIAA, cmdunn@tamu.edu

†Professor and Director, Vehicle Systems & Control Laboratory, Aerospace Engineering Department. Associate Fellow AIAA, *valasek@tamu.edu*

‡Graduate Research Assistant, Vehicle Systems & Control Laboratory, Aerospace Engineering Department. Student Member AIAA, *kentonkirk@gmail.com*

updraft.[8] A UAV can track a moving, non-predefined target, such as a thermal, by changing its turn rate and speed. Tracking a moving thermal can be improved by examining wind disturbance rejection controls.[9]

This paper develops a learning agent for guiding a powered UAV from one thermal location to another by considering the control of the UAV to be a Reinforcement Learning problem. This is done as a precursor to autonomous soaring using unpowered UAVs. To initiate the Reinforcement Learning process, one simply needs to define a set of states and actions that are used in a specific type of Reinforcement Learning algorithm, known as Q-learning. The Q-learning algorithm involves updating a Q-matrix that contains the states, actions, and the value of a certain state-action pair. The states for this problem consist of bank angle, heading angle, and position in the global coordinate system. Upon defining a goal, without supervision the agent determines the thermals' intensities and ranks them in order of highest intensity and smallest distance from the aircraft. If there is only one thermal, that updraft becomes the goal. The aircraft then tracks the thermal and navigates to it using the most efficient route. Rewards are based on the global position of the aircraft and thermals.

II. Q-Learning

Reinforcement Learning is a machine learning technique that uses reinforcing rewards from the environment to improve a policy that determines the best states of a system or the best actions to take given the current state of the system. The most widely used Reinforcement Learning technique is known as Watkins' Q-learning. Q-learning is a temporal-difference method that uses the current estimate of the action-value function to determine how to maximize rewards from the environment in an off-policy manner. It is called off-policy because the policy being used for decision making during an episode is not necessarily the same policy that is being updated with rewards.

While the policy being updated is typically a greedy policy utilizing the action-value function, the policy used during an episode for choosing actions can be exploratory. An agent is given a specific goal, and through interaction with its environment it will either explore to discover better actions for the future, or exploit the knowledge it has already acquired to guarantee an increase in positive reward.[10] An ϵ-greedy policy uses a probability, ϵ, to determine whether an action will be exploration or exploitation.

A learned action-value function, Q, is often referred to as the *Q-matrix*. The Q-matrix contains nearly all possible combinations of state-action pairs once learning is complete. Therefore, the more states or the more actions that are available to an agent, the bigger the Q-matrix becomes. In Q-learning, the final Q-matrix will be optimal if every state-action pair is visited infinitely many times. Since this is not possible, learning is typically ended once the performance is deemed "good enough". As the Q-matrix becomes large, more learning episodes are required for the agent to appropriately experience every state-action pair. Along with specific state-action pairs are corresponding values in the Q-matrix based on the amount and magnitude of rewards given. The values contribute to the policy function. A higher value listed in the Q-matrix for a state-action pair signifies a high benefit of existing in that state and taking that action. However, in order to update or add values to the Q-matrix, an agent needs to explore. If an agent explores, it acts without knowing if it will receive a positive, negative, or zero reward. However, without exploring, an agent could likely never learn the most efficient route to a goal. A balance between exploring and exploiting knowledge is required in the Reinforcement Learning process. It is for this reason that an ϵ-greedy policy is used. The specific update rule that is used for evolving the Q-matrix is shown in Equation 1.

$$Q_{k+1}(s,a) \leftarrow Q_k(s,a) + \alpha[r_{k+1} + \gamma \max_{a'} Q_k(s',a') - Q_k(s,a)] \tag{1}$$

In Equation 1, α is the learning rate, γ is the discount factor, k indicates the current timestep, r is the reward, s is the state, a is the action, s' is the next state, and a' is the next action. Both α and γ are always between 0 and 1. The learning rate establishes the importance of new information compared to previously obtained information. If the value of α is 0, the agent only considers the old information and does not learn anything new; if the value is 1, the agent simply considers the new information without regard to previously gained information. Furthermore, γ measures the importance of knowing the value of future rewards when choosing an action. If γ is near 0, then the agent will primarily consider immediate rewards, while a value approximately equal to 1 will cause the agent to focus mainly on future rewards. This update rule is used in Watkins' Q-learning as shown in the full algorithm below.

Watkins' Q-learning Algorithm:[10]

Initialize $Q(s, a)$ arbitrarily

Repeat (for each episode) :

Initialize s

Repeat (for each step of episode) :

Choose a from s using policy derived from Q(e.g., $\epsilon-$greedy)

Take action a, observe r, $s\prime$

$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a\prime} Q(s\prime, a\prime) - Q(s, a)]$

$s \leftarrow s\prime$;

until s is terminal

III. Representations

For the search and localization guidance task involving the use of Regions of Interest consisting of thermal updrafts, close consideration to states, actions, goals, rewards, and other parameters is important. This particular experiment involves a mathematical simulation model of a UAV moving in two dimensions. States are chosen to be bank angle, heading angle, and X and Y position in the global coordinate system. The three actions that are available to the vehicle include a five degree increase or decrease in bank angle, or no change in bank angle.

For this simulation, the starting state includes arbitrary values of bank angle, heading angle, and X and Y global positions. The available actions are presented in increments of five degrees change in bank angle to permit a continuous and smooth incremental learning process. Four states are included in the method because this is a minimum amount of states needed for this agent to still adequately learn how to reach a goal. If more states were used, there would consequently be more state-action pair combinations. This would require more learning episodes to be completed to allow the agent more opportunities to sample all possible state-action combinations. It is therefore important to keep the amount of states and actions to a minimum.

The goals for this experiment are based on the thermal updrafts. Specifically, they are the magnitudes of the velocity in the center of the thermal. The values of the velocity are assumed to be known prior to the learning. An agent will notice if it has reached a goal if the X and Y coordinates of the state in which it is currently located is within a certain pre-defined range of the X and Y global positions of the center of the thermal. Rewards are subsequently based on the position of the aircraft. A reward of +20 is given if the agent reaches a goal, and a negative reward of the same magnitude is rendered if the agent reaches the boundary of the grid. Any other action that does not immediately place the agent in the goal range or a position on the boundary receives a reward of zero. Magnitudes of the rewards are somewhat subjective. A reward with a very large magnitude can lead to values with large magnitudes in the Q-matrix. However, these values will still principally be proportional to values in the Q-matrix calculated based on rewards of smaller magnitudes. Rewards of less than unity would obviously cause values in the Q-matrix to be unnecessarily minuscule. The magnitude of the rewards has no major effect on the learning process.

IV. Air Vehicle Modeling

The UAV model in the simulations is Pegasus. It has a wingspan of 144 in, and its fuselage is 75 in long and about 15 in wide. Its average cruise speed is in the range of 30-50 m/s. A model of Pegasus is shown in Figure 1. The equations of motion used in the simulation are for constant altitude turning flight.

Figure 1: Photograph of Pegasus UAV

V. Agent Learning

Learning begins with a declaration of a state that the user chooses or is chosen randomly by the computer. An *action matrix* is declared, and includes as inputs the values that can be added to the bank angle. Variables representing the size of the *action matrix*, and the number of states are also created for easy reference throughout the program. The Q-matrix is formed at this time as an empty matrix that can be expanded as values are added to it. It has the same amount of columns as there are actions. In this instance, it begins as three cells because there are three actions available for the agent. Inside each of these three cells is another matrix with number of columns equal to the amount of states plus one additional column to contain the values of the corresponding state-action pair. Because the agent is allowed to explore at the very beginning of the learning process, it chooses any of the three actions available at random. A new function, *nextstate* is called. Its inputs include the state in which the agent existed prior to its move, the action that it chose to take, the *action matrix*, time step, cruise speed, gravitational constant, and the number of states.

Since the bank angle, heading angle, and X and Y position that comprise the new state will exist to infinite decimal places, their values are rounded to the closest four decimal places at the end of the *nextstate* function.

Next, another function is called to determine the reward that the agent deserves after a certain move. As previously mentioned, the agent will receive a positive reward for reaching the goal range, a negative reward for reaching a boundary point, or a reward of zero for any other move that does not immediately lead the agent to the goal. In order to prevent the agent from more easily maximizing its total reward by changing its bank and heading angle as opposed to its position, the reward function only gives rewards based on X and Y position. The Q-matrix is updated in a proceeding function. The reward value is placed in a cell of the Q-matrix based on the action taken and the state that the agent was in before the action. The agent's current state now becomes the new starting state and a new cycle begins. When the agent reaches the predefined goal, the episode ends and a new episode initiates.

At this point, an agent can either take actions randomly or *greedily*. A greedy action involves an agent choosing an action that it already knows will lead to a high reward. Before any action is chosen, however, the *egreedy* function is called. It takes as inputs the current state, the Q-matrix, the number of actions, and the current value of ϵ. The function outputs the action-value of the current state and selected action, as well as the best action that the agent should take if it is going to exploit its knowledge. Otherwise, the *egreedy* function will choose a random action for the agent to take.

As new states are being determined based on the chosen action, the heading angle, bank angle, and X and Y position values will not be whole numbers. Instead of rounding the X and Y coordinates and heading angle immediately in the learning process, these values remain continuous during the process of calculating the X and Y coordinates and bank and heading angle for the next state. However, at the time that the values in the Q-matrix need to be updated, the X and Y coordinates and heading angle are rounded according to the discretization matrix. This allows exact states in the Q-matrix to be found and updated. Heading and bank angle values are rounded to the closest number evenly divisible by five. On the other hand, X and Y coordinate position values are rounded to numbers evenly divisible by 50. This rounding is important because an agent is required to move from one grid point to another. The grid is separated in increments of 50 m in both the abscissa and ordinate axis. At this point, the *k-nearest neighbor* algorithm is used to transfer the agent to the closest point on the grid. The next state is then calculated using the actual values of X and Y coordinates and heading angle. Importantly, the bank angle is kept as a value divisible by 5 during

the learning process and the Q-matrix update process. Consequently, the agent behaves more realistically; the X and Y coordinates and heading angle are rounded only in the process of updating the Q-matrix and determining the next best action.

An important step in this process is saving the Q-matrix at various time increments. This enables additional learning for the agent that can ensure a more precise Q-matrix. In this example, the Q-matrix is saved every 200 episodes. Furthermore, the code allows the user to begin the Q-learning process and terminate it whenever is necessary. Later, the user can choose to continue the learning where it previously left off and upload the already partially learned Q-matrix. Consequently, the previously learned Q-matrix can be improved. As each new simulation is completed, files named according to the current date are saved with the values of all parameters and of course the Q-matrix. This organizes each process and expedites retrieval of previous files when needed.

The Monte Carlo method is finally used to determine the amount of times that the agent reached the goal. The Monte Carlo process completed here also noted the path of the agent, the action it took at each state, and the amount of positive and negative rewards given. These values are stored in separate matrices. Based on this data, one can determine the effectiveness of the learning process. Additionally, because the Q-learning process ends as soon as the agent discovers the goal, it is necessary to end the cycle in the Monte Carlo episodes once the agent found the goal. In reality, an aircraft does not stop moving once it reaches a goal. However, to ensure the proper performance of the learning process, it is sufficient to code a break at the time that the agent reaches the goal.

VI. Simulation

The objective of the simulation described in this paper is to show how Q-learning affects the way an agent navigates through its environment. It is desirable that this learning method be precise and time efficient. The mathematical simulation of the UAV was executed using the commercial Matlab software program.

The simulation involves an agent beginning the Q-learning process along the boundary of the grid. Initial X and Y boundary coordinate positions are random, but the large magnitude of episodes required promotes initial experience at every boundary point. The agent starts at every grid point on the boundary including the corners of the grid. Furthermore, initial heading and bank angle of the aircraft is properly defined for this simulation. Specifically, if the agent begins at a boundary point that is along the X axis, its initial state consists a 0 degree heading angle. If the agent begins along the Y axis, its initial heading angle is 90 degrees. For every starting boundary condition, the agent begins with a 0 degree bank angle and a heading angle that allows the agent to be pointing directly to the center of the grid. Consequently, an agent starting at a corner grid point possesses a 45, -45, 135, or -135 degree heading angle depending on the location of the corner. These values are determined based on the parameter constraints chosen by the user.

When the action taken to affect the bank angle is known, the subsequent state can be determined using Equations 2, 3, 4, and 5 relating aircraft dynamics:

$$\dot{\psi} = \frac{\phi g}{u} \tag{2}$$

$$\Delta\psi = (timestep)\dot{\psi} \tag{3}$$

$$dx = (timestep)u\cos(\Delta\psi) \tag{4}$$

$$dy = (timestep)u\sin(\Delta\psi), \tag{5}$$

where ψ is the heading angle, ϕ is the bank angle, g is the gravitational constant, and u is the vehicle's cruise speed. These equations are used because only constant cruise speed and constant radius turns are considered. The agent is assumed to move at a constant cruise speed of 35 m/s, the time step is established as 2 s, and the acceleration of gravity used is 9.8 m/s^2.

Before the Q-learning process begins, a thermal location and intensity are randomly generated using a script that involves running a check case of a NASA DFRC updraft model. It is displayed on a grid that is a 1000 m square in the X and Y coordinate system. It should be noted that these dimensions can be altered to allow for a greater learning space. However, for initial learning, it is significant to begin the process on a grid this size. The global position of the maximum intensity in the center of the updraft is known and presented in the updraft model, and the velocity at this coordinate is stored as the goal for the learning process. The user declares a range of velocities that acts as an invisible radius around the thermal. The

agent will attempt to at least reach the thermal somewhere in its range to receive a positive reward. In this case, the maximum of the range is 4 m/s and the minimum is slightly greater than 0 m/s. Specifically, the minimum range value will always be 95 percent of the maximum intensity velocity. This ensures that positive reward values are not given to the agent when it is in an area with a zero velocity thermal gust.

The program is initially set to run about 150,000 episodes. Within each episode, the agent is allowed 500 cycles, or opportunities, to take an action. The parameters are simply a basic starting point to observe the learning process. Not all cycles will always be used within each episode however. If the agent reaches a boundary point, for example, the cycle will end and a new episode will begin. Other constants are also initialized before the learning occurs. The learning rate is set to 0.1 and the discount factor takes a value of 0.7. The value of ϵ, the parameter utilized in the ϵ-greedy method that promotes random action, is initially unity. This means that the agent will begin the process solely exploring and choosing actions at random. After every 5 percent of episodes are completed, ϵ will decrease by 0.05 to ensure exploitation. However, this value is not permitted to reach a number less than 0.05 because it is not necessary to completely ignore the exploring process.

Certain constraints are required for several parameters. For the two-dimensional learning process described in this paper, both X and Y coordinates are restricted between 0 and 1000 m to prevent the aircraft to enter negative coordinate positions outside of the grid. The heading angle is permitted to exist between -180 and 180 degrees, while bank angle is required to be between -45 and 45 degrees. These values can potentially be altered, but they are reasonable and effective for this specific simulation.

VII. Numerical Examples

The first simulation completed 150,000 episodes. One thermal was placed at the coordinates (732.0689, 702.2730) which rounds to the grid point (750, 700). One trajectory chosen by the agent is shown below in Figure 2. The agent began learning at the grid point (0,350) with a heading angle of 90 degrees and a bank angle of 0 degrees. Figure 3 shows the agent's time history. The top two graphs present time versus coordinate position. The dotted red line represents the respective coordinate of the thermal and the dotted black lines above and below represent the range of the thermal. The bottom two graphs present the relationship between time and change in bank and heading angles for this specific trajectory.

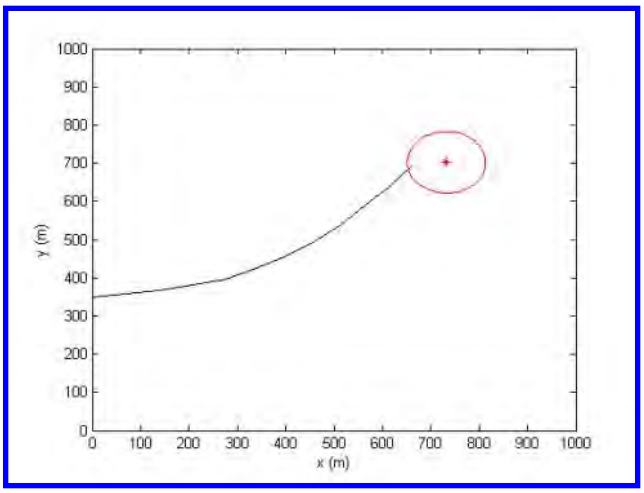


Figure 2: Trajectory of the UAV

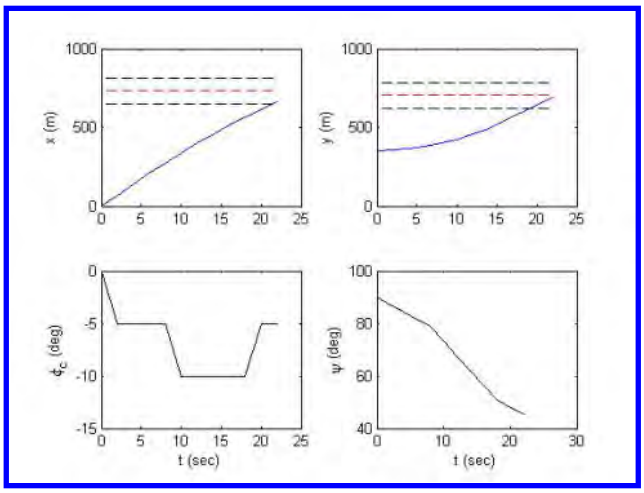


Figure 3: Time History of the Simulation

The next simulation completed approximately 130,000 episodes. Three thermals were placed at the coordinates (181.0450, 815.9635), (384.2516, 269.0323), and (595.0047, 193.0734) which round respectively to the grid points (200, 800), (400, 250), and (600, 200). Each thermal had a different range of intensity. This simulation involved requiring the agent to travel to the closest most intense thermal. In the learning process, reward-shaping was created based on updraft velocity. When the locations and intensities of the three updrafts were known, the distance from the starting point of the agent to the center of each thermal was calculated using a simple distance formula. For this simulation, velocity was given twice as much significance as distance. Therefore, if the agent traveled to the thermal of highest velocity, it would receive a reward

twice as great as traveling to the thermal with the shortest distance from the agent's initial state. In the code, these metrics relating the rewards based on velocity and distance are represented as variables and can be altered if necessary. A single trajectory of this simulation is shown in Figure 4. The agent began learning at the grid point (1000, 950) with a heading angle of -90 degrees and a bank angle of 0 degrees. Through learning, the agent appropriately chose to travel to the thermal at the location (181.0450,815.9635). Figure 5 shows the agent's time history for this specific trajectory.

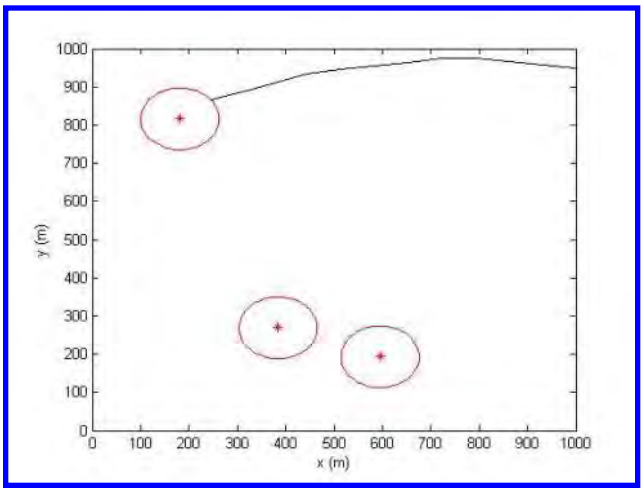


Figure 4: Trajectory of the UAV

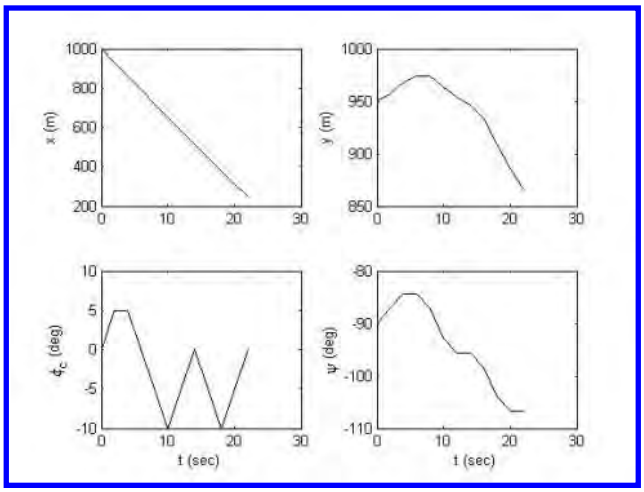


Figure 5: Time History of the Simulation

VIII. Conclusions and Future Work

Results presented in the paper demonstrate the potential for Reinforcement Learning, specifically Q-learning, to guide an Unmanned Air System to a specified Region of Interest from a specified starting state. There are several conclusions that can be drawn from these results:

1. The Q-learning technique is effective at navigating a UAV to fixed thermal locations. In the first simulation, the agent reached the goal about 85 percent of the time; in the second simulation, the agent reached the goal approximately 99 percent of the time.

2. As more learning episodes are completed, the navigation ability of the UAV improves. In the first simulation, for example, 100,000 episodes were completed and a Monte Carlo test showed that the agent reached the goal 81 percent of the time. After 50,000 more episodes were completed based on learning previously obtained, the first simulation generated success 84 percent of the time. This relation shows that the success for the first simulation increased about 3 percent after completing 50,000 more learning episodes.

3. Adding additional thermals increases the computational complexity linearly. When comparing the two simulations, the second simulation with three goals was approximately three times more computationally expensive than the first simulation.

4. Adding additional thermals to the navigation area increases the probability that the UAV will find a thermal updraft.

Future work will extend the approach to the autonomous soaring problem. This will be done by making the learning more precise by refining the grid spacing using the Adaptive Action Grid Technique.[11] Additionally, an agent will learn how to maneuver in a three dimensional space. This will involve changes in altitude and will require an update and expansion of the state and action matrices. Eventually, this machine learning process will be efficient and applicable to the autonomous soaring of a real sailplane.

Acknowledgments

This material is based upon work supported in part by the U.S. Air Force Office of Scientific Research under contract FA9550-08-1-0038, with technical monitor Dr.Fariba Fahroo. It is also supported by Texas A&M University from the Undergraduate Summer Research Grant program. This support is gratefully acknowledged by the authors. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Air Force.

References

[1]Allen, M., "Autonomous Soaring for Improved Endurance of a Small Uninhabited Air Vehicle," *43rd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, 10-13 January 2005.

[2]Wharington, J. and Herszberg, I., "Control of a High Endurance Unmanned Air Vehicle," *21st Congress of International Council of the Aeronautical Sciences*, Melbourne, Australia, 13-18 September 1998.

[3]Kahveci, N. E., Ioannou, P. A., and Mirmirani, M. D., "Adaptive LQ Control With Anti-Windup Augmentation to Optimize UAV Performance in Autonomous Soaring Applications," *IEEE Transactions on Control Systems Technology*, Vol. 16, 2008, pp. 691–707.

[4]Akhtar, N., *Control System Development for Autonomous Soaring*, Ph.D. thesis, Cranfield University, January 2010.

[5]G. C. Bower, T. C. F. and Naiman, A. D., "Dynamic Environment Mapping for Autonomous Thermal Soaring," *AIAA Guidance, Navigation, and Control Conference*, Toronto, Canada, 2-5 August 2010.

[6]Lawrance, N. R. J. and Sukkarieh, S., "Autonomous Exploration of a Wind Field with a Gliding Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 34, 2011, pp. 719–733.

[7]C. Antal, O. G. and Levi, S., "Adaptive Autonomous Soaring of Multiple UAVs Using Simultaneous Perturbation Stochastic Approximation," *49th IEEE Conference on Decision and Control*, Atlanta, Georgia, 15-17 December 2010.

[8]K. Andersson, I. Kaminer, V. D. and Cichella, V., "Thermal Centering Control for Autonomous Soaring: Stability Analysis and Flight Test Results," *Journal of Guidance, Control, and Dynamics*, Vol. 35, 2012, pp. 963–975.

[9]Vaughn, A. C., *Path Planning and Control of Unmanned Aerial Vehicles in the Presence of Wind*, Ph.D. thesis, University of California, Berkeley, 2002.

[10]Sutton, R. and Barto, A., *Reinforcement Learning: An Introduction*, The MIT Press, Cambridge, MA, 1998.

[11]Lampton, A. K., *Discretization and Approximation Methods for Reinforcement Learning of Highly Reconfigurable Systems*, Ph.D. thesis, Texas A&M University, 2009.

AIAA Guidance, Navigation, and Control Conference
13 - 16 August 2012, Minneapolis, Minnesota

AIAA 2012-4453

Tracking Control for a Non-Minimum Phase Autonomous Helicopter

Anshu Siddarth* and John Valasek†

Texas A & M University, College Station, TX 77843-3141.

This paper develops a general control algorithm for precision output tracking of nonlinear non-minimum phase dynamics of an autonomous three degree-of-freedom unmanned helicopter. Previous approaches in flight control literature have shown approximate tracking by neglecting the coupling between the forces and moments generated by the control effectors. However, it is shown that this coupling is significant in the model under study and cannot be neglected. In this paper the coupling is retained and natural time-scale decomposition of the vehicle model is employed for accomplishing asymptotic tracking. The design procedure determines the desired internal state trajectory and the control scheme to stabilize the helicopter in hover. Stability is analyzed using Lyapunov methods and results show that the approach is able to accomplish perfect tracking while stabilizing the closed-loop system and keeping all closed-loop signals bounded.

Nomenclature

$\mathbf{f}, \mathbf{g}, \mathbf{h}$	sufficiently smooth vector fields
$\mathbf{s}$	intermediate variables of the system
$\mathbf{u}$	control input vector of the system
$\mathbf{x}$	slow variables of the system
$\mathbf{z}$	fast variables of the system
a_{1s}	longitudinal tilt of the tip path plane of the main rotor with respect to the shaft, rad
F_x	body force in the forward direction, N
F_z	body force in the vertical direction, N
g	acceleration due to gravity, m/sec^2
h_M	distance between c.g and main rotor positive in the upward direction, m
I_y	moment of inertia about pitch-axis, kgm^2
l_M	distance between the c.g and main rotor along forward direction, m
M	pitching moment acting on the helicopter, Nm
m	mass of the vehicle, kg
q	body pitch-rate, rad/sec
Q_T	tail rotor torque, Nm
t	slow time-scale, seconds
T_M	thrust of the main rotor, N
T_T	thrust of the tail rotor, N
u	body forward velocity, m/sec
w	body vertical velocity, m/sec(positive down)
X, Y, Z	North-East-Down helicopter frame
X_f, Z_f	body forces positive along north and down respectively
X_{app}, Z_{app}	approximate body forces acting along north and down respectively
x	inertial position, positive pointing north, m

*Graduate Research Assistant, Vehicle Systems & Controls Laboratory, Department of Aerospace Engineering, Student Member AIAA, *anshun1@tamu.edu*

†Professor and Director, Vehicle Systems & Controls Laboratory, Department of Aerospace Engineering, Associate Fellow AIAA, *valasek@tamu.edu, http://vscl.tamu.edu/valasek* (Corresponding Author)

z inertial position, positive down, m

Subscripts

$*$ trim quantity
0 reference quantity
a derivative with respect to angle a_{1s}
d desired manifold
r reference trajectory

Symbols

$\alpha, \alpha_1, \beta, \beta_1, K_\theta, K_q$ feedback gains
ϵ, ϵ_1 small quantity, singular perturbation parameter
γ sum of longitudinal tilt and desired pitch-attitude manifold $a_{1s} + \theta_d$
τ fast time-scale $\frac{t-t_0}{\epsilon}$, seconds
θ body pitch-attitude angle, rad
$O()$ order symbol
β_i $i > 1$, Lipschitz constants

Superscripts

$\breve{}$ derivative with respect to intermediate time-scale $\frac{t-t_0}{\epsilon_1}$
$\hat{}$ dimensionless quantity
$\tilde{}$ error between the system state and desired trajectory

I. Introduction

Several important flight control problems are characterized by unstable internal dynamics consequently resulting in performance limitations.[1] Some common studies include acceleration control of tail-controlled missiles,[2] control of planar Vertical Take-off and Landing (VTOL) aircraft[3] and Conventional Take-off and Landing (CTOL) aircraft.[4] This paper examines the internal dynamics and synthesizes a stabilizing controller for a three degree-of-freedom longitudinal dynamics of an autonomous helicopter.

Hover control of a helicopter is one of the most challenging non-minimum phase control problems. To qualitatively analyze this behaviour consider the helicopter shown in Figure 1. The motion of the helicopter is described in North-East-Down frame shown as (X, Y, Z) in the figure. Assume that the helicopter model is allowed to pitch only about the Y axis. T_M and T_T are the thrusts generated by the main and the tail rotor respectively that keep the vehicle aloft. The angle a_{1s} is the longitudinal tilt the tip path plane makes with respect to the shaft of the main rotor. Side view of Figure 1 shows that non-zero tilt induces a component of the main rotor thrust along the horizontal X axis and consequently the helicopter propels forward. Hence, in order to remain in hover the main rotor thrust and the angle a_{1s} need to be controlled. However, changing this angle has another consequence. The forward component of the thrust that it creates induces a clockwise pitching moment about the center of gravity of the vehicle causing the nose to drop. In order to remain level, the angle a_{1s} needs to be corrected. But doing so alters the forces acting on the helicopter and the vehicle departs from hover. For the helicopter under study, it will be shown in Section II that desired T_M and a_{1s} required to maintain hover lead to unstable oscillatory pitching motion.

Previous studies for hover control assume that the dynamical behaviour of a helicopter is similar to that of a VTOL aircraft as both these vehicles have direct control over the aerodynamic lift. Hence several studies employ the control developments proposed for VTOL aircraft.[3] Formulation in [3] assumes that the force contribution from the longitudinal tilt angle a_{1s} is negligible. Such a simplification removes the coupling between the forces and the pitching moment and makes the resultant dynamical model; approximately input-output linearizable. Reference [5] used feedback linearization for stabilizing the resulting approximate model in order to guarantee bounded transient errors. More recently back-stepping has been used for control of small autonomous helicopters.[6,7,8] Other control techniques based upon the approximate model include dynamic-inversion[9] and neural-network based adaptation.[10] In order to mitigate the limitations due to under-actuation some techniques take advantage of the inherent multiple time-scale behaviour of helicopters. Reference [11] compared linear and nonlinear control designs for the approximate model using the fast rotational dynamics as virtual control variables. A similar approach was proposed in [12] wherein Lyapunov based methods were used to guarantee stability of a radio/control helicopter model using the approximate dynamics.

As a consequence of neglecting the coupling between the forces and the moments, application of aforementioned methods is limited in operating regime and to reference commands that do not require to be precisely followed. Exact output tracking was demonstrated by retaining the coupling terms in [13] through stable-inversion of a linear helicopter model. This inversion computed the desired input-state trajectory that along with feedforward and feedback control led to asymptotic output tracking. Approach in [13] emphasized that internal-state feedback is necessary to stabilize a non-minimum phase system. However, the method required an infinite time preview and knowledge of the complete output trajectory beforehand.

From the above discussion it is understood that helicopter control design poses three major challenges. First, the coupling between forces and moments generated due to rotor is significant and must not be ignored during control design.[14] But retaining this coupling makes the system non-minimum phase and difficult to stabilize. Second, a non-minimum phase system cannot be asymptotically stabilized in real-time with available control techniques and control design requires substantial offline processing. Third, current real-time implementable approaches that are independent of the reference trajectory are limited in performance and operating regime.

This paper presents a control design procedure that addresses the above technical challenges and validates the general nonlinear control procedure developed by the authors in [15], [16] for a three-dimensional longitudinal model of an autonomous helicopter. The paper makes three major contributions. First, the control design takes advantage of the natural time-scale separation and unlike the techniques discussed in [11], [12] the coupling between the forces and the moments of the helicopter model is retained. It is shown that this coupling allows design of a sequential procedure for computing the desired internal states that ensure asymptotic output tracking. Second, the full-state feedback controller designed is real-time implementable and is independent of any particular operating condition and desired output trajectory. Third, the controller designed is causal and does not require any knowledge or preview of the output trajectory beforehand.

The paper is organized as follows. Section II describes the helicopter model under study and examines analytically the non-minimum phase properties of the vehicle. The nonlinear control design and stability of the closed-loop system is analyzed in Section III. Simulation validation for hover control is discussed in Section IV. Finally, conclusions are presented in Section V.

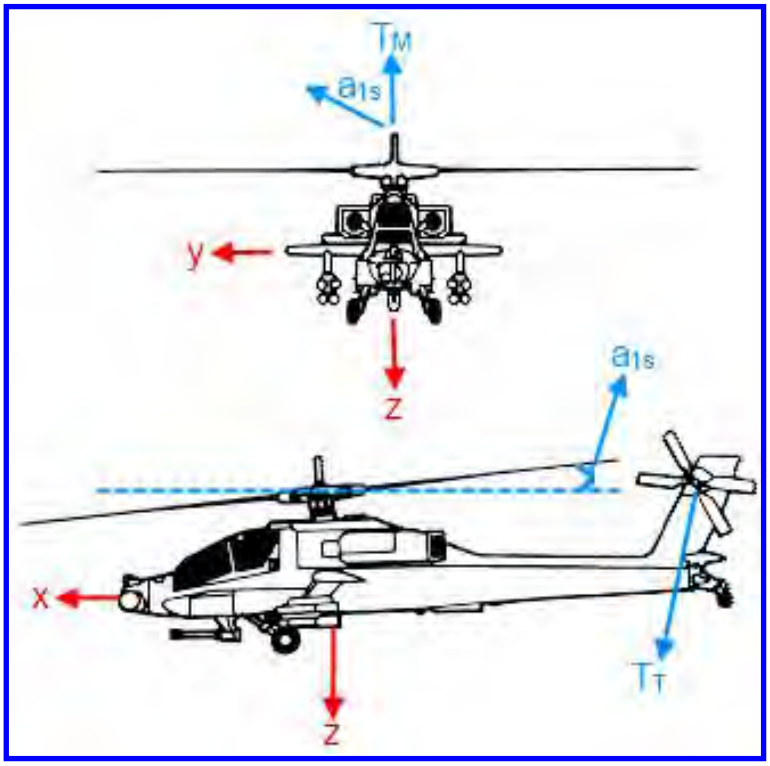


Figure 1. Coordinate frame and forces acting on a helicopter

II. Model Description & Open-Loop Analysis

In this section the governing equations of the helicopter model are presented. Then, the exact input-output linearization of the model is carried out and it is shown that the system has oscillatory internal dynamics. The effect of neglecting the coupling between the forces and moments is also discussed. Finally, a time-scale analysis of the model under study is carried out and essential concepts of singular perturbation theory recalled.

A. Vehicle Description

The helicopter model is written with respect to earth-fixed inertial coordinates. The forces and the moments act in the body frame (see Figure 1). The origin of the body fixed frame is the center of gravity of the platform and it is assumed that this moves with the motion of the fuselage. Reference is made to the nomenclature for the meaning of the symbols. The three degree-of-freedom equations of motion of a symmetric helicopter model in hover (assuming the lateral/directional components are in equilibrium) are as follows

$$\begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} u \\ w \end{bmatrix} \tag{1a}$$

$$\begin{bmatrix} m\dot{u} \\ m\dot{w} \end{bmatrix} = \begin{bmatrix} -qw + F_x \\ qu + F_z \end{bmatrix} + \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} 0 \\ mg \end{bmatrix} \tag{1b}$$

$$\dot{\theta} = q \tag{1c}$$

$$I_y \dot{q} = M \tag{1d}$$

From a rigorous standpoint, the above set should be augmented with dynamic equations of longitudinal flapping. However, it is assumed that the time-constant for the flapping of conventional rotor blades corresponds to one-quarter of a rotor revolution[17][pp 558-559]. This justifies the use of rigid-body equations for describing the motion.

Table 1. Helicopter Model Parameters

Parameter	Value
m	$4.9kg$
I_y	$0.271256kgm^2$
h_M	$0.2943m$
l_M	$-0.015m$
Q_T	$0.0110Nm$
M_a	$25.23Nm/rad$

The body forces (F_x, F_z) and pitching moment M are generated by the main rotor and controlled by T_M, main rotor thrust and a_{1s} longitudinal tilt of the tip path plane of the main rotor with respect to the shaft. The aerodynamic model given below is taken from [5].

$$F_x = -T_M \sin a_{1s} \tag{2a}$$

$$F_z = -T_M \cos a_{1s} \tag{2b}$$

$$M = M_a a_{1s} - F_x h_M + F_z l_M - Q_T \tag{2c}$$

with the system parameters given in Table 1.

B. Exact & Approximate Input-Output Linearization

The non-minimum phase properties of the model under consideration are analyzed by studying the input-output relationship. The desired outputs for the control design are the inertial coordinates of the vehicle,

namely (x, z) pointing north and down respectively. Control inputs available are the main rotor thrust T_M and longitudinal tilt a_{1s}. Taking second derivative of each output,

$$\begin{bmatrix} \ddot{x} \\ \ddot{z} \end{bmatrix} = \frac{1}{m} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} F_x \\ F_z \end{bmatrix} + \begin{bmatrix} 0 \\ g \end{bmatrix} \tag{3}$$

it is found that the relative degree of each output is two. This implies that the rotational dynamics given in (1c),(1d) constitute the internal dynamics of the system.

In order to analyze the internal stability of the system, the zero dynamics of the system needs to be examined. Toward this end, the control vector (T_M, a_{1s}) that constraints the outputs and its derivatives on the origin is computed. From (3) and the aerodynamic relations given in (2) the following solution is determined.

$$\begin{bmatrix} T_M \\ a_{1s} \end{bmatrix} = \begin{bmatrix} mg \\ -\theta \end{bmatrix} \tag{4}$$

Using the moment relation given in (2c) and the constrained control solution (4) the zero dynamics are characterized by the following equations

$$\dot{\theta} = q \tag{5a}$$

$$\dot{q} = \frac{1}{I_y} \left[-M_a\theta - mg(h_M \sin\theta - l_M \cos\theta) - Q_T \right] \tag{5b}$$

The stability of the above system is analyzed by linearizing about the trim values $\theta_* = 0.018rad$ and $q_* = 0rad/sec$.

$$\begin{bmatrix} \Delta\dot{\theta} \\ \Delta\dot{q} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{1}{I_y}(-M_a - mgh_M \cos\theta_* + mgl_M \sin\theta_*) & 0 \end{bmatrix} \begin{bmatrix} \Delta\theta \\ \Delta q \end{bmatrix} \tag{6}$$

The linearized eigenvalues are $\pm 12.0439j$ and no conclusions about the stability of the system can be drawn. Rewrite the internal dynamics (5a) and (5b) as

$$\ddot{\theta} = \frac{1}{I_y} \left(-M_a\theta - mg(h_M \sin\theta - l_M \cos\theta) - Q_T \right) \tag{7}$$

to notice that the pitch-attitude dynamics does not contain any damping terms. In order to analyze its stability consider the quadratic positive-definite Lyapunov function $V_\theta = \frac{1}{2}\frac{M_a}{I_y}\theta^2 + \frac{1}{2}q^2$. The rate of change of the Lyapunov function along the trajectories of (5) is

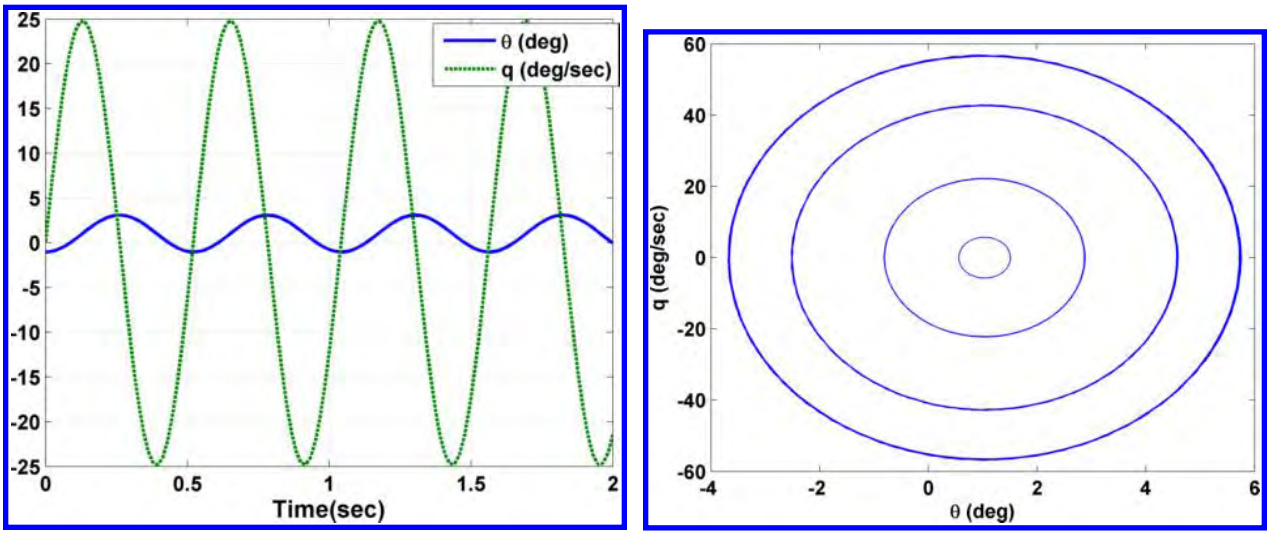
$$\dot{V}_\theta = \frac{M_a}{I_y}\theta\dot{\theta} + q\dot{q} \tag{8a}$$

$$= -\frac{mg}{I_y} h_M q \sin\theta + \frac{mg}{I_y} l_M q \cos\theta - \frac{1}{I_y} Q_T q \tag{8b}$$

$$= -\left[\frac{Q_T}{I_y} + \frac{mg}{I_y} h(\theta) \right] q \tag{8c}$$

Note the function $h(\theta) = h_M \sin\theta - l_M \cos\theta$ is monotonically increasing on the set $\theta \in [-\pi/2, \pi/2]$. This observation along with the parameters given in Table 1 conclude that $\dot{V}_\theta < 0$ on the set $\{\theta \in [-0.0509, \pi/2] \bigcap q \in [0, \infty)\} \bigcup \{\theta \in [-pi/2, -0.0509] \bigcap q \in (-\infty, 0]\}$. On this set (θ_*, q_*) is the only equilibrium point and hence from the Poincaré-Bendixson[18] criterion it is concluded that a family of periodic orbits exist. This conclusion is confirmed in simulation and the results are presented in Figure 2. In fact the conclusions drawn from the Poincaré-Bendixson criterion are conservative since the simulation shows that a continuum of periodic orbits exist for the complete state-space. Thus the control inputs that stabilize the inertial position of the helicopter excite the periodic behaviour in pitch and exact input-output linearization is not a desirable control solution for the longitudinal model under study.

Notice the non-minimum phase behaviour is due to the nonlinear coupling between forces and pitching moment denoted by $h(\theta)$ in (8). This coupling comes through longitudinal tilt solution determined in (4)



(a) Time Response (b) Phase Portrait

Figure 2. Simulation illustrating the oscillatory response of the pitching motion

that produces the required translational forces. This dependence is explicitly seen by expanding the force terms on the right-hand side of (3).

$$\begin{aligned} X_f &= -T_M \sin(\theta + a_{1s}) \\ Z_f &= -T_M \cos(\theta + a_{1s}) + mg \end{aligned} \tag{9}$$

In the above equations X_f and Z_f represent the forces in the inertial plane acting along the north and down directions respectively. Approximate input-output linearization of the output dynamics is possible by neglecting the dependence of the longitudinal tilt on the forces. The approximate forces thus obtained are

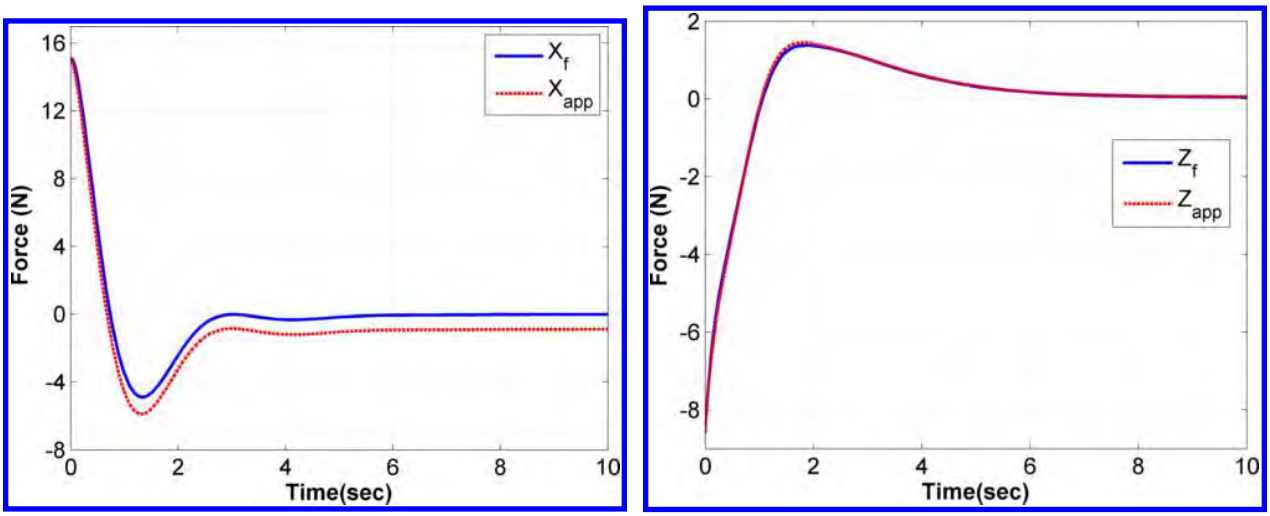
$$\begin{aligned} X_{app} &= -T_M \sin\theta \\ Z_{app} &= -T_M \cos\theta + mg. \end{aligned} \tag{10}$$

The exact and approximate forces acting on the helicopter under study is shown in Figure 3 for hover simulated in Section IV. Initially the helicopter is flying at an arbitrary flight condition and the forces are non-zero. Notice after two seconds the vehicle enters steady state and the exact horizontal and vertical forces become identically zero. However, the approximate horizontal force remains non-zero. The error between the exact and the approximate forces is shown in Figure 4. The error is over 100% in the horizontal forces while negligible in the vertical forces. This conclusion is consistent with the fact that rotor blade tilt induces a horizontal component of force in the helicopter and is not negligible. As mentioned in the introduction some studies use the approximate form given in (10) for control design. However, this large error limits these methods to guarantee only local bounded tracking. In this paper, the coupling terms are retained and asymptotic tracking is guaranteed.

C. Time-Scale Analysis of the Helicopter Model

In this section, an important observation regarding inherent time-scale characteristics of the model under consideration is made. This is done by studying the rate of change of the non-dimensional system equations. Toward this end, define a set of reference parameters $(t_0, x_0, z_0, u_0 = w_0 = V_0, \theta_0, q_0, m_0, F_{x0}, F_{z0}, M_0, g_0, I_{y0})$ and denote the respective dimension-less quantities as

$$\begin{aligned} \hat{t} = t/t_0 \quad \hat{x} = x/x_0 \quad \hat{z} = z/z_0 \quad \hat{u} = u/u_0 \\ \hat{w} = w/w_0 \quad \hat{\theta} = \theta/\theta_0 \quad \hat{q} = q/q_0 \end{aligned} \tag{11}$$



(a) Horizontal force (b) Vertical force

Figure 3. Exact and approximate forces acting on the helicopter model in hover

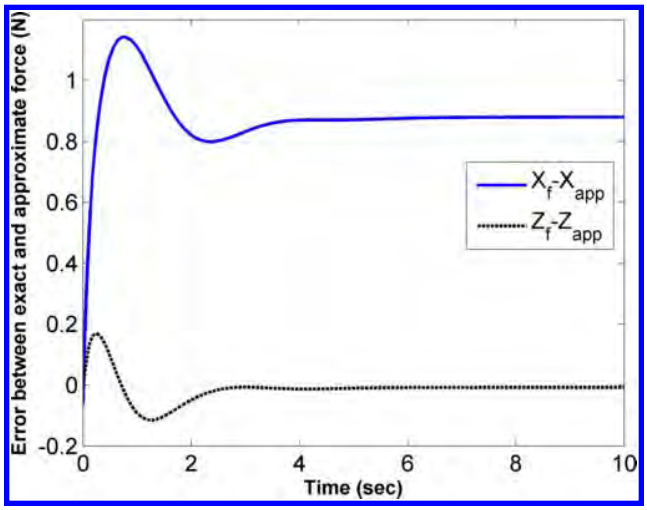


Figure 4. Error between exact and approximate force

The original dimensional equations given in (1) are transformed into non-dimensional form using definitions given in (11).

$$\begin{aligned}
\frac{d\hat{x}}{d\hat{t}} &= \left[\frac{t_0 V_0}{x_0}\right]\{\hat{u}\cos\theta + \hat{w}\sin\theta\} \\
\frac{d\hat{z}}{d\hat{t}} &= \left[\frac{t_0 V_0}{z_0}\right]\{-\hat{u}\sin\theta + \hat{w}\cos\theta\} \\
\frac{d\hat{u}}{d\hat{t}} &= -\left[\frac{t_0 q_0}{m_0}\right]\frac{\hat{q}\hat{w}}{\hat{m}} + \left[\frac{t_0 F_{x0}}{m_0 V_0}\right]\frac{\hat{F}_x}{\hat{m}} - \left[\frac{t_0 g_0}{V_0}\right]\hat{g}\sin\theta \\
\frac{d\hat{w}}{d\hat{t}} &= \left[\frac{t_0 q_0}{m_0}\right]\frac{\hat{q}\hat{u}}{\hat{m}} + \left[\frac{t_0 F_{z0}}{m_0 V_0}\right]\frac{\hat{F}_z}{\hat{m}} + \left[\frac{t_0 g_0}{V_0}\right]\hat{g}\cos\theta \\
\frac{d\hat{\theta}}{d\hat{t}} &= \left[\frac{t_0 q_0}{\theta_0}\right]\hat{q} \\
\frac{d\hat{q}}{d\hat{t}} &= \left[\frac{t_0 M_0}{q_0 I_{y0}}\right]\frac{\hat{M}}{\hat{I}_y}
\end{aligned} \tag{12}$$

Without loss of generality assign $\left[\frac{t_0 V_0}{x_0}\right] = \left[\frac{t_0 V_0}{z_0}\right] = 1$ and $\left[\frac{q_0 I_{y0}}{t_0 M_0}\right] = \epsilon$ where $\epsilon << 1$. This leads to

$$\begin{aligned}
\frac{d\hat{x}}{d\hat{t}} &= \{\hat{u}\cos\theta + \hat{w}\sin\theta\} \\
\frac{d\hat{z}}{d\hat{t}} &= \{-\hat{u}\sin\theta + \hat{w}\cos\theta\} \\
\frac{d\hat{u}}{d\hat{t}} &= -\left[\frac{\epsilon t_0^2 M_0}{m_0 I_{y0}}\right]\frac{\hat{q}\hat{w}}{\hat{m}} + \left[\frac{t_0 g_0}{V_0}\right]\left\{\frac{\hat{F}_x}{\hat{m}} - \hat{g}\sin\theta\right\} \\
\frac{d\hat{w}}{d\hat{t}} &= \left[\frac{\epsilon t_0^2 M_0}{m_0 I_{y0}}\right]\frac{\hat{q}\hat{u}}{\hat{m}} + \left[\frac{t_0 g_0}{V_0}\right]\left\{\frac{\hat{F}_z}{\hat{m}} + \hat{g}\cos\theta\right\} \\
\frac{d\hat{\theta}}{d\hat{t}} &= \left[\frac{\epsilon t_0^2 M_0}{I_{y0}\theta_0}\right]\hat{q} \\
\epsilon\frac{d\hat{q}}{d\hat{t}} &= \frac{\hat{M}}{\hat{I}_y}
\end{aligned} \tag{13}$$

where $F_{x0} = F_{z0} = m_0 g_0$ has been used. Notice that for any reasonable value of the mass of the vehicle $\left[\frac{\epsilon t_0^2 M_0}{m_0 I_{y0}}\right] = \epsilon$. Then $m_0 = \left[\frac{t_0^2 M_0}{I_{y0}}\right]$ and $\left[\frac{\epsilon t_0^2 M_0}{I_{y0}\theta_0}\right] = \left[\frac{\epsilon m_0}{\theta_0}\right]$ is an $O(1/\epsilon_1)$ quantity as the ratio of pitch-angle and mass of the vehicle is very small and $\epsilon_1 > \epsilon$. Finally, assuming that the vehicle is in hover $\left[\frac{t_0 g_0}{V_0}\right] = 1$ the non-dimensional form is obtained

$$\begin{aligned}
\frac{d\hat{x}}{d\hat{t}} &= \{\hat{u}\cos\theta + \hat{w}\sin\theta\} \\
\frac{d\hat{z}}{d\hat{t}} &= \{-\hat{u}\sin\theta + \hat{w}\cos\theta\} \\
\frac{d\hat{u}}{d\hat{t}} &= -\epsilon\frac{\hat{q}\hat{w}}{\hat{m}} + \left\{\frac{\hat{F}_x}{\hat{m}} - \hat{g}\sin\theta\right\} \\
\frac{d\hat{w}}{d\hat{t}} &= \epsilon\frac{\hat{q}\hat{u}}{\hat{m}} + \left\{\frac{\hat{F}_z}{\hat{m}} + \hat{g}\cos\theta\right\} \\
\epsilon_1\frac{d\hat{\theta}}{d\hat{t}} &= \hat{q} \\
\epsilon\frac{d\hat{q}}{d\hat{t}} &= \frac{\hat{M}}{\hat{I}_y}
\end{aligned} \tag{14}$$

Notice the above equations indicate that the rotational dynamics evolves faster than the translational counterpart. Such class of dynamical systems are called singularly perturbed and their analysis is carried out using singular perturbation theory. The above equations can be cast in the following compact form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{s}, \mathbf{z}, \mathbf{u}, \epsilon_1, \epsilon) \tag{15a}$$
$$\epsilon_1 \dot{\mathbf{s}} = \mathbf{h}(\mathbf{x}, \mathbf{s}, \mathbf{z}, \mathbf{u}, \epsilon_1, \epsilon) \tag{15b}$$
$$\epsilon \dot{\mathbf{z}} = \mathbf{g}(\mathbf{x}, \mathbf{s}, \mathbf{z}, \mathbf{u}, \epsilon_1, \epsilon) \tag{15c}$$

where $\mathbf{x} = [x, z, u, w]^T$ are the slow variables, $\mathbf{s} = [\theta]^T$ is the intermediate variable, $\mathbf{z} = [q]^T$ is the fast variable and $\mathbf{u} = [T_M, a_{1s}]^T$ is the control input to the system. The singular perturbation parameters ϵ and ϵ_1 characterize the different time scales in the system and satisfy $0 < \epsilon < \epsilon_1 << 1$. For presenting the concepts of singular perturbation theory, consider the two-time scale counterpart of (15)

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{z}, \mathbf{u}, \epsilon) \tag{16a}$$
$$\epsilon \dot{\mathbf{z}} = \mathbf{g}(\mathbf{x}, \mathbf{z}, \mathbf{u}, \epsilon) \tag{16b}$$

The system considered in (16) is labeled the *Slow System* and the independent variable t is called the *slow time-scale*. This system is equivalently written as the *Fast System*

$$\mathbf{x}' = \epsilon \mathbf{f}(\mathbf{x}, \mathbf{z}, \mathbf{u}, \epsilon) \tag{17a}$$
$$\mathbf{z}' = \mathbf{g}(\mathbf{x}, \mathbf{z}, \mathbf{u}, \epsilon) \tag{17b}$$

where $'$ represents derivative with respect to *fast time-scale*, $\tau = \frac{t - t_0}{\epsilon}$ and t_0 is the initial time. Note that in the slow system the slow states evolve at an ordinary rate whereas the fast states move at a rate of $O\left(\frac{1}{\epsilon}\right)$. In the fast system the fast states evolve at an ordinary rate and the slow variables move slowly at a rate of $O(\epsilon)$. Geometric singular perturbation theory[19] examines the behaviour of these singularly perturbed systems by studying the geometric constructs of the reduced-order models which are obtained by substituting $\epsilon = 0$ in (16) and (17). This results in *reduced slow subsystem*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{z}, \mathbf{u}, 0) \tag{18a}$$
$$\mathbf{0} = \mathbf{g}(\mathbf{x}, \mathbf{z}, \mathbf{u}, 0) \tag{18b}$$

and *reduced fast subsystem*

$$\mathbf{x}' = \mathbf{0} \tag{19a}$$
$$\mathbf{z}' = \mathbf{g}(\mathbf{x}, \mathbf{z}, \mathbf{u}, 0) \tag{19b}$$

The dynamics of the resulting reduced slow subsystem are constrained to lie upon an six dimensional smooth manifold defined by the set of points $(\mathbf{x}, \mathbf{u}) \in \mathbb{R}^4 \times \mathbb{R}^2$ that satisfy the algebraic equations (18b):

$$\mathcal{M}_0 : \mathbf{z_d} = \mathbf{z_d}(\mathbf{x}, \mathbf{u}) \tag{20}$$

This set of points is identically the fixed points of the reduced fast subsystem (19b). Furthermore, the flow on this manifold is described by the differential equations

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{z_d}(\mathbf{x}, \mathbf{u}), \mathbf{u}) \tag{21}$$

if the reduced fast subsystem is stable about the manifold $\mathcal{M}_0$. If the dynamics of (21) are locally asymptotically stable about the manifold, then it can be concluded that the complete system (16) is also locally asymptotically stable.[20]

III. Control Formulation and Stability Analysis

Singular perturbation theory concludes that the stability properties of the vehicle depend upon the identification of the manifold $\mathcal{M}_0$ for the internal states (θ, q). In general time-scale control approaches[21][pp 315-320] solve the nonlinear set of algebraic equations (18b) for the manifold first and then design a stabilizing

controller for the reduced system given in (21). Note however that for the helicopter model the moment equation (1d) is nonlinear and may possess multiple roots for the pitch-attitude angle. For a problem wherein the operating region is known apriori then one of these roots may be chosen. But this process soon becomes cumbersome and requires substantial vehicle knowledge. Additionally, this procedure restricts the results to local operating regimes. The authors have studied this problem for general singularly perturbed systems in [15,22]. In this paper an alternate approach is proposed that ensures uniqueness and the global nature of the results by considering the fast states as additional control variables. This allows computation of an unique reference for the internal states and maintains complete system stability. These ideas are mathematically formulated and analyzed in this section.

A. Control Synthesis

Using the procedure described in Section II.C, the reduced slow subsystem for (1) is obtained as

$$\begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \cos\theta_d & \sin\theta_d \\ -\sin\theta_d & \cos\theta_d \end{bmatrix} \begin{bmatrix} u \\ w \end{bmatrix} \tag{22a}$$

$$\begin{bmatrix} m\dot{u} \\ m\dot{w} \end{bmatrix} = \begin{bmatrix} -q_d w + F_x \\ q_d u + F_z \end{bmatrix} + \begin{bmatrix} \cos\theta_d & -\sin\theta_d \\ \sin\theta_d & \cos\theta_d \end{bmatrix} \begin{bmatrix} 0 \\ mg \end{bmatrix} \tag{22b}$$

where θ_d and q_d are manifolds to be determined. Take additional derivatives of the position coordinates to rewrite (22) as

$$\begin{bmatrix} \ddot{x} \\ \ddot{z} \end{bmatrix} = \frac{1}{m} \begin{bmatrix} \cos\theta_d & \sin\theta_d \\ -\sin\theta_d & \cos\theta_d \end{bmatrix} \begin{bmatrix} F_x \\ F_z \end{bmatrix} + \begin{bmatrix} 0 \\ g \end{bmatrix} \tag{23}$$

Equation (23) shows that the pitch-attitude angle along with the control variables effect the position dynamics. Thus, employ the pitch-attitude angle and the main rotor thrust T_M to accomplish the control objective. Toward this end, rewrite (23) as

$$m\ddot{x} = -T_M \sin(a_{1s}(\theta_d, q_d) + \theta_d) \tag{24a}$$

$$m\ddot{z} = -T_M \cos(a_{1s}(\theta_d, q_d) + \theta_d) + mg \tag{24b}$$

Note in forming the reduced slow subsystem the fast variables have been assumed to be on the desired manifolds (θ_d, q_d). Hence, the longitudinal tilt used in the design of slow control variables is a function of these desired manifolds. Further, define the tracking errors $\tilde{x} := x - x_r$ and $\tilde{z} := z - z_r$. Let the desired dynamics be specified as

$$m\ddot{x} = m(\ddot{x}_r - \alpha\dot{\tilde{x}} - \beta\tilde{x}) \tag{25a}$$

$$m\ddot{z} = m(\ddot{z}_r - \alpha_1\dot{\tilde{z}} - \beta_1\tilde{z}) \tag{25b}$$

Combining (24) and (25), the following relations are obtained

$$T_M = m\sqrt{(\ddot{x}_r - \alpha\dot{\tilde{x}} - \beta\tilde{x})^2 + (\ddot{z}_r - \alpha_1\dot{\tilde{z}} - \beta_1\tilde{z} - g)^2} \tag{26}$$

$$\theta_d = \arctan\frac{(\ddot{x}_r - \alpha\dot{\tilde{x}} - \beta\tilde{x})}{(\ddot{z}_r - \alpha_1\dot{\tilde{z}} - \beta_1\tilde{z} - g)} - a_{1s}(\theta_d, q_d) \tag{27}$$

Remark 1. The choice of using main rotor thrust, T_M over the longitudinal tilt for stabilization of the reduced slow subsystem was made considering their actuation time constants. It is well understood that thrust generation takes longer than rotation of an actuator surface or in this case the rotor blade. While previous work of authors in References [22], [23] assumed infinitely fast actuators, this paper helps in assigning control tasks according to actuator bandwidth.

Equations (26) and (27) complete the design for the slow variables of the system. Notice however that the manifold q_d is unknown at this point. Toward this end, formulate the intermediate subsystem as

Reduced Intermediate Subsystem

$$\breve{\mathbf{x}} = 0 \tag{28a}$$

$$\breve{\theta} = q_d \tag{28b}$$

$$M = 0 \tag{28c}$$

where $\breve{}$ is derivative with respect to $\frac{t-t_0}{\epsilon_1}$. The manifold q_d must be designed to ensure the pitch-attitude follows θ_d. This can be satisfied by the following relation obtained using dynamic inversion

$$q_d = -K_\theta(\theta - \theta_d) \tag{29}$$

where K_θ is the feedback gain.

The desired manifolds given in (27) and (29) depend on the longitudinal tilt a_{1s} which is unknown. From the discussion detailed in Section II.C, it is known (29) is a fixed point of the *Reduced Fast Subsystem*

$$\mathbf{x}' = 0 \tag{30a}$$

$$\theta' = 0 \tag{30b}$$

$$q' = \frac{M}{I_y} \tag{30c}$$

Thus, it is required that the following relation holds for all time

$$M = -I_y K_q(q - q_d) \tag{31}$$

where K_q is the feedback gain. Rearrange (31) using the definitions in (2c),(27) and(29) to get

$$T_M h_M \sin(a_{1s}) - T_M l_M \cos(a_{1s}) + M_a a_{1s} = Q_T - I_y K_q(q - q_d) \tag{32}$$

The nonlinear equation in (32) is solved for the control a_{1s} using the small-angle assumption

$$a_{1s} = \left[\frac{Q_T + T_M l_M}{T_M h_M + M_a}\right] - \left[\frac{I_y K_q}{T_M h_M + M_a}\right] \tilde{q} \tag{33}$$

where $\tilde{q} := q - q_d$. For completeness substitute (33) back in (27) and (29) to compute the desired internal states

$$\theta_d = \arctan \frac{(\ddot{x}_r - \alpha\dot{\tilde{x}} - \beta\tilde{x})}{(\ddot{z}_r - \alpha\dot{\tilde{z}} - \beta_1\tilde{z} - g)} - \left[\frac{Q_T + T_M l_M}{T_M h_M + M_a}\right] \tag{34a}$$

$$q_d = -K_\theta\theta + K_\theta \arctan \frac{(\ddot{x}_r - \alpha\dot{\tilde{x}} - \beta\tilde{x})}{(\ddot{z}_r - \alpha\dot{\tilde{z}} - \beta_1\tilde{z} - g)} - K_\theta\left[\frac{Q_T + T_M l_M}{T_M h_M + M_a}\right] \tag{34b}$$

This completes the control design procedure.

B. Stability Analysis

The following theorem summarizes the main result of the paper.

Theorem 1. *Suppose the controls T_M and a_{1s} of the system (1) are designed according to the feedback relations given in (26) and (33). Then for initial conditions in the operating region $|\tilde{\theta}| < 15deg$, $|a_{1s}| \leq 25deg$ and $0 < T_M \leq 69.48$ the control uniformly asymptotically stabilizes the non-minimum phase helicopter model (1) and equivalently drives the states $x(t) \to x_r(t)$ and $z(t) \to z_r(t)$ keeping all other states and control inputs bounded.*

Proof. The closed-loop system system in error coordinates is given as

$$\begin{aligned}
\dot{\tilde{x}} &= \tilde{x}_1 \\
\dot{\tilde{x}}_1 &= \frac{1}{m}F_x\cos(\tilde{\theta}+\theta_d) + \frac{1}{m}F_z\sin(\tilde{\theta}+\theta_d) - \dot{x}_{1r} \\
\dot{\tilde{z}} &= \tilde{z}_1 \\
\dot{\tilde{z}}_1 &= -\frac{1}{m}F_x\sin(\tilde{\theta}+\theta_d) + \frac{1}{m}F_z\cos(\tilde{\theta}+\theta_d) + g - \dot{z}_{1r} \\
\dot{\tilde{\theta}} &= q_d + \tilde{q} - \dot{\theta}_d \\
\dot{\tilde{q}} &= \frac{M_d + (M - M_d)}{I_y} - \dot{q}_d
\end{aligned} \tag{35}$$

where $\tilde{\theta} := \theta - \theta_d$, $\tilde{q} := q - q_d$ and $M_d = M_a a_{1s} + T_M h_M a_{1s} - T_M l_M - Q_T$ is the moment obtained after making the small-angle approximation in arriving at (33). The closed-loop system is equivalently written as

$$\begin{aligned}
\dot{\tilde{x}} &= \tilde{x}_1 \\
\dot{\tilde{x}}_1 &= \frac{1}{m}F_x\cos\theta_d + \frac{1}{m}F_z\sin\theta_d - \dot{x}_{1r} \\
&\quad + \frac{1}{m}F_x\left[\cos(\tilde{\theta}+\theta_d) - \cos\theta_d\right] + \frac{1}{m}F_z\left[\sin(\tilde{\theta}+\theta_d) - \cos\theta_d\right] \\
\dot{\tilde{z}} &= \tilde{z}_1 \\
\dot{\tilde{z}}_1 &= -\frac{1}{m}F_x\sin\theta_d + \frac{1}{m}F_z\cos\theta_d + g - \dot{z}_{1r} \\
&\quad - \frac{1}{m}F_x\left[\sin(\tilde{\theta}+\theta_d) - \sin\theta_d\right] + \frac{1}{m}F_z\left[\cos(\tilde{\theta}+\theta_d) - \cos\theta_d\right] \\
\dot{\tilde{\theta}} &= q_d + \tilde{q} - \dot{\theta}_d \\
\dot{\tilde{q}} &= \frac{M_d + (M - M_d)}{I_y} - \dot{q}_d
\end{aligned} \tag{36}$$

Using the relations in (25), (29) and (31) rearrange (36) to get

$$\begin{aligned}
\dot{\tilde{x}} &= \tilde{x}_1 \\
\dot{\tilde{x}}_1 &= -\alpha\tilde{x}_1 - \beta\tilde{x} + \frac{1}{m}\cos\theta_d[F_x - F_x(a_{1s}(\theta_d, q_d))] + \frac{1}{m}\sin\theta_d[F_z - F_z(a_{1s}(\theta_d, q_d))] \\
&\quad + \frac{1}{m}F_x\left[\cos(\tilde{\theta}+\theta_d) - \cos\theta_d\right] + \frac{1}{m}F_z\left[\sin(\tilde{\theta}+\theta_d) - \cos\theta_d\right] \\
\dot{\tilde{z}} &= \tilde{z}_1 \\
\dot{\tilde{z}}_1 &= -\alpha_1\tilde{z}_1 - \beta_1\tilde{z} - \frac{1}{m}\sin\theta_d[F_x - F_x(a_{1s}(\theta_d, q_d))] + \frac{1}{m}\cos\theta_d[F_z - F_z(a_{1s}(\theta_d, q_d))] \\
&\quad - \frac{1}{m}F_x\left[\sin(\tilde{\theta}+\theta_d) - \sin\theta_d\right] + \frac{1}{m}F_z\left[\cos(\tilde{\theta}+\theta_d) - \cos\theta_d\right] \\
\dot{\tilde{\theta}} &= -K_\theta\tilde{\theta} + \tilde{q} - \dot{\theta}_d \\
\dot{\tilde{q}} &= -K_q\tilde{q} + \frac{M - M_d}{I_y} - \dot{q}_d
\end{aligned} \tag{37}$$

Closed-loop system stability of the system states is analyzed using the Lyapunov function approach. Consider a positive-definite and decrescent Lyapunov function candidate

$$V(\tilde{x}, \tilde{x}_1, \tilde{z}, \tilde{z}_1, \tilde{\theta}, \tilde{q}) = \frac{1}{2}\left[\beta\tilde{x}^2 + \tilde{x}_1^2 + \beta_1\tilde{z}^2 + \tilde{z}_1^2 + \tilde{\theta}^2 + \tilde{q}^2\right] \tag{38}$$

for the complete closed-loop system. The derivative of V along the trajectories of (37) is given by

$$\begin{aligned}\dot{V} = &-\alpha\tilde{x}_1^2 + \frac{1}{m}\cos\theta_d[F_x - F_x(a_{1s}(\theta_d,q_d))]\tilde{x}_1 + \frac{1}{m}\sin\theta_d[F_z - F_z(a_{1s}(\theta_d,q_d))]\tilde{x}_1 \\ &+ \frac{1}{m}F_x\left[\cos(\tilde{\theta}+\theta_d) - \cos\theta_d\right]\tilde{x}_1 + \frac{1}{m}F_z\left[\sin(\tilde{\theta}+\theta_d) - \cos\theta_d\right]\tilde{x}_1 \\ &-\alpha_1\tilde{z}_1^2 - \frac{1}{m}\sin\theta_d[F_x - F_x(a_{1s}(\theta_d,q_d))]\tilde{z}_1 + \frac{1}{m}\cos\theta_d[F_z - F_z(a_{1s}(\theta_d,q_d))]\tilde{z}_1 \\ &-\frac{1}{m}F_x\left[\sin(\tilde{\theta}+\theta_d) - \sin\theta_d\right]\tilde{z}_1 + \frac{1}{m}F_z\left[\cos(\tilde{\theta}+\theta_d) - \cos\theta_d\right]\tilde{z}_1 \\ &-K_\theta\tilde{\theta}^2 + \tilde{\theta}\tilde{q} - \tilde{\theta}\dot{\theta}_d - K_q\tilde{q}^2 + \frac{M - M_d}{I_y}\tilde{q} - \tilde{q}\dot{q}_d\end{aligned} \tag{39}$$

Using the Lipschitz behaviour of the vector fields on the domain defined in Theorem 1 the following conditions hold

$$|\sin(\tilde{\theta}+\theta_d) - \sin\theta_d| \le 0.35|\tilde{\theta}| \tag{40}$$

$$|F_x - F_x(a_{1s}(\theta_d,q_d))| \le |T_M|\left|\frac{I_yK_q}{T_Mh_M + M_a}\right||\tilde{q}| \tag{41}$$

$$|\cos(\tilde{\theta}+\theta_d) - \cos\theta_d| \approx 0 \tag{42}$$

$$|F_z - F_z(a_{1s}(\theta_d,q_d))| \approx 0 \tag{43}$$

Note conditions given in (42) and (43) give bounds on the magnitude of the error between the exact and approximate vertical force. This bound remains close to zero for large changes in $\tilde{\theta}$ and this condition was numerically verified for the model under study in Section II. Resulting derivative of the Lyapunov function given in (39) using conditions (40) through (43) becomes

$$\begin{aligned}\dot{V} \le &-\alpha\tilde{x}_1^2 + \frac{1}{m}|T_M|\left|\frac{I_yK_q}{T_Mh_M + M_a}\right||\tilde{x}_1||\tilde{q}| + 0.35\frac{1}{m}|T_M||\tilde{x}_1||\tilde{\theta}| \\ &-\alpha_1\tilde{z}_1^2 + 0.35|T_M|\left|\frac{I_yK_q}{T_Mh_M + M_a}\right||\tilde{z}_1||\tilde{q}| + 0.35\frac{1}{m}|T_M||a_{1s}||\tilde{z}_1||\tilde{\theta}| \\ &-K_\theta\tilde{\theta}^2 + \tilde{\theta}\tilde{q} - \tilde{\theta}\dot{\theta}_d - K_q\tilde{q}^2 + \frac{M - M_d}{I_y}\tilde{q} - \tilde{q}\dot{q}_d.\end{aligned} \tag{44}$$

The time derivative of the manifolds θ_d and q_d is determined next. Toward this end, rearrange (27) as

$$\tan\gamma = \frac{X_{des}(t)}{Z_{des}(t)} \tag{45}$$

where $\gamma = \theta_d + a_{1s}(\theta_d, q_d)$, $X_{des} = \ddot{x}_r - \alpha\tilde{x}_1 - \beta\tilde{x}$ and $Z_{des} = \ddot{z}_r - \alpha_1\tilde{z}_1 - \beta\tilde{z} - g$ have been defined for convenience. Differentiate (45) to get

$$\dot{\gamma} = \frac{\cos\gamma}{T_M/m}\dot{X}_{des} - \frac{\sin\gamma}{T_M/m}\dot{Z}_{des} \tag{46}$$

using the fact $T_M/m = \sqrt{(X_{des}^2 + Z_{des}^2)}$ and definition of the angle γ. The time rate of change of the longitudinal tilt $a_{1s}(\theta_d, q_d)$ is determined by differentiating (33) along the inertial position trajectories.

$$\begin{aligned}\dot{a_{1s}} &= \frac{d}{dt}\left[\frac{T_Ml_M + Q_T}{T_Mh_M + M_a}\right] \\ &= \left[\frac{l_MM_a - h_MQ_T}{(T_Mh_M + M_a)^2}\right]\dot{T}_M\end{aligned} \tag{47}$$

where $\dot{T}_M = m\sin\gamma\dot{X}_{des} + m\cos\gamma\dot{Z}_{des}$. Combine (46) and (47), to determine the derivative of the manifolds

$$\begin{aligned}\dot{\theta}_d &= m\left[\frac{\cos\gamma}{T_M} - a_T\sin\gamma\right]\dot{X}_{des} + m\left[\frac{-\sin\gamma}{T_M} - a_T\cos\gamma\right]\dot{Z}_{des} \\ \dot{q}_d &= -K_\theta\dot{\tilde{\theta}}\end{aligned} \tag{48}$$

where $a_T = \frac{L_M M_a - h_M Q_T}{(T_M h_M + M_a)^2}$ and the various derivatives are a function of closed-loop system dynamics. Using properties (40) through (43) and (37)

$$|\dot{X}_{des}| \leq \alpha\beta|\tilde{x}| + (\alpha^2 - \beta)|\tilde{x}_1| + \frac{1}{m}|T_M|\left|\frac{\alpha I_y K_q}{T_M h_M + M_a}\right||\tilde{q}| + 0.35\frac{\alpha}{m}|T_M||\tilde{\theta}| \tag{49}$$
$$|\dot{Z}_{des}| \leq \alpha_1\beta_1|\tilde{z}| + (\alpha_1^2 - \beta_1)|\tilde{z}_1| + \frac{0.35}{m}|T_M|\left|\frac{\alpha_1 I_y K_q}{T_M h_M + M_a}\right||\tilde{q}| + 0.35\frac{\alpha_1}{m}|T_M||\tilde{\theta}|.$$

Combine (48), (49) and (44) to get

$$\begin{aligned}\dot{V} \leq & -\alpha\tilde{x}_1^2 + \frac{1}{m}|T_M|\left|\frac{I_y K_q}{T_M h_M + M_a}\right||\tilde{x}_1||\tilde{q}| + 0.35\frac{1}{m}|T_M||\tilde{x}_1||\tilde{\theta}| \\ & -\alpha_1\tilde{z}_1^2 + 0.35|T_M|\left|\frac{I_y K_q}{T_M h_M + M_a}\right||\tilde{z}_1||\tilde{q}| + 0.35\frac{1}{m}|T_M||a_{1s}||\tilde{z}_1||\tilde{\theta}| \\ & -K_\theta\tilde{\theta}^2 + |\tilde{\theta}||\tilde{q}| + (|\tilde{\theta}| + K_\theta|\tilde{q}|)|\dot{\theta}_d| - K_q\tilde{q}^2 + (K_\theta - K_\theta^2)|\tilde{\theta}||\tilde{q}|\end{aligned} \tag{50}$$

By definition a_T is a small quantity and $|cos\gamma| = |\sin\gamma| \leq 1$, define $\kappa = \frac{m}{|T_M|}$ which is again a small quantity. Substitute for time rate of change of the manifold θ_d into (50) to get

$$\begin{aligned}\dot{V} \leq & -\alpha\tilde{x}_1^2 + \frac{1}{m}|T_M|\left|\frac{I_y K_q}{T_M h_M + M_a}\right||\tilde{x}_1||\tilde{q}| + 0.35\frac{1}{m}|T_M||\tilde{x}_1||\tilde{\theta}| \\ & -\alpha_1\tilde{z}_1^2 + 0.35|T_M|\left|\frac{I_y K_q}{T_M h_M + M_a}\right||\tilde{z}_1||\tilde{q}| + 0.35\frac{1}{m}|T_M||a_{1s}||\tilde{z}_1||\tilde{\theta}| \\ & -K_\theta\tilde{\theta}^2 + |\tilde{\theta}||\tilde{q}| - K_q\tilde{q}^2 + (K_\theta - K_\theta^2)|\tilde{\theta}||\tilde{q}| \\ & +\kappa(|\tilde{\theta}| + K_\theta|\tilde{q}|)\Big[\alpha\beta|\tilde{x}| + (\alpha^2 - \beta)|\tilde{x}_1| + \alpha_1\beta_1|\tilde{z}| + (\alpha_1^2 - \beta_1)|\tilde{z}_1| \\ & -\frac{1}{m}|T_M|\left|\frac{I_y K_q}{T_M h_M + M_a}\right|(\alpha + 0.35\alpha_1)|\tilde{q}| - 0.35(\alpha + \alpha_1)\frac{1}{m}|T_M||\tilde{\theta}|\Big]\end{aligned} \tag{51}$$

Rearrange (51) to get

$$\dot{V} \leq -\Psi^T\mathbb{K}\Psi \tag{52}$$

where $\Psi = [\tilde{x}, \tilde{x}_1, \tilde{z}, \tilde{z}_1, \tilde{\theta}, \tilde{q}]^T$ and matrix $\mathbb{K}$ is given below

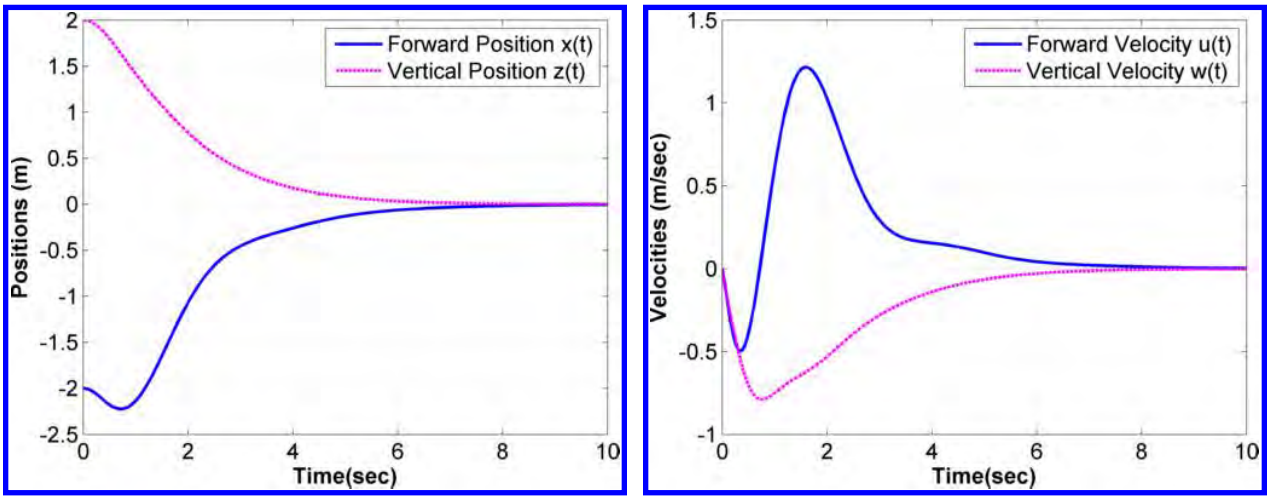
$$\mathbb{K} = \begin{bmatrix} 0 & 0 & 0 & 0 & \mu_1 & \mu_2 \\ 0 & -\alpha & 0 & 0 & \mu_3 & \mu_4 \\ 0 & 0 & 0 & 0 & \mu_5 & \mu_6 \\ 0 & 0 & 0 & -\alpha_1 & \mu_7 & \mu_8 \\ \mu_1 & \mu_3 & \mu_5 & \mu_7 & -K_\theta - 0.35(\alpha + \alpha_1)\frac{|T_M|}{m} & \mu_9 \\ \mu_2 & \mu_4 & \mu_6 & \mu_8 & \mu_9 & -K_q - \frac{|T_M|}{m}\kappa K_\theta(\alpha + 0.35\alpha_1)\left|\frac{I_y K_q}{T_M h_M + M_a}\right| \end{bmatrix} \tag{53}$$

where $\mu_1 = \frac{\kappa\alpha\beta}{2}$, $\mu_2 = K_\theta\mu_1$, $\mu_3 = \frac{0.35|T_M|}{2m} + 0.5\kappa(\alpha^2 - \beta)$, $\mu_4 = \frac{1}{2m}|T_M|\left|\frac{I_y K_q}{T_M h_M + M_a}\right| + 0.5\kappa K_\theta(\alpha^2 - \beta)$, $\mu_5 = \frac{\kappa\alpha_1\beta_1}{2}$, $\mu_6 = K_\theta\mu_5$, $\mu_7 = \frac{0.1527|T_M|}{2m} + 0.5\kappa(\alpha_1^2 - \beta_1)$, $\mu_8 = \frac{0.35}{2m}|T_M|\left|\frac{I_y K_q}{T_M h_M + M_a}\right| + 0.5\kappa K_\theta(\alpha_1^2 - \beta_1)$, $\mu_9 = 0.5(K_\theta - K_\theta^2 + 1) - \kappa\frac{|T_M|}{2m}\left|\frac{I_y K_q}{T_M h_M + M_a}\right|(\alpha + 0.35\alpha_1) - 0.35\kappa K_\theta(\alpha + \alpha_1)\frac{1}{2m}|T_M|$ are constants function of the feedback gains. Hence, the matrix $\mathbb{K}$ is negative semi-definite by appropriate choice of the feedback gains. Note the semi-definiteness property is due to the small values of constants μ_1, μ_2, μ_5 and μ_6. Since $\dot{V} \leq 0$ and $V > 0$, all terms in $V \in \mathcal{L}_\infty$ that is $\{\tilde{x}, \tilde{x}_1, \tilde{z}, \tilde{z}_1, \tilde{\theta}, \tilde{q}\} \in \mathcal{L}_\infty$. Furthermore, since the reference trajectory states are bounded, all terms in expressions for T_M and a_{1s} in (26) and (33) respectively are bounded. Hence the right hand side of the closed-loop system in (37) is bounded and thus $\dot{\Psi} \in \mathcal{L}_\infty$. Thus using Barbalat's lemma[24] it is concluded that signals of vector $\Psi \to 0$ as $t \to \infty$ and the result in Theorem 1 follows. This completes the stability analysis. □

IV. Simulation Study: Hover Control

The purpose of this section is to illustrate the preceding theoretical developments and demonstrate the controller performance for an autonomous helicopter model. The reference trajectory and all its derivatives are set to zero to illustrate the stabilizing performance of the controller for the open-loop non-minimum phase system (discussed in Section II.B). The feedback gains were chosen to preserve the time-scale nature of the helicopter model $\alpha = \alpha_1 = 2, \beta = \beta_1 = 1$, $K_\theta = 3$ and $K_q = 10$. The various constants for matrix $\mathbb{K}$ are $\mu_1 = \mu_5 = 0.082$, $\mu_2 = \mu_6 = 0.245$, $\mu_3 = 2.26$, $\mu_4 = 0.755$, $\mu_7 = 1.06$, $\mu_8 = 0.5$ and $\mu_9 = -4.68$. The corresponding eigenvalues of the matrix $\mathbb{K}$ are $\lambda_{1,2} = 0.00$, $\lambda_3 = -1.65$, $\lambda_4 = -1.99$, $\lambda_5 = -8.62$ and $\lambda_6 = -22.39$ and Theorem 1 guarantees asymptotic stability. The initial conditions chosen were $x(0) = -2m$, $z(0) = 2m$, $u(0) = w(0) = 0m/sec$, $\theta(0) = 15deg$ and $q(0) = 30deg/sec$.

Figure 5 and Figure 6 present the closed-loop response of the helicopter. The controller demonstrated asymptotic tracking irrespective of the desired reference trajectory in the domain $(x, z, u, w, \theta, q) \in [-50, 50]m \times [-15, 50]m \times [-30, 20]m/sec \times [-5, 20]m/sec \times (-\pi/2, \pi/2)rad \times [-\pi, \pi]rad/sec$. Notice that the large initial condition errors die out within the first 6seconds. The forward velocity is increased in order to correct the error in forward position. Close output tracking is a result of precision desired manifold following by the internal states. The pitch-attitude angle settles down to the trim value of $0.018rad(1.03deg)$ that is automatically computed by the manifold (34a). The time-scale behaviour of the system states is apparent in the time histories. Notice that the pitch-rate starts to follow the desired manifold within 2seconds followed by the response of the pitch-attitude angle closely tracking the desired manifold within 4seconds. The transient errors of the slowest and also the outputs of the problem under study die out in 6seconds. The control inputs are shown in Figure 7. The control inputs settle down to the trim values $T_M = 48.02N$ and $a_{1s} = -0.018rad(-1.03deg)$ once the system errors have stabilized about the origin. The two-dimensional trajectory of the helicopter is shown in Figure 8. Initially the helicopter corrects the large error in the pitch-attitude angle. This is done by reducing the requirements on pitch-rate and in turn the longitudinal tilt. After this correction, the vehicle starts climbing to the desired hover position. From then on, the helicopter remains in hover.



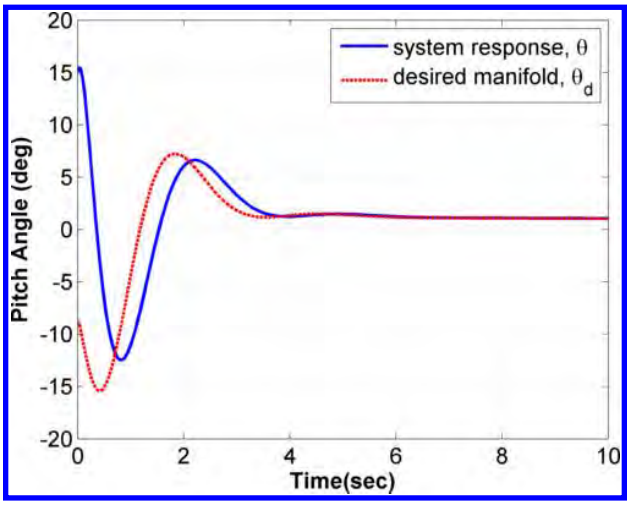
(a) Position Time Histories (b) Velocity Time Histories

Figure 5. Closed-Loop Output Response of the Helicopter

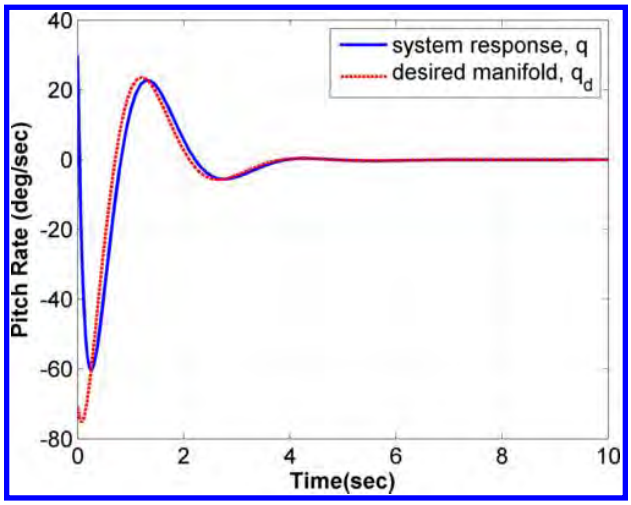
V. Conclusions

A control formulation for output tracking of an autonomous nonlinear non-minimum phase helicopter was developed. The desired internal-state reference and feedback control to stabilize the unstable internal dynamics were computed using the inherent time-scales of the system. Controller performance was demonstrated through numerical simulation for the helicopter in hover.

Based on the results presented in the paper, the following conclusions are drawn. The final output tracking error for the positions remained within $|0.0010|$. This perfect output tracking was a result of perfect internal

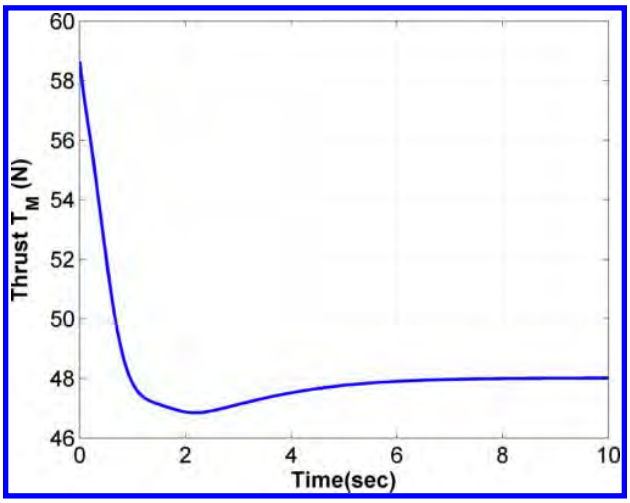


(a) Pitch-Attitude Angle Time History

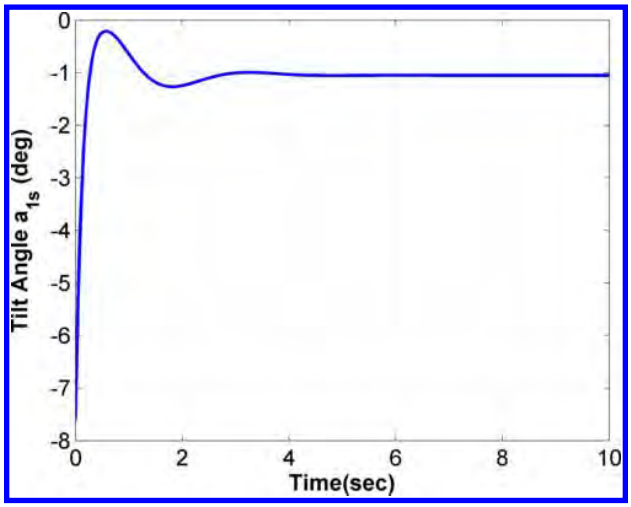


(b) Pitch-Rate Time History

Figure 6. Closed-Loop Internal Dynamics of the Helicopter



(a) Main Rotor Thrust Time History



(b) Longitudinal Tilt Angle Time History

Figure 7. Control Inputs to the Helicopter

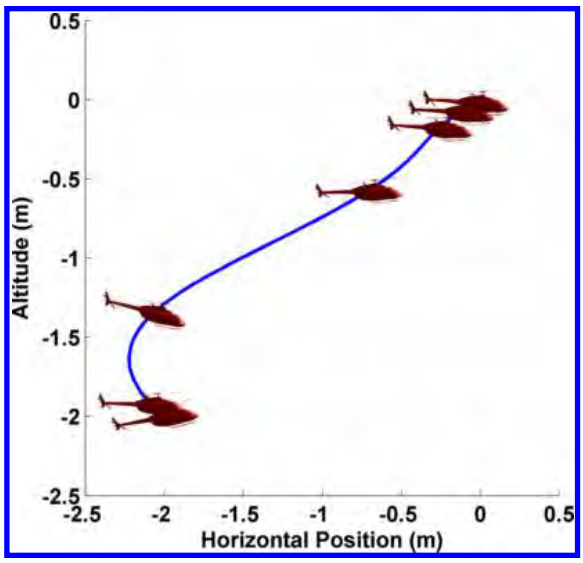


Figure 8. Closed-loop Trajectory of the Helicopter

state tracking that was achieved by the nonlinear feedback law. The results of Theorem 1 are restricted in operating regime due to the small angle approximation made in (33). Unlike previous approaches this limitation is not due to simplifications made to the dynamical model and can be improved by use of non-affine control methods. In fact the conclusions regarding operating region of the controller from Theorem 1 are conservative. As shown in the simulation section, the controller demonstrates stable performance for a large operating region. Additionally, the controller is causal and does not require any prior information or preview of the desired reference.

Acknowledgements

This material is based upon work supported in part by the U.S. Air Force Office of Scientific Research under contract FA9550-08-1-0038, with technical monitor Dr.Fariba Fahroo. This support is gratefully acknowledged by the authors. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Air Force.

References

[1]L.Qui and Davison, E., "Performance Limitations of Non-Minimum Phase Systems in the Servomechanism Problem," *Automatica*, Vol. 29, 1993, pp. 337–349.

[2]Lee, J. and Ha, I., "Autopilot Design for Highly Maneuvering STT Missiles via Singular Perturbation-like Technique," *IEEE Transactions on Control System Technology*, Vol. 7, 1999, pp. 527–541.

[3]Hauser, J., Sastry, S. S., and Meyer, G., "Nonlinear Control Design for Slightly Nonminimum Phase Systems: Application to V/STOL Aircraft," *Automatica*, Vol. 28, 1992, pp. 665–679.

[4]Tomlin, C., Lygeors, J., Benvenuti, L., and Sastry, S., "Output Tracking for a Non-Minimum Phase Dynamic CTOL Aircraft Model," *Proceedings of the 34th Conference on Decision and Control*, 1995, pp. 1867–1872, New Orleans, LA.

[5]Koo, T. J., Hoffman, F., Shim, H., and Sastry, S., "Control Design and Implementation of Autonomous Helicopter," *Invited session in Conference on Decision & Control*, 1998, Florida.

[6]Mahony, R., Hamel, T., and Dzul, A., "Hover control via Lyapunov Control for an Autonomous Model Helicopter," *Proceedings of the 38th Conference on Decision & Control*, 1999, pp. 3490–3495, Phoenix, Arizona.

[7]Frazzoli, E., Dahleh, M. A., and Feron, E., "Trajectory Tracking Control Design for Autonomous Helicopters Using a Backstepping Algorithm," *Proceedings of the American Control Conference*, 2000, pp. 4102–4107, Chicago, Illinois.

[8]Zhou, H., Pei, H., and Zhao, Y., "Trajectory Tracking Control of a Small Unmanned Helicopter Using MPC and Backstepping," *Proceedings of the American Control Conference*, 2011, pp. 1583–1589, San Francisco, CA.

[9]Johnson, E. and Kannan, S., "Adaptive Trajectory Control for Autonomous Helicopters," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 28, 2005, pp. 524–538.

[10]Lee, C.-T. and Tsai, C.-C., "Improved Nonlinear Trajectory Tracking Using RBFNN for a Robotic Helicopter," *International Journal of Robust and Nonlinear Control*, Vol. 20, 2010, pp. 1079–1096.

[11]Gonzalez, A., Mahtani, R., Bejar, M., and Ollero, A., "Control and Stability Analysis of an Autonomous Helicopter," *Proceedings of the 2004 World Automation Congress*, 2004, pp. 399–404.

[12]Esteban, S., Aracil, J., and Gordilli, F., "Lyapunov Based Asymptotic Stability Analysis of a Three-Time Scale Radio/Control Helicopter Model," *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Honolulu, Hawaii, August 2008, AIAA-2008-6566.

[13]Devasia, S., "Output Tracking with Nonhyperbolic and Near Nonhyperbolic Internal Dynamics: Helicopter Hover Control," *Journal of Guidance, Control and Dynamics*, Vol. 20, 1997, pp. 573–580.

[14]Avanzini, G. and de Matteis, G., "Two-Timescale Inverse Simulation of a Helicopter Model," *Journal of Guidance, Control and Dynamics*, Vol. 24, 2001, pp. 330–339.

[15]Siddarth, A. and Valasek, J., "Tracking Control Design for Non-Standard Nonlinear Singularly Perturbed Systems," *Proceedings of American Control Conference*, Montreal, Canada, June 2012.

[16]Siddarth, A. and Valasek, J., "Output Tracking of Non-Minimum Phase Dynamics," *AIAA Guidance, Navigation and Control Conference*, Portland,Oregon, August 2011, AIAA-2011-6487.

[17]Prouty, R. W., *Helicopter Performance, Stability and Control*, PWS Publishers, 1986, 558-559.

[18]Khalil, H. K., *Nonlinear Systems*, Prentice Hall, Upper Saddle River, NJ, 3rd ed., December 2001.

[19]Fenichel, N., "Geometric Singular Perturbation Theory for Ordinary Differential Equations," *Journal of Differential Equations*, Vol. 31, 1979, pp. 53–98.

[20]Kaper, T. J., "An Introduction to Geometric Methods and Dynamical Systems Theory for Singular Perturbation Problems," *Analyzing Multiscale Phenomena Using Singular Perturbation Methods: Proceedings of Symposia in Applied Mathematics*, edited by J. Cronin and J. Robert E.O'Malley, American Mathematical Society, 1986, pp. 85–131.

[21]Kokotović, P., Khalil, H. K., and Reilly, J. O., *Singular Perturbation Methods in Control: Analysis and Design*, Academic Press, 1986.

[22]Siddarth, A. and Valasek, J., "Kinetic State Tracking for a Class of Singularly Perturbed Systems," *Journal of Guidance, Control and Dynamics*, Vol. 34, No. 3, 2011, pp. 734–749.

[23]Siddarth, A. and Valasek, J., "Global Tracking Control Structures for Nonlinear Singularly Perturbed Aircraft," *Advances in Aerospace Guidance, Navigation and Control*, edited by F. Holzapfel and S. Theil, Springer, 2011, pp. 235–246.

[24]Slotine, J.-J. E. and Li, W., *Applied Nonlinear Control*, Prentice Hall, 1991.

Adaptive Dynamic Inversion Control of Linear Plants With Control Position Constraints

John Valasek, *Senior Member, IEEE,* Maruthi Ram Akella, *Member, IEEE,* Anshu Siddarth, *Student Member, IEEE,* and Elizabeth Rollins, *Student Member, IEEE*

***Abstract*—For a class of linear time-invariant systems with uncertain parameters, this paper proposes and develops a notion of the Domain of Control Authority to achieve stable adaptation in the presence of control position limits. The Domain of Control Authority defines the subspace in which the plant state can be driven in any arbitrary direction by bounded control. No restrictions are placed on the stability of the open-loop system. To address the problem of containing the state within the Domain of Control Authority, a switching control strategy with a direction consistent control constraint mechanism is developed for an unstable plant. This restricts the resultant direction of the rate of change of the state to be the same as the direction of the desired reference state. Stability proofs are provided, and controller performance is demonstrated with numerical examples of a two degree-of-freedom dynamic model and an F-16XL aircraft model.**

***Index Terms*—Adaptive control, control saturation constraints, direction consistent control constraint mechanism, dynamic inversion, linear systems.**

I. Introduction

ACTUATOR saturation is a major consideration for all practical control systems, and many strategies to overcome its effects have been studied. For example, Hu and Lin have done seminal work in analyzing the controllability and stabilization of unstable, linear time-invariant systems with input saturation [1]–[3]. They explicitly identified the null controllable region of the state-space for linear systems with saturated linear feedback. However, their work does not address systems with uncertain parameters. Traditionally, adaptive control assumes full control authority and lacks a theoretical basis for control in the presence of actuator saturation limits. Saturation is a problem for adaptive systems since continued adaptation in the presence of saturation may lead to instability. In the past, much effort has been expended for adaptive control design in the presence of input saturation constraints [4]. Karason and Annaswamy presented the concept of modifying the error proportional to the control deficiency [5]. They laid out a rigorous mathematical proof of the boundedness of signals for a model reference framework and identified a set of initial conditions of the plant and the controller for which a stable controller could be realized. Akella, Junkins, and Robinett devised a methodology to impose actuator saturation constraints on the adaptive control law analogous to Pontryagin's principle for optimal control in order to make the error energy rate as negative as possible with admissible controls [6]. They identified a boundary layer term, which is the difference between the calculated control and the applied control, and imposed conditions on the adaptive update laws to bound the boundary layer thickness. More recently, Johnson and Calise applied the concept of "pseudo-control hedging" to adaptive control, which is a fixed gain adjustment to the reference model that is proportional to the control deficiency [7]. Lavretsky and Hovakimyan have proposed a new design approach called "positive μ-modification" that guarantees that the control never incurs saturation [8]. In [9] the "$\mathcal{L}_1$ adaptive controller" is extended to include control constraints for linear plants with known control influence. Hong and Yao [10] synthesized a robust controller specifically for precision control of linear motor drive systems using backstepping, while addressing the different physical uncertainties. Kahveci and Ioannou [11] extended the anti-windup compensator design for stable systems with actuator position and rate limits, and a similar problem was addressed in Leonessa *et al.* [12] by modifying the reference to maintain system stability and control within bounds.

Dynamic Inversion is an approach which has been widely used in recent years for the control of nonlinear systems, especially in the field of aerospace engineering [13]–[16]. A fundamental assumption in this approach is that the inherent plant dynamics are modeled accurately, and therefore can be canceled exactly by the feedback functions. In practice, this assumption is not realistic; the dynamic inversion controller requires some level of robustness to suppress undesired behavior due to plant uncertainties. To overcome this robustness problem, an adaptive model of the plant dynamics sometimes is used to facilitate the inversion, which is then updated in real-time based on the response of the system. This gives rise to an entire class of controllers which may be referred to as adaptive dynamic inversion controllers [17].

This paper investigates problems introduced in adaptive dynamic inversion control schemes due to bounds on the control

Manuscript received October 21, 2010; revised February 03, 2011; accepted June 11, 2011. Manuscript received in final form June 13, 2011. Date of publication July 25, 2011; date of current version May 22, 2012. Recommended by Associate Editor N. Hovakimyan. This work was supported in part by the U.S. Air Force Office of Scientific Research under Contract FA9550–08-1–0038, with technical monitor Dr. F. Fahroo, and by the Texas Institute for Intelligent Bio-Nano Materials and Structures. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. Air Force or NASA.

J. Valasek, A. Siddarth and E. Rollins are with the Vehicle Systems and Controls Laboratory, Aerospace Engineering Department, Texas A&M University, College Station, TX 77843-3141 USA (e-mail: valasek@tamu.edu; anshun1@tamu.edu; erollins@tamu.edu).

M. R. Akella is with the Department of Aerospace Engineering and Engineering Mechanics, University of Texas at Austin, Austin, TX 78712-0235 USA (e-mail: makella@mail.utexas.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCST.2011.2159976

and develops a three component control scheme to overcome them. The contributions of this paper are the identification of the domain of attraction considering the control position limit and the development of a switching control strategy to contain the plant within this domain. Another novel idea is that of a direction consistent control constraint mechanism in the presence of control saturation. This is achieved in part by preserving the control input direction. While the idea of preserving the control input direction using control allocation is not entirely new [18], it is restricted to preserving the direction of the control vector only. This paper formalizes and extends a concept by Tandale and Valasek in [19] that not only preserves the direction of the control, but also attempts to preserve the direction of the resultant rate of change of the state to be the same as that of the desired rate. Additionally, a modified adaptation mechanism is implemented to prevent incorrect adaptation arising from trajectory errors due to control saturation. Here the mathematical development of the control scheme and the adaptation mechanisms is presented, along with proofs for the convergence of the tracking error and the stability of the overall control scheme.

This paper is organized as follows. Section II describes the class of plants that are considered. Section III defines the concept of the Domain of Control Authority (DCA) for plants with bounded control. The switching control strategy and the direction consistent control constraint mechanism are explained in Section IV. The development of the control law and the modified update law to prevent the incorrect update of parameters due to saturation is presented in Section VI. Section VII presents simulation results for a two-dimensional planar plant and an F-16XL aircraft model. Finally, conclusions are presented in Section VIII.

II. System Dynamics

Consider linear time-invariant continuous dynamic systems of the form

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}_a \tag{1}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state vector, $A \in \mathbb{R}^{n\times n}$ is the matrix of unknown plant parameters, $\mathbf{u}_a(t) \in \mathbb{R}^n$ is the vector of applied controls driving the system, and $B \in \mathbb{R}^{n\times n}$ is the unknown control effectiveness matrix. For this work, the number of controls equals the number of states so that the control effectiveness matrix is square and non-singular to permit dynamic inversion. Each control u_{a_i} is symmetrically bounded between the values $[-u_{m_i}, +u_{m_i}]$. The plant matrices A and B are not known exactly. The nominal values of the plant matrices $\bar{A}$ and $\bar{B}$ are specified, with a percentage uncertainty for each element of the plant matrix given as

$$A_{ij} = \bar{A}_{ij} + \triangle A_{ij} \tag{2}$$

$$B_{ij} = \bar{B}_{ij} + \triangle B_{ij} \tag{3}$$

where $|\triangle A_{ij}| \leq 0.05|\bar{A}_{ij}|$ and $|\triangle B_{ij}| \leq 0.05|\bar{B}_{ij}|$.

The reference trajectory is specified in terms of $\mathbf{x}_r$, which is chosen such that it is uniformly continuous, bounded, and differentiable with first order continuous, bounded derivatives $\dot{\mathbf{x}}_r$. The control objective is to track any feasible reference trajectory that can be followed within the control limits. For trajectories that are not feasible with respect to the control limits, the objective is to track the reference trajectory as closely as possible, while maintaining stability and ensuring that all signals remain bounded. Further, it is assumed that the entire state vector is measurable and that no observer is necessary to estimate the states.

III. Domain of Control Authority (DCA)

One of the most fundamental issues associated with the control of a system is controllability. While unconstrained controllability [20] has been well understood for several decades, the understanding of constrained controllability is incomplete [4]. The following discussion considers how bounds on controls affect controllability. While a linear scalar plant is used to elucidate the concepts, the discussion extends to multiple-input-multiple-output (MIMO) plants in which the number of controls equals the number of states. Consider

$$\dot{x} = -a^*x + b^*u_a \tag{4}$$

where $x(t) \in \mathbb{R}^1$, and a^*, b^* are unknown scalars with $b^* \neq 0$ and $a^* > 0$ such that the inherent dynamics are stable. The applied control u_a is bounded symmetrically as $u_a \in [-u_m, +u_m]$, where $u_m > 0$ is a known control limit.

There are two types of constraints that may be imposed on the plant state-space because of the bounds on the control.

1) ***Control Authority Constraint:*** If the plant is open-loop stable, the only diverging tendency that can propagate the system away from the equilibrium point is provided by the control. This diverging tendency can be infinite for a system which is controllable and has unbounded control. If the control is bounded, there will be a boundary in the state-space beyond which the converging tendency of the plant is greater than the diverging tendency due to the bounded control.
2) ***Tracking Constraint:*** Consider the plant model from (4). Since the control is bounded within $[-u_m, +u_m]$, the rate of change of the state at any point of time is bounded by

$$\dot{x} = \begin{cases} -a^*x + b^*u_m, & \text{if } u = u_m \\ -a^*x - b^*u_m, & \text{if } u = -u_m \end{cases} \tag{5}$$

where x is the plant state at that instant of time. Any reference trajectory that the plant can successfully track must satisfy the rate bounds listed in (5).

The controllability test for linear systems ensures that the control can affect every state, but does not consider the effect of bounds on the control. To have complete authority over the plant, the bounded control must be able to overcome the inherent plant dynamics and prescribe the desired dynamics.

A. Case 1: Stable Plant ($a^* > 0$)

Considering the plant model of (4), the inherent plant dynamics are given by the term $-a^*x$. The control authority is limited by the bounds on the control to values of $\pm b^*u_m$, so there exists boundaries in the plant state-space beyond which the inherent plant dynamics will dominate the control effort, and the plant will not be controllable. These boundaries will be reached when an extremal control is necessary to cancel the inherent plant dynamics. In the interior region the control has the ability

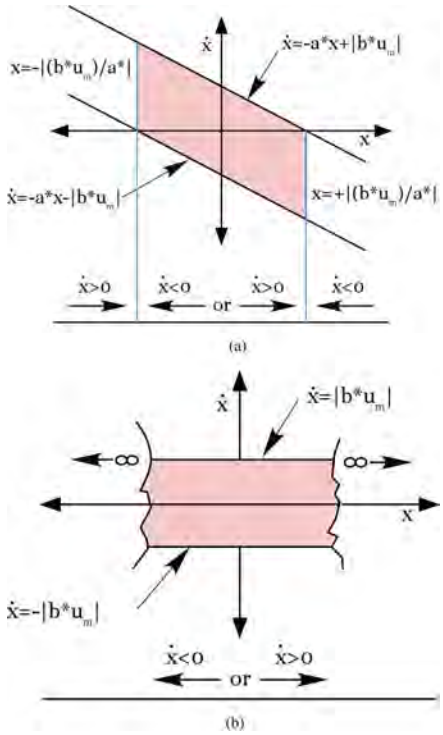


Fig. 1. Phase plot showing the domain for a traceable trajectory for an open-loop stable plant and for a neutrally stable plant. (a) Phase plot for an open-loop stable plant. (b) Phase plot for a neutrally stable plant.

to nullify the inherent plant dynamics without reaching its extremal values. This interior region is called the DCA. Referring to Fig. 1(a), the boundaries in terms of the plant state are

$$-a^* x_{\text{bound}} = -b^* u_a \tag{6}$$

$$-a^* x_{\text{bound}} = \pm b^* u_m \tag{7}$$

$$x_{\text{bound}} = \pm \frac{b^* u_m}{a^*}. \tag{8}$$

Equation (8) gives the vertical bound, and (5) gives the bounds on the rate of change of the state. Outside of the DCA boundary $\dot{x} < 0$ and the plant moves toward the origin. If the initial state is within the DCA, then the control cannot drive the state outside the DCA. If the initial state is outside the DCA, the inherent plant dynamics will drive the state into the DCA. Thus the bounded control does not lead to instability for an open-loop stable plant, but only to a limited operational envelope for the plant.

B. Case 2: Neutrally Stable Plant $(a^* = 0)$

The derivative of the state is affected only by the control

$$\dot{x} = b^* u_a. \tag{9}$$

The plant state is not bounded and can take any value on $[-\infty, +\infty]$, but the rate of change of the state is limited due to the control bound. The rate limits for a traceable trajectory [see Fig. 1(b)] are

$$|\dot{x}_r| \leq b^* u_m. \tag{10}$$

C. Case 3: Unstable Plant: $(a^* < 0)$

For current state x the unforced response $-a^* x$ drives the plant away from the state $x = 0$. If the plant reaches a state where the destabilizing tendency becomes greater than the maximum restoring contribution that the control can provide, then the state continues to diverge, such that $x \to \infty$ if $|x| > |b^* u_m / a^*|$. These points determine the boundary of the DCA. If the state crosses these points, stability of the system cannot be recovered.

IV. Switching Control Strategy

Consider a plant that is required to track an arbitrary reference trajectory using a dynamic inversion controller. The bounded control must cancel the inherent plant dynamics yet retain sufficient control effort to prescribe a rate of change of the state in any arbitrary direction of the state-space. If an extremal value of control is necessary to cancel the inherent plant dynamics, then there is at least one direction in which the plant state cannot be driven. Therefore, *the DCA consists of the set of the system equilibrium states. Depending on the system stability, some of these plant states can be driven in any arbitrary direction by a bounded control*. The boundary of the DCA is defined by the states in which at least one control must take on its extremal value in order to cancel the inherent plant dynamics. Outside the DCA open-loop stable plants can never cross the DCA boundary and remain bounded. Unstable plants diverge since the inherent diverging tendency dominates the maximum possible converging tendency that the control can provide.

The solution strategy proposed here is to identify the DCA and to develop a control law to prevent the plant state from crossing the DCA boundary. The control required to perform the tracking objective may be applied when the state is not near the DCA boundary. Once the state nears the DCA boundary, the control can be switched to a stabilizing control that cancels the plant dynamics and provides a restoring tendency toward the origin. It should be noted that whenever the state is within the DCA, the magnitude of the rate of change of the state is restricted because of the bounded control, but the direction of the rate of change of the state is not limited. The Sections IV-A and IV-B discuss methods to identify the DCA boundary and the concept of stabilizing control.

A. Enforcing the Switching Control Law Without Explicit Identification of the DCA Boundary

The DCA is defined by the states where at least one control must equal its extremal value in order to cancel the inherent plant dynamics. At the boundary of the DCA, (1) becomes

$$0 = A^* \mathbf{x}_{\text{boundary}} + B^* \mathbf{u}_{\text{extremal}}. \tag{11}$$

The entire DCA boundary can be evaluated by substituting all possible values that the vector $\mathbf{u}$ can take such that $u_i = \pm u_{m_i}$ for at least one i, where $i = 1, \ldots, n$ and u_i indicates the ith control input.

Consider a 2-D state-space for simplicity of analysis. The DCA for this 2-D state-space defines a rectangular parallelogram whose vertices are obtained from (11) when both compo-

nents of the control vector are equal to any one of the four possible permutations of the extremal values $(\pm u_{m_1}, \pm u_{m_2})$. The edges of the parallelogram correspond to cases when only one component of the control vector is equal to an extremal value. The other component of the control vector can equal any value within the control bounds. Consequently, the DCA boundary can be calculated and stored. As the state approaches the DCA boundary, the control can be switched from a tracking control to a stabilizing control. This idea easily extends to n-dimensions, where the DCA is an n-dimensional parallelepiped. However, this approach requires explicit identification and storage of the DCA boundary, which can be computationally intensive for higher dimensional plants.

An alternate approach for determining the switching control law is to use a scalar measure that keeps track of how close the operating point is to the DCA boundary, instead of defining the DCA boundary explicitly. This approach can be implemented by identifying the control component necessary to cancel the inherent plant dynamics, $\mathbf{u}_{\text{cancel}}$, which can be obtained by solving the following equation at run-time:

$$\mathbf{u}_{\text{cancel}} = -B^{*-1}A^{*}\mathbf{x}. \tag{12}$$

The applied control can be switched from tracking to stability as $\mathbf{u}_{\text{cancel}}$ approaches $\mathbf{u}_{\text{extremal}}$, which occurs when at least one component of $\mathbf{u}_{\text{cancel}}$ is equal to $\mathbf{u}_m$. Since (12) is simple to solve can be solved at every time step, this approach eliminates the need for prior explicit identification of the DCA.

B. Direction Consistent Control Constraint Mechanism

For a multi-input plant the bounded control not only restricts the magnitude of the applied control, but also changes the direction of the system. Fig. 2 illustrates this concept. Consider a scenario with two controls u_1 and u_2. Assume that the control calculated by the control algorithm to track a desired reference $\mathbf{u}_c$ is greater than the control bounds shown by the box. If each control is saturated to its respective maximum, the control applied to the plant, $\mathbf{u}_a$, has a significantly different direction compared to $\mathbf{u}_c$. When this control is applied to the plant the resulting rate of change of state also has a different direction than the desired direction. Here we develop a control strategy that implements $\mathbf{u}_a$, direction consistent shown in Fig. 2 that is within the position limits, which not only maintains the same direction as $\mathbf{u}_c$, but also attempts to preserve the direction of the resultant rate of change of the state so that the direction of the resultant rate of change of the state is the same as that of the desired rate.

Consider Fig. 3 in which the plant is of the form $\dot{\mathbf{x}} = A\mathbf{x}+B\mathbf{u}$ and the desired control required to track the reference trajectory, $\mathbf{u}_{\text{desired}}$, is calculated. If $\mathbf{u}_{\text{desired}}$ is outside the control bounds, the saturated version of the control $\mathbf{u}_{\text{sat}}$ is applied. In Fig. 3(a), each component of $\mathbf{u}_{\text{desired}}$ is saturated to its respective maximum value. Consequently, $\mathbf{u}_{\text{sat}}$ has a different direction compared to $\mathbf{u}_{\text{desired}}$, and the resultant direction of $\dot{\mathbf{x}}_{\text{sat}}$ is different from $\dot{\mathbf{x}}_{\text{desired}}$. In Fig. 3(b), the saturation is enforced in such a way that the direction of $\mathbf{u}_{\text{sat}}$ is the same as that of $\mathbf{u}_{\text{desired}}$.

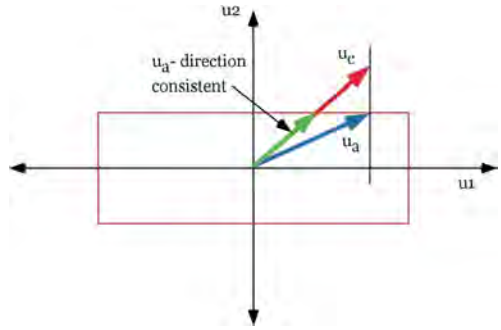


Fig. 2. Direction consistent control constraint mechanism.

However, the preservation of the control direction does not ensure that the resultant direction of $\dot{\mathbf{x}}_{\text{sat}}$ is the same as that of $\dot{\mathbf{x}}_{\text{desired}}$.

The control is calculated in two parts. The control necessary to cancel the inherent plant dynamics is calculated first, and then the control which produces a rate of change of the state in the desired direction is calculated. Referring to Fig. 3(c), the first component of the calculated control is equal to $-A\mathbf{x}$, and the second part is equal to $\dot{\mathbf{x}}_{\text{desired}}$. The first part of the calculated control will be within the control position limits since the plant state is restricted within the DCA. The second part of the control is subjected to direction consistent control saturation, which preserves the direction of the control vector. Therefore, the saturated version of the second part of the control equals $\dot{\mathbf{x}}_{\text{sat}}$, which also ensures that the direction of the resultant rate of change of the state is the same as the desired rate.

V. Tracking Control Law

The plant model is of the form given by (1)

$$\dot{\mathbf{x}} = \bar{A}\mathbf{x} + \bar{B}\mathbf{u}_a \tag{13}$$

where $\bar{A}$ and $\bar{B}$ are known constant matrices of compatible dimensions and the following assumptions hold.

Assumption A1: $\bar{B}$ is non-singular.

Assumption A2: The initial condition $\mathbf{x}_0 \in \mathbb{R}^n$ is such that $\nu \doteq \bar{B}^{-1}\bar{A}\mathbf{x}_0$ satisfies $|\nu_k| < u_{m_k}$ for all $k = 1, 2, \ldots, n$.

From this condition, due to continuity, there always exists a scalar $\lambda > 0$ such that $\nu_s = \bar{B}^{-1}(\bar{A} + \lambda I)\mathbf{x}_0$ also satisfies $|\nu_{s_k}| < u_{m_k}$ for all $k = 1, 2, \ldots, n$.

Assumption A3: We assume here that $\lambda > 0$ is chosen such that $|\nu_{s_k}| < u_{m_k}$ holds.

The tracking error is defined as

$$\mathbf{e}(t) = \mathbf{x}(t) - \mathbf{x}_r(t). \tag{14}$$

Differentiating the tracking error with respect to time, and substituting (13) into (14),

$$\dot{\mathbf{e}} = \bar{A}\mathbf{x} + \bar{B}\mathbf{u}_a - \dot{\mathbf{x}}_r. \tag{15}$$

Adding and subtracting $\lambda\mathbf{e}$ to (15), the equation for the error dynamics becomes

$$\dot{\mathbf{e}} = -\lambda\mathbf{e} + (\bar{A} + \lambda I)\mathbf{x} + \bar{B}\mathbf{u}_a - (\dot{\mathbf{x}}_r + \lambda\mathbf{x}_\mathbf{r}). \tag{16}$$

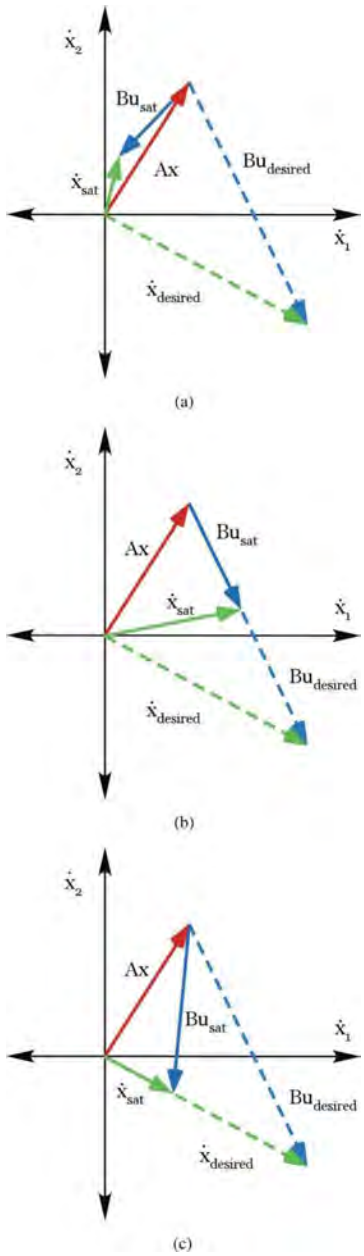


Fig. 3. Rate of change of the state due to various control saturation strategies. (a) Each component saturated to maximum. (b) Direction consistent control preserving the direction of the control vector. (c) Direction consistent control preserving the direction of the resultant rate of change of the state.

If

$$\mathbf{u}_a = -\bar{B}^{-1}(\bar{A} + \lambda I)\mathbf{x} + \bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r}) \tag{17}$$

we have $\dot{\mathbf{e}} = -\lambda \mathbf{e}$. The parameter λ thus governs the closed-loop dynamic system behaviour and needs to be chosen appropriately such that assumptions A2 and A3 are satisfied. We now state the following definitions:

$$\mathbf{u}_t = -\bar{B}^{-1}(\bar{A} + \lambda I)\mathbf{x} + \bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r}) \tag{18}$$

and

$$\mathbf{u}_s = -\bar{B}^{-1}(\bar{A} + \lambda I)\mathbf{x}. \tag{19}$$

Note that (18), referred in this work as the tracking control, has two components. The first term $-\bar{B}^{-1}(\bar{A} + \lambda I)\mathbf{x}$ cancels the inherent plant dynamics. The second term $\bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda \mathbf{x}_r)$ prescribes the desired dynamics necessary to track the reference trajectory. The first component is used as a measure of how close the state is to the DCA boundary and will be referred to as the stability control $\mathbf{u}_s$ defined by (19).

Further, define for each kth control input

$$u_{p_k} = \frac{|u_{s_k}|}{u_{m_k}} \tag{20}$$

where $k = 1, 2, 3, \ldots, n$. Then, the maximum of this ratio of stability control to the bound on each kth actuator is defined as

$$\mathbf{u}_{m_p} = \max(u_{p_1}, u_{p_2}, \ldots, u_{p_n}). \tag{21}$$

The quantity $\mathbf{u}_{m_p}$ is used as a measure of how close the current state is to the DCA boundary. Given the definition of $\mathbf{u}_{m_p}$ by virtue of (A3), we are ensured of $\mathbf{u}_{m_p}(0) < 1$. It should be noted that a value of $\mathbf{u}_{m_p} = 1$ indicates that the state is at the DCA boundary.

For a point which is inside the DCA, but approaching the boundary, the magnitude of $\mathbf{u}_{m_p}$ keeps increasing from 0 and reaches 1 on the boundary. To avoid saturation, choose scalar parameters s_1, s_2, p_1, and p_2 all close to 1, satisfying $0 < \mathbf{u}_{m_p}(0) < s_1 < s_2 < p_1 < p_2 < 1$ that decide the switch between tracking and stability control. Whenever, $\mathbf{u}_{m_p} > s_2$, the tracking control is close to being subjected to saturation. In this case, the concept of direction consistent control mechanism is implemented and the saturated version of the tracking control law is employed, given by the following equation:

$$\mathbf{u}_{t_{\text{sat}}} = -\bar{B}^{-1}(\bar{A}+\lambda I)\mathbf{x} + (1-\mathbf{u}_{m_p})\text{Sat}_{dc}\left\{\bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r})\right\}. \tag{22}$$

The direction consistent control saturation function Sat_{dc} maintains the direction of the resultant control to be the same as the direction of the desired control in spite of control saturation. The saturation function is defined in Section V-A.

A. Saturation Function Definitions

For any $\mathbf{u}_{\text{des}} \in \mathbb{R}^n$, we adopt a hard saturation

$$\mathcal{S}(\mathbf{u}_{\text{des}_k}) = \min(\mathbf{u}_{m_k}, |\mathbf{u}_{\text{des}_k}|)\text{sgn}(\mathbf{u}_{\text{des}_k}) \tag{23}$$

for all $k = 1, 2, \ldots, n$ or alternately, a "soft" saturation of the form

$$\mathcal{S}(\mathbf{u}_{\text{des}_k}) = \tanh\left(\frac{\beta \mathbf{u}_{\text{des}_k}}{\mathbf{u}_{m_k}}\right)\mathbf{u}_{m_k} \tag{24}$$

for all $k = 1, 2, \ldots, n$, where β is any positive scalar parameter. Qualitatively, $\beta \to \infty$ implies that the "soft" saturation approaches the "hard" saturation definition stated earlier. If each of the controls is allowed to saturate independently to the maximum allowed value, the applied control is

$$\mathbf{u}_{\text{app}_k} = \mathcal{S}(\mathbf{u}_{\text{des}_k}), \quad \text{for all} \quad k = 1, 2, \ldots, n \tag{25}$$

where $\mathcal{S}$ is the saturation function. If direction consistency is to be maintained, the proportion p_k to which each control is saturated is calculated by

$$q_k = \frac{\mathcal{S}(u_{\text{des}_k})}{u_{\text{des}_k}}, \quad \text{for all} \quad k = 1, 2, \ldots, n. \tag{26}$$

The minimum saturation proportion $q_{\min}$ is identified

$$q_{\min} = \min(q_1, q_2, \ldots, q_n) \tag{27}$$

and all controls are saturated with the same proportion. Note, by construction $q_k < 1$ for all k and accordingly, $q_{\min} < 1$ also holds. As a result, the direction of the applied control vector is the same as the calculated control vector $\mathbf{u}_{\rm des}$. Therefore

$$\mathrm{Sat}_{dc}(\mathbf{u}_{\rm des}) = q_{\min}\mathbf{u}_{\rm des}. \tag{28}$$

B. Complete Control Law

Continuing with the discussion of $\mathbf{u}_{m_p}$, whenever its value is greater than p_2 the stability control is employed. The other design parameters s_1 and p_1 are used in order to produce smooth transitions between controls; from $\mathbf{u_t}$ and $\mathbf{u}_{\mathbf{t}_{\rm sat}}$, and from $\mathbf{u}_{\mathbf{t}_{\rm sat}}$ to $\mathbf{u_s}$. This transition can be accomplished by a linear or higher-order interpolation as $\mathbf{u}_{m_p}$ takes values from s_1 to s_2 and p_1 to p_2. Finally, the applied control $\mathbf{u}_a$ is defined as

$$\begin{aligned}\mathbf{u}_a &= \mathbf{u}_t \text{ if } |\mathbf{u}_{t_k}| < \mathbf{u}_{m_k} \text{ and } \mathbf{u}_{m_p} < s_1 \forall\, k = 1, 2, \ldots, n\\ &\quad \text{else,}\\ \mathbf{u}_a &= \begin{cases} \mathbf{u}_b, & \text{if } s_1 \le \mathbf{u}_{m_p} < s_2\\ \mathbf{u}_{t_{\rm sat}}, & \text{if } s_2 \le \mathbf{u}_{m_p} < p_1\\ \mathbf{u}_c, & \text{if } p_1 \le \mathbf{u}_{m_p} < p_2\\ \mathbf{u}_s, & \text{if } \mathbf{u}_{m_p} \ge p_2 \end{cases}\end{aligned} \tag{29}$$

where

$$\mathbf{u}_b = f(\mathbf{u}_t, \mathbf{u}_{t_{\rm sat}}, s_1, s_2, \mathbf{u}_{m_p}) \tag{30}$$

$$\mathbf{u}_c = f(\mathbf{u}_{t_{\rm sat}}, \mathbf{u}_s, p_1, p_2, \mathbf{u}_{m_p}) \tag{31}$$

have been introduced for smooth transitions between controls. The function f is a third-order interpolation scheme defined as

$$\begin{aligned} f(\mathbf{u}_1, \mathbf{u}_2, c_1, c_2, \mathbf{u}_{m_p}) &= \mathbf{u}_1 + (3\xi^2 - 2\xi^3)(\mathbf{u}_2 - \mathbf{u}_1)\\ \xi &= \frac{\mathbf{u}_{m_p} - c_1}{c_2 - c_1}. \end{aligned} \tag{32}$$

Next, it is shown that selecting p_2 such that $\mathbf{u}_{m_p}(0) < p_2 < 1$ ensures that $\mathbf{u}_{m_p}(t) \le p_2 \forall\, t$. This can be proved as follows. Let $\mathbf{u}_{m_p} < p_2$ for some $t = t_1 > 0$ and let $\mathbf{u}_{t_k} < \mathbf{u}_{m_k}$ and $\mathbf{u}_{m_p} < s_1$, then the applied control $\mathbf{u}_a = \mathbf{u}_t$. Then for $t > t_1$, $x(t) = e^{-\lambda(t-t_1)}x(t_1) + \text{function}(x_r, \dot{x}_r, \lambda)$ and $\mathbf{u}_{m_p}(t) = e(-\lambda(t-t_1))\mathbf{u}_{m_p}(t_1) + \text{function}(x_r, \dot{x}_r, \lambda, \bar{A}, \bar{B})$. If $\mathbf{u}_{m_p}(t) > s_1$ the control smoothly switches to $\mathbf{u}_b$. The control continues to switch from there on depending on the value of parameter $\mathbf{u}_{m_p}$. Suppose there exists a finite time $t^* > 0$ such that $\mathbf{u}_{m_p}(t^*) = p_2$ and $\mathbf{u}_{m_p}(t) \ge p_2, \forall\, t > t^*$. Then, $\mathbf{u}_a(t) = \mathbf{u}_s(t) = -\bar{B}^{-1}(\bar{A} + \lambda I)\mathbf{x}(t) \quad \forall\, t \ge t^*$. This would yield

$$\mathbf{x}(t) = \mathbf{x}(t^*)e^{-\lambda(t-t^*)} \quad \forall\, t \ge t^*. \tag{33}$$

Using the definition of $\mathbf{u}_{m_p}(t)$ we have for all time

$$\mathbf{u}_{m_p}(t) = e^{-\lambda(t-t^*)}\mathbf{u}_{m_p}(t^*) \tag{34}$$

$$= e^{-\lambda(t-t^*)}p_2 \quad \le p_2 \tag{35}$$

which is a contradiction. Thus, $\mathbf{u}_{m_p}(t) \le p_2$ for all time $t \ge 0$. Finally, note that the closed-loop tracking error system is given by

$$\dot{\mathbf{e}} = -\lambda\mathbf{e} + \bar{B}\delta \tag{36}$$

where δ is a bounded signal of time whose explicit characterization is

$$\delta = \begin{cases} 0, & \text{if } |\mathbf{u}_{t_k}| < \mathbf{u}_{m_k} \text{ and } \mathbf{u}_{m_p} < s_1\\ \mathbf{u}_b - \mathbf{u}_t, & \text{if } s_1 \le \mathbf{u}_{m_p} < s_2\\ \mathbf{u}_{t_{\rm sat}} - \mathbf{u}_t, & \text{if } s_2 \le \mathbf{u}_{m_p} < p_1\\ \mathbf{u}_c - \mathbf{u}_t, & \text{if } p_1 \le \mathbf{u}_{m_p} < p_2\\ \mathbf{u}_s - \mathbf{u}_t, & \text{if } \mathbf{u}_{m_p} \ge p_2 \end{cases} \tag{37}$$

or

$$\delta = 0, \quad \text{if} |\mathbf{u}_{t_k}| < \mathbf{u}_{m_k} \text{and} \mathbf{u}_{m_p} < s_1 \tag{38}$$

$$\begin{aligned}\delta &= (3\xi^2 - 2\xi^3)((1 - \mathbf{u}_{m_p})\mathrm{Sat}_{dc}(\bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda\mathbf{x_r})))\\ &\quad - (3\xi^2 - 2\xi^3)\bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda\mathbf{x_r}),\\ &\text{if } s_1 \le \mathbf{u}_{m_p} < s_2, \text{ where } \xi = \frac{\mathbf{u}_{m_p} - s_1}{s_2 - s_1}\end{aligned} \tag{39}$$

$$\begin{aligned}\delta &= (1 - \mathbf{u}_{m_p})\mathrm{Sat}_{dc}(\bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda\mathbf{x_r}))\\ &\quad - (\bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda\mathbf{x}_r)),\\ &\text{if } s_2 \le \mathbf{u}_{m_p} < p_1\end{aligned} \tag{40}$$

$$\begin{aligned}\delta &= (1 - \mathbf{u}_{m_p})\mathrm{Sat}_{dc}(\bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda\mathbf{x_r}))\\ &\quad - (\bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda\mathbf{x_r}))\\ &\quad + (3\xi^2 - 2\xi^3)\left((1 - \mathbf{u}_{m_p})\mathrm{Sat}_{dc}(\bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda\mathbf{x_r}))\right),\\ &\text{if } p_1 \le \mathbf{u}_{m_p} < p_2, \text{ where } \xi = \frac{\mathbf{u}_{m_p} - p_1}{p_2 - p_1}\end{aligned} \tag{41}$$

$$\delta = -(\bar{B}^{-1}(\dot{\mathbf{x}}_r + \lambda\mathbf{x}_r)) \text{ if } \mathbf{u}_{m_p} \ge p_2. \tag{42}$$

Due to boundedness for δ, we have $\mathbf{e} \in \mathcal{L}_\infty$ ensuring boundedness for all closed-loop signals.

VI. Adaptive Case

Before proceeding to the control law for the case of uncertain parameters, the following assumption is made.

Assumption B1: Both B and $\bar{B}$ are non-singular. Further, suppose there exists a symmetric matrix L that is either positive or negative definite such that $LB = \bar{B}$, or

$$L = [I + \Delta B\bar{B}^{-1}]^{-1}. \tag{43}$$

Assume also that the function $\mathrm{sgn}(L)$ is known and defined such that $\mathrm{sgn}(L) = 1$ when L is positive definite, and $\mathrm{sgn}(L) = -1$ when L is negative definite. Additionally define matrix P,

$$P = L\mathrm{sgn}(L) \tag{44}$$

and let $p_M = \underset{\triangle B}{\lambda_{\max}}[P]$ and $p_m = \underset{\triangle B}{\lambda_{\min}}[P]$ be the maximum and minimum eigenvalues of P, respectively.

Define $\theta = B^{-1}(A + \lambda I)$, $\bar{\theta} = \bar{B}^{-1}(\bar{A} + \lambda I)$, $\phi = B^{-1}$, and $\bar{\phi} = \bar{B}^{-1}$, where λ is the specified closed-loop eigenvalue as

defined in earlier section. Thus $\phi = (\bar{B} + \triangle B)^{-1}$, and through the application of the matrix-inversion lemma

$$\phi = \bar{B}^{-1} - (\bar{B} + \triangle B)^{-1} \triangle B \bar{B}^{-1} \tag{45}$$
$$= \bar{\phi} + \triangle\phi \tag{46}$$

where $\triangle\phi = -(\bar{B} + \triangle B)^{-1} \triangle B \bar{B}^{-1}$. Now, given $\bar{B}$ and the fact that the uncertainty $\triangle B$ satisfies $B_{ij} = \bar{B}_{ij} + \triangle B_{ij}$ from (3), we denote

$$\triangle\phi_{ij}{}^{\max} = \max_{\triangle B} [\triangle\phi_{ij}]$$
$$\triangle\phi_{ij}{}^{\min} = \min_{\triangle B} [\triangle\phi_{ij}]. \tag{47}$$

Thus, we have

$$\phi_{ij} \in \left[\bar{\phi}_{ij} - \triangle\phi_{ij}{}^{\min}, \bar{\phi}_{ij} + \triangle\phi_{ij}{}^{\max}\right] \tag{48}$$

wherein the values of $\triangle\phi_{ij}{}^{\min}$ and $\triangle\phi_{ij}{}^{\max}$ can be precomputed as well-posed optimization problems represented via (47). Then the feasible set of values for ϕ may be defined as

$$\phi_f = \phi \in \mathbb{R}^{n\times n}$$
$$\text{such that } \phi_{ij} \in \left[\bar{\phi}_{ij} + \triangle\phi_{ij}{}^{\min}, \bar{\phi}_{ij} + \triangle\phi_{ij}{}^{\max}\right]. \tag{49}$$

Similarly, for $\theta = B^{-1}(A + \lambda I)$, we can obtain

$$\begin{aligned}\theta &= [\bar{B}^{-1}(\bar{A} + \lambda I + \triangle A)] \\ &\quad - (\bar{B} + \triangle B)^{-1} \triangle B \bar{B}^{-1} (\bar{A} + \lambda I + \triangle A) \\ &= \bar{B}^{-1}(\bar{A} + \lambda I) + \bar{B}^{-1}\triangle A \\ &\quad - (\bar{B} + \triangle B)^{-1} \triangle B \bar{B}^{-1} [\bar{A} + \lambda I + \triangle A] \\ &= \bar{\theta} + \triangle\theta\end{aligned} \tag{50}$$

where $\triangle\theta = \bar{B}^{-1}\triangle A - (\bar{B}+\triangle B)^{-1}\triangle B\bar{B}^{-1}[\bar{A}+\lambda I+\triangle A]$. For any given $\bar{A}$, $\bar{B}$, and $\lambda > 0$ and for uncertainties $\triangle A$ and $\triangle B$ subject to (2) and (3) we can again predetermine

$$\triangle\theta_{ij}{}^{\max} = \max_{\triangle A, \triangle B} [\triangle\theta_{ij}]$$
$$\triangle\theta_{ij}{}^{\min} = \min_{\triangle A, \triangle B} [\triangle\theta_{ij}] \tag{51}$$

such that

$$\theta_{ij} \in \left[\bar{\theta}_{ij} + \triangle\theta_{ij}{}^{\min}, \bar{\theta}_{ij} + \triangle\theta_{ij}{}^{\max}\right]. \tag{52}$$

As before, the feasible set of values for θ can be defined as

$$\theta_f = \left\{\theta \in \mathbb{R}^{n\times n} | \theta_{ij} \in \left[\bar{\theta}_{ij} - \triangle\theta_{ij}{}^{\min}, \bar{\theta}_{ij} + \triangle\theta_{ij}{}^{\max}\right]\right\}. \tag{53}$$

Further, define column vector complements of the matrices ϕ and θ

$$\phi_v = [\phi_{11}, \ldots, \phi_{1n}, \phi_{21}, \ldots, \phi_{2n}, \phi_{n1}, \ldots, \phi_{nn}]^T \tag{54}$$
$$\theta_v = [\theta_{11}, \ldots, \theta_{1n}, \theta_{21}, \ldots, \theta_{2n}, \theta_{n1}, \ldots, \theta_{nn}]^T \tag{55}$$

with each row of θ defined as

$$\chi_k = [\theta_{k1}, \theta_{k2}, \ldots, \theta_{kn}] \text{ for all } k = 1, \ldots, n. \tag{56}$$

Assumption B2: We denote $\tilde{\nu}_s = -\hat{\theta}(0)\mathbf{x}_0$, where the *hat* over the variable denotes its estimated value. From (56), $\tilde{\nu}_{s,k} = -\hat{\chi}_k(0)\mathbf{x}_0$ and for any given $\mathbf{x}_0 \in \mathbb{R}^n$

$$|\tilde{\nu}_{s,k}| = |\hat{\chi}_k(0)\mathbf{x}_0| \tag{57}$$
$$\leq \max_{\theta_f} \|\chi_k\|_1 \|\mathbf{x}_0\|_\infty \tag{58}$$
$$\leq \mu_k \|\mathbf{x}_0\| \tag{59}$$

with $\mu_k = \max_{\theta_f} \|\hat{\chi}_k\|_1$. Assume that for any $\mathbf{x}_0 \in \mathbb{R}^n$, $\mu_k\|\mathbf{x}_0\| \leq \alpha p_2 \mathbf{u}_{m_k}$ for all $k = 1, 2, \ldots, n$. For some p_2 selected such that $0 \ll p_2 < 1$, select the scalar α such that

$$\alpha = \min_k \sqrt{\frac{p_m}{p_M \beta_k} - \frac{\Theta\mu_k}{\gamma p_2^2 p_M \beta_k \mathbf{u}_{m_k}^2}} \tag{60}$$

$$\text{with } \Theta = \sum_{i=1}^{n}\sum_{j=1}^{n} \left(\Delta\theta_{ij}^{\max} + \Delta\theta_{ij}^{\min}\right)^2 \tag{61}$$

$$\text{and } \beta_k = \frac{\max_{\theta_f} \|\chi_k\|_1^2}{\min_{\theta_f} \|\chi_k\|_1^2} \tag{62}$$

for all $k = 1, 2, \ldots, n$. Note that this assumption is more conservative than non-adaptive case.

Following the control law formulation laid out in the previous section, but this time written in terms of θ and ϕ

$$\dot{\mathbf{e}} = -\lambda\mathbf{e} + B(\mathbf{u}_a + \theta\mathbf{x} - \phi(\dot{\mathbf{x}}_r + \lambda\mathbf{x}_\mathbf{r})). \tag{63}$$

For systems with unknown θ and ϕ matrices, the control law is defined as

$$\mathbf{u}_a = \mathbf{u}_t \text{ if } |\mathbf{u}_{t_k}| < \mathbf{u}_{m_k} \text{ and } \mathbf{u}_{m_p} < s_1 \forall\, k = 1, \ldots n$$
$$\text{else,}$$
$$\mathbf{u}_a = \begin{cases} \mathbf{u}_b, & \text{if } s_1 \leq \mathbf{u}_{m_p} < s_2 \\ \mathbf{u}_{t_{\text{sat}}}, & \text{if } s_2 \leq \mathbf{u}_{m_p} < p_1 \\ \mathbf{u}_c, & \text{if } p_1 \leq \mathbf{u}_{m_p} < p_2 \\ \mathbf{u}_s, & \text{if } \mathbf{u}_{m_p} \geq p_2 \end{cases} \tag{64}$$

where

$$\mathbf{u}_t = -\hat{\theta}\mathbf{x} + \hat{\phi}(\dot{\mathbf{x}}_r + \lambda\mathbf{x}_\mathbf{r}) \tag{65}$$
$$\mathbf{u}_s = -\hat{\theta}\mathbf{x} \tag{66}$$
$$\mathbf{u}_{t_{\text{sat}}} = -\hat{\theta}\mathbf{x} + (1 - \mathbf{u}_{m_p})\mathbf{Sat}_{\mathbf{dc}}(\hat{\phi}(\dot{\mathbf{x}}_r + \lambda\mathbf{x}_\mathbf{r})) \tag{67}$$

$\mathbf{u}_b$ and $\mathbf{u}_c$ are given in (30) and (31) and $\hat{\theta}$ and $\hat{\phi}$ are estimated values of θ and ϕ. It is important to point that in this case the definition of $\mathbf{u}_{m_p}$ is revised to

$$\mathbf{u}_{m_p} = \max_k [\mathbf{u}_{p,k}] \tag{68}$$
$$\mathbf{u}_{p_k} = \frac{\|\hat{\chi}_k\|_1 \|\mathbf{x}\|}{\alpha \mathbf{u}_{m_k}} \tag{69}$$

where $k = 1, 2, 3, \ldots, n$. Substituting for the control law defined in (64), the closed-loop error dynamics reduce to

$$\dot{\mathbf{e}} = -\boldsymbol{\lambda}\mathbf{e} - B(\hat{\theta} - \theta)\mathbf{x} + B(\hat{\phi} - \phi)(\dot{\mathbf{x}}_r + \lambda\mathbf{x}_\mathbf{r}) + B\delta \tag{70}$$

where

$$\delta = \mathbf{u}_a - \mathbf{u}_t \tag{71}$$

that is

$$\delta = 0 \text{ if } |\mathbf{u}_{t_k}| < \mathbf{u}_{m_k} \text{and} \mathbf{u}_{m_p} < s_1 \tag{72}$$

$$\begin{aligned}\delta =& (3\xi^2 - 2\xi^3)((1 - \mathbf{u}_{m_p})\mathrm{Sat}_{dc}(\hat{\phi}(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r}))) \\ &- (3\xi^2 - 2\xi^3)\hat{\phi}(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r})\end{aligned}$$

$$\text{if } s_1 \leq \mathbf{u}_{m_p} < s_2, \text{ where } \xi = \frac{\mathbf{u}_{m_p} - s_1}{s_2 - s_1} \tag{73}$$

$$\delta = (1 - \mathbf{u}_{m_p})\mathrm{Sat}_{dc}(\hat{\phi}(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r})) - (\hat{\phi}(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r}))$$

$$\text{if } s_2 \leq \mathbf{u}_{m_p} < p_1 \tag{74}$$

$$\begin{aligned}\delta =& (1 - \mathbf{u}_{m_p})\mathrm{Sat}_{dc}(\hat{\phi}(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r})) - (\hat{\phi}(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r})) \\ &+ (3\xi^2 - 2\xi^3)\left((1 - \mathbf{u}_{m_p})\mathrm{Sat}_{dc}(\hat{\phi}(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r}))\right)\end{aligned}$$

$$\text{if } p_1 \leq \mathbf{u}_{m_p} < p_2, \text{ where } \xi = \frac{\mathbf{u}_{m_p} - p_1}{p_2 - p_1} \tag{75}$$

$$\delta = -(\hat{\phi}(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r})) \text{ if } \mathbf{u}_{m_p} \geq p_2 \tag{76}$$

or

$$\delta = \delta(\hat{\phi}, \mathbf{x}_r, \dot{\mathbf{x}}_r) \text{ for all } \mathbf{u}_{m_p}. \tag{77}$$

In terms of column vector complements defined in (56)–(57), rewrite the closed-loop error dynamics as

$$\dot{\mathbf{e}} = -\boldsymbol{\lambda}\mathbf{e} - B\Lambda_x(\hat{\theta}_v - \theta_v) + B\Omega(\hat{\phi}_v - \phi_v) + B\delta \tag{78}$$

with terms Λ_x and Ω defined such that

$$\Lambda_x(\hat{\theta}_v - \theta_v) = (\hat{\theta} - \theta)\mathbf{x} \tag{79}$$

$$\Omega(\hat{\phi}_v - \phi_v) = (\hat{\phi} - \phi)(\dot{\mathbf{x}}_r + \lambda \mathbf{x_r}). \tag{80}$$

A. Adaptive Laws

To satisfy the given bounds on parameters and avoid parameter drift, the projection scheme from [21] is adopted

$$\hat{\theta}_{v,k} = \Gamma(\theta_{v,k}) \tag{81}$$

$$\hat{\phi}_{v,k} = \Gamma(\phi_{v,k}) \tag{82}$$

where Γ is defined as

$$\Gamma(\bullet) = \begin{cases} \breve{\bullet}, & \text{if } \breve{\bullet} \in (\bullet^{\min}, \bullet^{\max}) \\ \bullet^{\min}, & \text{if } \breve{\bullet} \leq \bullet^{\min} \\ \bullet^{\max}, & \text{if } \breve{\bullet} \geq \bullet^{\max}. \end{cases} \tag{83}$$

The adaptive laws selected to be

$$\begin{aligned}\dot{\breve{\theta}}_v =& \gamma {\Lambda_x}^T \bar{B}^T \mathbf{e}\, \mathrm{sgn}(L) - \gamma\sigma_1(\breve{\theta}_v - \hat{\theta}_v) \text{ if } \mathbf{u}_a \neq \mathbf{u}_s \\ \text{else } \dot{\breve{\theta}}_v =& \gamma {\Lambda_x}^T \bar{B}^T \mathbf{x}\, \mathrm{sgn}(L) - \gamma\sigma_1(\breve{\theta}_v - \hat{\theta}_v)\end{aligned} \tag{84}$$

and if at some $t = t_* \geq 0$, $\mathbf{u}_{m_p}(t_*) = p_2$, $\breve{\theta}(t_*) \neq \hat{\theta}(t_*^-)$, reset $\breve{\theta}(t)$ such that $\breve{\theta}(t_*) = \hat{\theta}(t_*^-)$.

$$\dot{\breve{\phi}}_v = -\gamma\Omega^T \bar{B}^T \mathbf{e}\, \mathrm{sgn}(L) - \gamma\sigma_2(\breve{\phi}_v - \hat{\phi}_v) \tag{85}$$

for any $\sigma_1, \sigma_2 > \lambda/\gamma$ and the adaptive gain γ is chosen to satisfy

$$\gamma > \max_k \left[\frac{\Theta\mu_k}{p_m p_2^2 \mathbf{u}_{m_k}^2}\right]. \tag{86}$$

B. Stability Analysis

To prove stability of the control laws in (64) and the adaptive laws specified in (84)–(85), choose the following Lyapunov function candidate:

$$\begin{aligned}V =& \mathbf{e}^T P \mathbf{e} + \frac{1}{\gamma}\left[\|\breve{\theta}_v - \theta_v\|^2 - \|\breve{\theta}_v - \hat{\theta}_v\|^2\right] \\ &+ \frac{1}{\gamma}\left[\|\breve{\phi}_v - \phi_v\|^2 - \|\breve{\phi}_v - \hat{\phi}_v\|^2\right].\end{aligned} \tag{87}$$

Details of the proof that this Lyapunov function is non-negative are presented in [21]. Now take the time derivative of (87), and noting that the true parameters are constant

$$\begin{aligned}\dot{V} =& -2\lambda \mathbf{e}^T P \mathbf{e} - 2\mathbf{e}^T PB\Lambda_x(\hat{\theta}_v - \theta_v) \\ &+ 2\mathbf{e}^T P\Omega(\hat{\phi}_v - \phi_v) + 2\mathbf{e}^T PB\delta \\ &+ \frac{2}{\gamma}\left[\dot{\breve{\theta}}_v^T(\breve{\theta}_v - \theta_v) - (\dot{\breve{\theta}}_v - \dot{\hat{\theta}}_v)^T(\breve{\theta}_v - \hat{\theta}_v)\right] \\ &+ \frac{2}{\gamma}\left[\dot{\breve{\phi}}_v^T(\breve{\phi}_v - \phi_v) - (\dot{\breve{\phi}}_v - \dot{\hat{\phi}}_v)^T(\breve{\phi}_v - \hat{\phi}_v)\right].\end{aligned} \tag{88}$$

For the case $\mathbf{u}_a \neq \mathbf{u}_s$ substitute for P from (44) and the adaptive laws from (84)–(85)

$$\begin{aligned}\dot{V} =& -2\lambda \mathbf{e}^T P \mathbf{e} + 2\mathbf{e}^T \bar{B}\delta\, \mathrm{sgn}(L) - 2\sigma_1(\breve{\theta}_v - \hat{\theta}_v)^T(\hat{\theta}_v - \theta_v) \\ &- 2\sigma_2(\breve{\phi}_v - \hat{\phi}_v)^T(\hat{\phi}_v - \phi_v).\end{aligned} \tag{89}$$

Using completion of squares

$$\begin{aligned}\dot{V} \leq& -\lambda \mathbf{e}^T P \mathbf{e} + \frac{\|\bar{B}\|^2 {\delta_M}^2}{\lambda p_m} - 2\sigma_1(\breve{\theta}_v - \hat{\theta}_v)^T(\hat{\theta}_v - \theta_v) \\ &- 2\sigma_2(\breve{\phi}_v - \hat{\phi}_v)^T(\hat{\phi}_v - \phi_v)\end{aligned} \tag{90}$$

where $\delta_M = \sup_t \|\delta(t)\|$. Note from (77) $\delta(t)$ is a function of $\hat{\phi}$ that is bounded due to the projection scheme adopted, reference trajectory $\mathbf{x}_r$ and its derivative $\dot{\mathbf{x}}_r$ that are bounded by choice and the parameter λ that is a positive scalar quantity chosen by the designer. By virtue of these signals, it is guaranteed that $\delta(t)$ is bounded for all time and its supremum exists. Furthermore

$$\begin{aligned}\dot{V} \leq& -\lambda \mathbf{e}^T P \mathbf{e} - \frac{\lambda}{\gamma}\left\{\|\breve{\theta}_v - \theta_v\|^2 - \|\breve{\theta}_v - \hat{\theta}_v\|^2\right\} \\ &- \frac{\lambda}{\gamma}\left\{\|\breve{\phi}_v - \phi_v\|^2 - \|\breve{\phi}_v - \hat{\phi}_v\|^2\right\} \\ &+ \frac{\lambda}{\gamma}\left\{\|\breve{\theta}_v - \theta_v\|^2 - \|\breve{\theta}_v - \hat{\theta}_v\|^2\right\} \\ &+ \frac{\lambda}{\gamma}\left\{\|\breve{\phi}_v - \phi_v\|^2 - \|\breve{\phi}_v - \hat{\phi}_v\|^2\right\} \\ &+ \frac{\|\bar{B}\|^2 {\delta_M}^2}{\lambda p_m} - 2\sigma_1(\breve{\theta}_v - \hat{\theta}_v)^T(\hat{\theta}_v - \theta_v) \\ &- 2\sigma_2(\breve{\phi}_v - \hat{\phi}_v)^T(\hat{\phi}_v - \phi_v).\end{aligned} \tag{91}$$

Since $\sigma_1 > \lambda/\gamma$ and $\sigma_2 > \lambda/\gamma$, we have

$$\begin{aligned}\dot{V} \leq& -\lambda V + \frac{\|\bar{B}\|^2 {\delta_M}^2}{\lambda p_m} \\ &+ \frac{\lambda}{\gamma}\sum_{k=1}^{n^2}(2\breve{\theta}_{v,k} - \theta_{v,k} - \hat{\theta}_{v,k})(\hat{\theta}_{v,k} - \theta_{v,k})\end{aligned}$$

$$- \frac{2\lambda}{\gamma}\sum_{k=1}^{n^2}(\hat{\theta}_{v,k}-\theta_{v,k})(\breve{\theta}_{v,k}-\hat{\theta}_{v,k}) + \frac{\lambda}{\gamma}\sum_{k=1}^{n^2}(2\breve{\phi}_{v,k}-\phi_{v,k}-\hat{\phi}_{v,k})(\hat{\phi}_{v,k}-\phi_{v,k}) - \frac{2\lambda}{\gamma}\sum_{k=1}^{n^2}(\hat{\phi}_{v,k}-\phi_{v,k})(\breve{\phi}_{v,k}-\hat{\phi}_{v,k}) \quad (92)$$

or

$$\dot{V} \leq -\lambda V + \frac{\|\bar{B}\|^2 {\delta_M}^2}{\lambda p_m} + \frac{\lambda}{\gamma}\sum_{k=1}^{n^2}(\hat{\theta}_{v,k}-\theta_{v,k})(\breve{\theta}_{v,k}-\theta_{v,k}) - \frac{\lambda}{\gamma}\sum_{k=1}^{n^2}(\hat{\theta}_{v,k}-\theta_{v,k})(\breve{\theta}_{v,k}-\hat{\theta}_{v,k}) + \frac{\lambda}{\gamma}\sum_{k=1}^{n^2}(\hat{\phi}_{v,k}-\phi_{v,k})(\breve{\phi}_{v,k}-\phi_{v,k}) - \frac{\lambda}{\gamma}\sum_{k=1}^{n^2}(\hat{\phi}_{v,k}-\phi_{v,k})(\breve{\phi}_{v,k}-\hat{\phi}_{v,k}). \quad (93)$$

Thus

$$\dot{V} \leq -\lambda V + \frac{\|\bar{B}\|^2 {\delta_M}^2}{\lambda p_m} + \frac{\lambda}{\gamma}\sum_{k=1}^{n^2}\left[(\hat{\theta}_{v,k}-\theta_{v,k})^2 + (\hat{\phi}_{v,k}-\phi_{v,k})^2\right] \quad (94)$$

or

$$\dot{V} \leq -\lambda V + \frac{\|\bar{B}\|^2 {\delta_M}^2}{\lambda p_m} + \frac{\lambda}{\gamma}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\left\{(\theta_{i,j}^{\max}-\theta_{v,k}^{\min})^2 + (\phi_{i,j}^{\max}-\phi_{v,k}^{\min})^2\right\}\right]. \quad (95)$$

Finally, it can be concluded that

$$\dot{V} \leq -\lambda V + \epsilon \quad (96)$$

for some positive constant ϵ. This ensures that the Lyapunov function V is uniformly bounded for the case $\mathbf{u}_a \neq \mathbf{u}_s$.

For the case $\mathbf{u}_a = \mathbf{u}_s$, there is an additional term in (89)

$$\dot{V} = -2\lambda \mathbf{e}^T P\mathbf{e} + 2\mathbf{e}^T\bar{B}\delta\,\mathrm{sgn}(L) - 2\sigma_1(\breve{\theta}_v-\hat{\theta}_v)^T(\hat{\theta}_v-\theta_v) - 2\sigma_2(\breve{\phi}_v-\hat{\phi}_v)^T(\hat{\phi}_v-\phi_v) + 2\mathbf{x}_r^T\bar{B}\Lambda_x(\hat{\theta}_v-\theta_v). \quad (97)$$

Now

$$2\mathbf{x}_r^T\bar{B}\Lambda_x(\hat{\theta}_v-\theta_v) = 2\mathbf{x}_r^T\bar{B}(\hat{\theta}-\theta)\mathbf{x} = 2\mathbf{x}_r^T\bar{B}(\hat{\theta}-\theta)\mathbf{x}_r + 2\mathbf{e}^T(\hat{\theta}-\theta)^T\bar{B}^T\mathbf{x}_r. \quad (98)$$

Define $\rho = \max_{\theta_1,\theta_2\in\theta_f}\|\theta_1-\theta_2\|$ and $x_{rm} = \sup_t\|\mathbf{x}_r(t)\|$. Then

$$\dot{V} \leq -\lambda\mathbf{e}^T P\mathbf{e} - 2\sigma_1(\breve{\theta}_v-\hat{\theta}_v)^T(\hat{\theta}_v-\theta_v) - 2\sigma_2(\breve{\phi}_v-\hat{\phi}_v)^T(\hat{\phi}_v-\phi_v) - \lambda\mathbf{e}^T P\mathbf{e} + 2\rho\|\bar{B}\|x_{rm}^2 + 2\mathbf{e}^T\bar{B}\delta\,\mathrm{sgn}(L) + 2\rho\|\mathbf{e}\|x_{rm}\|\bar{B}\| \quad (99)$$

or

$$\dot{V} \leq -\lambda\mathbf{e}^T P\mathbf{e} - 2\sigma_1(\breve{\theta}_v-\hat{\theta}_v)^T(\hat{\theta}_v-\theta_v) - 2\sigma_2(\breve{\phi}_v-\hat{\phi}_v)^T(\hat{\phi}_v-\phi_v) - \lambda\mathbf{e}^T p_m\|\mathbf{e}\|^2 + 2\rho\|\bar{B}\|x_{rm}^2 + 2\|\mathbf{e}\|\|\bar{B}\|[\delta_M+\rho x_{rm}]. \quad (100)$$

Let $\tilde{\delta}_m = \delta_M + \rho x_{rm}$, and replicate steps (91)–(94), to get

$$\dot{V} \leq -\lambda V + \frac{\|\bar{B}\|^2\tilde{\delta}_m^2}{\lambda p_m} + 2\rho\|\bar{B}\|x_{rm}^2 + \frac{\lambda}{\gamma}\sum_{i=1}^{n}\sum_{j=1}^{n}\left[\left(\theta_{ij}^{\max}-\theta_{ij}^{\min}\right)^2 + \left(\phi_{ij}^{\max}-\phi_{ij}^{\min}\right)^2\right]. \quad (101)$$

Similar to (96), it can be concluded that

$$\dot{V} \leq -\lambda V + \tilde{\epsilon} \quad (102)$$

for some other finite positive constant $\tilde{\epsilon}$. Therefore, from the uniform boundedness theorem, one concludes that $\mathbf{e} \in \mathcal{L}_\infty$, $\breve{\theta} \in \mathcal{L}_\infty$ and $\breve{\phi} \in \mathcal{L}_\infty$, which results in the boundedness of all closed-loop signals.

C. Control Saturation Analysis

The next issue to be analyzed is whether the control signal stays within saturation limits for all time. Suppose that at some time $t = t_* \geq 0$, $\mathbf{u}_{m_p}(t_*) = p_2$ such that, $\mathbf{u}_a(t_*^+) = \mathbf{u}_s(t_*^+)$. In this case

$$\|\hat{\chi}_k(t_*)\|_1^2\|\mathbf{x}(t_*)\|^2 \leq \alpha^2 {p_2}^2 {\mathbf{u}_{m_k}}^2 \quad (103)$$

from the definition of $\mathbf{u}_{m_p}$ in (68).

For time $t \geq t_*$, suppose that $\mathbf{u}_a = \mathbf{u}_s$, then the state evolves according to

$$\dot{\mathbf{x}} = -\lambda\mathbf{x} - B\Lambda_x(\hat{\theta}_v-\theta_v). \quad (104)$$

Consider the Lyapunov function candidate

$$V_s = \mathbf{x}^T P\mathbf{x} + \frac{1}{\gamma}\left[\|\breve{\theta}_v-\theta_v\|^2 - \|\breve{\theta}_v-\hat{\theta}_v\|^2\right]. \quad (105)$$

Notice that the form of (105) is similar to (87), and it can be verified that V_s is positive definite. Next, take the derivative of V_s along the trajectory in (104) to get

$$\dot{V}_s = 2\mathbf{x}^T P\left[-\lambda\mathbf{x} - B\Lambda_x(\hat{\theta}_v-\theta_v)\right] + \frac{2}{\gamma}(\hat{\theta}_v-\theta_v)\dot{\breve{\theta}}_v^T. \quad (106)$$

Substituting the adaptive law for $\dot{\breve{\theta}}_v$ in (84)

$$\dot{V}_s = -2\lambda\mathbf{x}^T P\mathbf{x} - 2\mathbf{x}^T\bar{B}\Lambda_x(\hat{\theta}_v-\theta_v)\mathrm{sgn}(L) + \frac{2}{\gamma}(\hat{\theta}_v-\theta_v)\left[\gamma{\Lambda_x}^T\bar{B}^T\mathbf{x}\,\mathrm{sgn}(L) - \gamma\sigma_1(\breve{\theta}_v-\hat{\theta}_v)\right]^T \quad (107)$$

$$\dot{V}_s = -2\lambda\mathbf{x}^T P\mathbf{x} - 2\sigma_1(\breve{\theta}_v-\hat{\theta}_v)^T(\hat{\theta}_v-\theta_v) \leq 0. \quad (108)$$

Thus, one can conclude $V_s(t) \leq V_s(t_*)$ for $t \geq t_*$. Further

$$p_m \|\mathbf{x}(t)\|^2 \leq \mathbf{x}^T(t) P \mathbf{x}(t) \leq V_s(t) \leq V_s(t_*). \tag{109}$$

But

$$\begin{aligned} V_s(t_*) &= \mathbf{x}^T(t_*) P \mathbf{x}(t_*) + \frac{1}{\gamma} \|\hat{\theta}_v - \theta_v\|^2 \\ V_s(t_*) &\leq p_M \|\mathbf{x}(t_*)\|^2 + \frac{\Theta}{\gamma}. \end{aligned} \tag{110}$$

Combining (109) and (110)

$$\begin{aligned} \|\mathbf{x}(t)\|^2 &\leq \frac{p_M}{p_m} \|\mathbf{x}(t_*)\|^2 + \frac{\Theta}{\gamma p_m} \\ &\leq \frac{p_M}{p_m} \frac{1}{\min_{\theta_f} \|\chi_k\|_1^2} \|\chi_k(t_*)\|_1{}^2 \|\mathbf{x}(t_*)\|^2 + \frac{\Theta}{\gamma p_m}. \end{aligned} \tag{111}$$

From (103)

$$\|\mathbf{x}(t)\|^2 \leq \frac{p_M}{p_m} \frac{1}{\min_{\theta_f} \|\chi_k\|_1^2} \alpha^2 p_2^2 \mathbf{u}_{m_k}^2 \|\mathbf{x}(t_*)\|^2 + \frac{\Theta}{\gamma p_m}. \tag{112}$$

Finally, from the definition of $\mathbf{u}_s$

$$|\mathbf{u}_{s,k}(t)|^2 \leq \|\hat{\chi}_k(t)\|_1^2 \|\mathbf{x}(t)\|^2 \leq \max_{\theta_f} \|\chi_k\|_1^2 \|\mathbf{x}(t)\|^2. \tag{113}$$

From (112), (62), and the choices of γ in (86) and α in (60)

$$\begin{aligned} |\mathbf{u}_{s,k}(t)|^2 &\leq \frac{p_M}{p_m} \beta_k \alpha^2 p_2^2 \mathbf{u}_{m_k}^2 + \frac{\Theta \mu_k}{\gamma p_m} && (114) \\ &\leq p_2^2 \mathbf{u}_{m_k}^2. && (115) \end{aligned}$$

Thus, every kth component of the control signal stays inside the bounds

$$-p_2 \mathbf{u}_{m_k} \leq \mathbf{u}_{s,k} \leq p_2 \mathbf{u}_{m_k} \tag{116}$$

for all $t \geq t_*$ such that $\mathbf{u}_a(t) = \mathbf{u}_s(t)$. Therefore, it is guaranteed that on enforcing the control saturation condition for the adaptive case, the control signal stays within the specified limits.

VII. Numerical Examples

A. Purpose and Scope

Validation of the theoretical developments presented above is demonstrated in this section through simulation. The examples demonstrate the direction consistent mechanism for two unstable systems. The first example is a generic second-order plant. The tracking results are studied for two sinusoidal trajectories of different magnitudes. The purpose of this example is to simulate the response of the closed-loop system for two cases; one with tracking control within bounds and the other with tracking control outside control limits. Whenever the tracking control is within bounds, it is expected that the system demonstrates perfect tracking. Direction consistency is demonstrated for reference trajectories that require more control effort than that available.

The next simulation develops and evaluates control laws for a lateral/directional linear model representative of the F-16XL aircraft. This example demonstrates that the control laws developed earlier are also applicable to systems of the form

$$m\ddot{\mathbf{x}} + c\dot{\mathbf{x}} + k\mathbf{x} = 0. \tag{117}$$

This example presents the necessary equations for implementing control for kinematic tracking. The motive of this example is to compare the response of the system with and without the switching control law. It is demonstrated that without implementation of the switching control law the system goes unstable, while with the switching mechanism the plant state remains within bounds and consistent with the reference.

B. Second-Order Unstable Plant With Unknown Parameters

This example demonstrates the concept of a direction consistent constraint mechanism for an unstable second-order plant. The control objective is to restrict the states of the system to follow specified sinusoidal trajectories that are out-of-phase. The nominal plant used in the simulation is specified as

$$\bar{A} = \begin{bmatrix} 0.2 & 0.1 \\ 0.1 & -0.5 \end{bmatrix} \tag{118}$$

$$\bar{B} = \begin{bmatrix} 1 & 3 \\ -2 & 1 \end{bmatrix}. \tag{119}$$

The true plant matrices are randomly generated with an uncertainty of 5% in each element of $\bar{A}$ and $\bar{B}$ for the simulation. The control vector is symmetrically bounded between $[\pm 1 \quad \pm 1]^T$.

1) Case 1(a): The peak-to-peak amplitudes of reference for both the states is chosen to be 2. The frequency of oscillation for the specified reference is 1 rad/s and the two references are out-of-phase. λ is chosen as 0.2. The other design variables are $\alpha = 0.8856$, $s_1 = 0.97$, $s_2 = 0.98$, $p_1 = 0.985$, and $p_2 = 0.99$, with the adaptive gains chosen to be $\gamma = 1$, $\sigma_1 = 2$, and $\sigma_2 = 0.2$. The initial conditions of the system state is chosen as $x(0) = [0.02 \quad 0.02]^T$. Figs. 4 and 5 present results for this case. The plant state in this case perfectly follows the reference and $\mathbf{u}_{m_p}$ always remains less than unity. Moreover, since the tracking control computed using dynamic-inversion always remains within bounds, the applied control smoothly follows it. Further, the adaptive parameters are seen to remain within pre-computed bounds. Since the input is not sufficiently rich, the adaptive parameters shown in Fig. 5 do not converge to true values.

2) Case 1(b): In this case the peak amplitudes of the reference is increased to observe direction consistency. The amplitudes are chosen as $\mathbf{x}_{ref_1} = 1.5$ and $\mathbf{x}_{ref_2} = 2$ with same frequency of oscillations as in the previous case. Figs. 6 and 7 present the results. The plant state remains bounded and direction consistent with the desired reference when switching control is implemented. With the switching control law and direction consistent mechanism the applied control smoothly switches from tracking control to the saturated control and is direction consistent with the desired tracking control, without

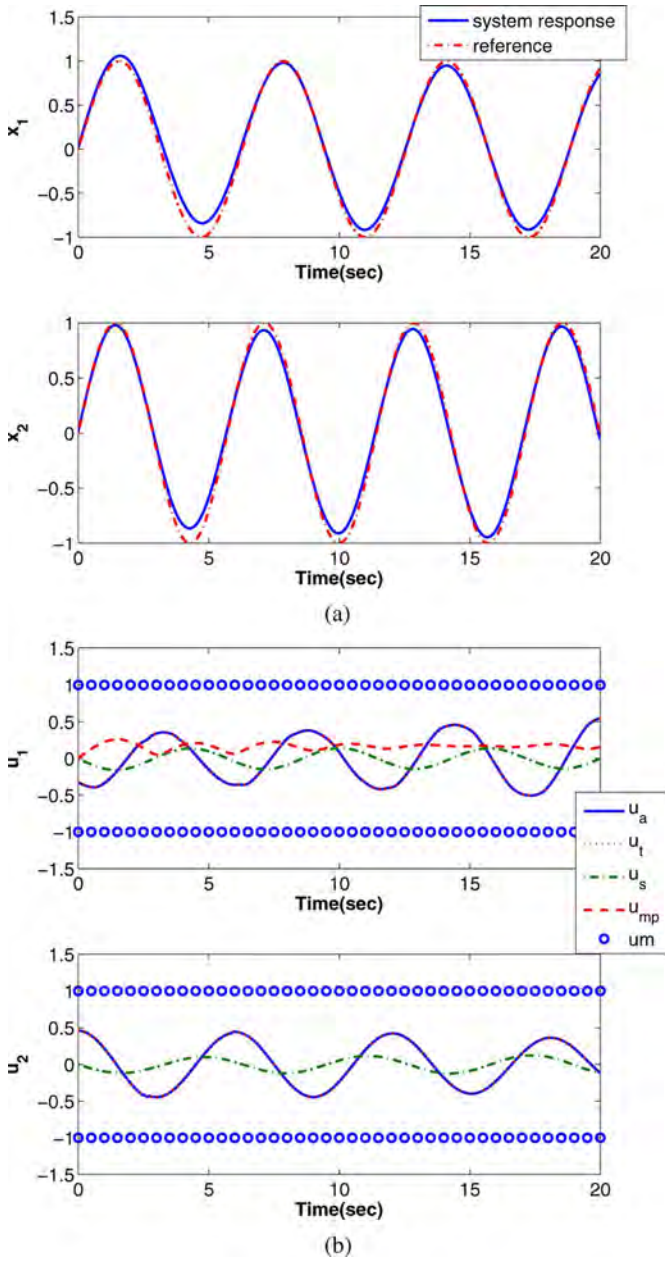


Fig. 4. Case 1(a) plant and control time histories. (a) Case 1(a): Plant State. (b) Case 1(a): Applied Control.

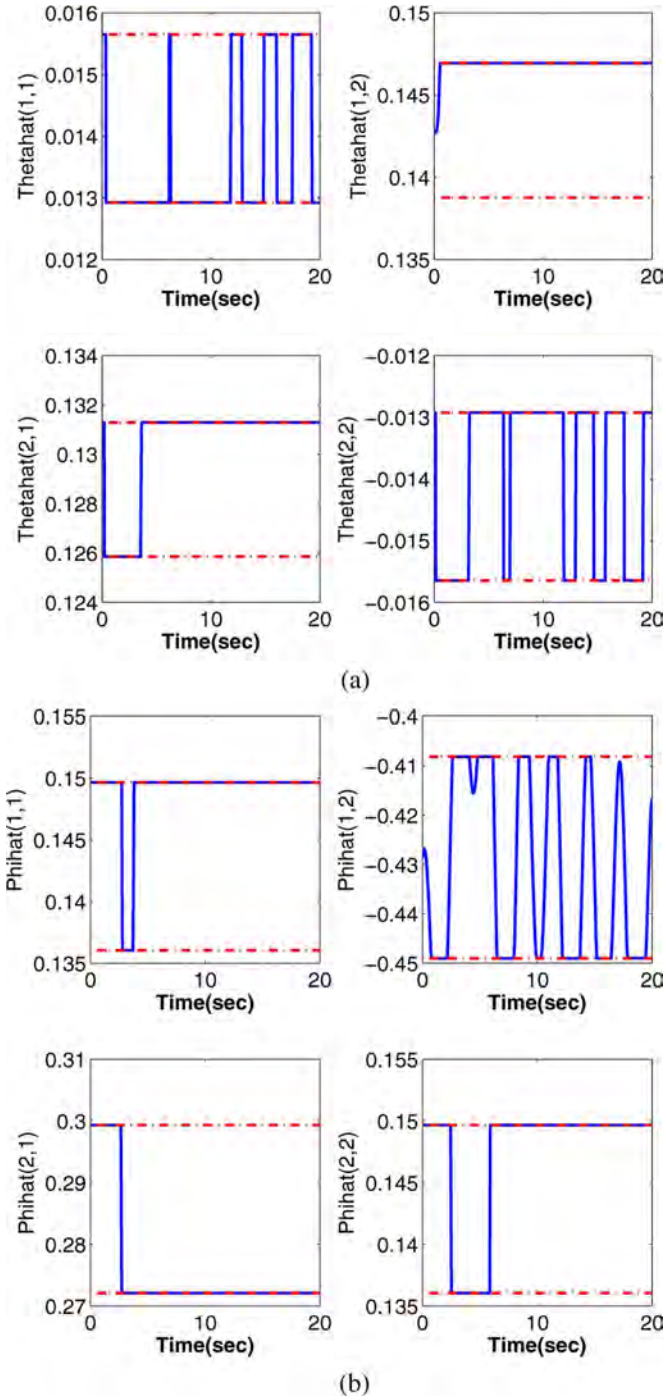


Fig. 5. Case 1(a): adaptive parameter time histories. (a) Case 1(a): Adaptive Parameter $\hat{\theta}$. (b) Case 1(a): Adaptive Parameter $\hat{\phi}$.

any control chattering as seen in Fig. 6(b). Also observe that since $\mathbf{u}_s$ remains within bounds, the ratio u_{mp} always remains less than 1.

Fig. 7 show the update of the adaptive parameters $\hat{\theta}$ and $\hat{\phi}$. The parameter projection successfully restricts the adaptive parameters within the parameter bounds. The adaptive parameters do not show any definite trend in the update. The important thing is to note that parameter convergence to a constant was demonstrated even in presence of errors due to control saturation.

C. F-16XL Aircraft

The objective is to command an aggressive maneuver which will saturate the controls. Using the F-16XL (see Fig. 8), the commanded maneuver is a bank angle doublet of $\pm$ 60 deg while simultaneously turning through a heading angle of -20 deg. The control effectors used here are aileron δ_{AA} and differential elevon δ_{EA}. While rudder is available as a control effector, it is not used here for the manueuver which constists primarily of rolling. The F-16XL linear model is displayed in the Appendix. All states and controls are perturbations from the steady, level, 1-g trimmed flight states given in Table I. The open-loop eigenvalues are $\lambda_{1,2} = -0.321 \pm 3.609j$, $\lambda_3 = -1.0138$, and $\lambda_4 = -0.032$.

1) Controller Synthesis: A reduced-order linear model is used to develop the controller. For the strictly lateral/directional

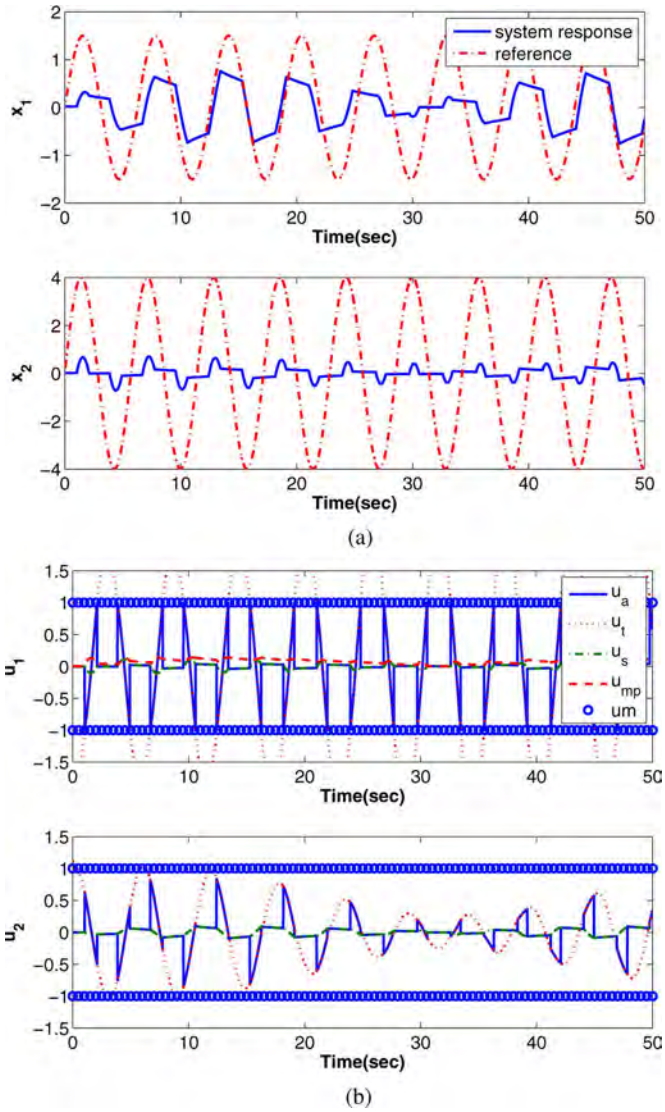


Fig. 6. Case 1(b) state and control time histories. (a) Case 1(b): Plant State. (b) Case 1(b): Computed Control.

maneuver performed here, the longitudinal dynamics are neglected. The model is cast into a structured form as a kinematic part and a dynamic part using only the roll rate and yaw rate states. However, the full model is used for simulation.

Kinematic part

$$\dot{\phi} = p \tag{120}$$

$$\dot{\psi} = r. \tag{121}$$

Dynamic part

$$\begin{bmatrix} \dot{p} \\ \dot{r} \end{bmatrix} = \bar{A} \begin{bmatrix} p \\ r \end{bmatrix} + \bar{B} \begin{bmatrix} \delta_{AA} \\ \delta_{EA} \end{bmatrix} \tag{122}$$

where ϕ is the bank angle, ψ is the heading angle, and p and r are the roll and yaw rates, respectively. The control effectors are limited to maximum position limits of $\pm$ 25 deg. Uncertainty in the aircraft dynamical model is addressed by randomly introducing errors into the stability and control derivatives during numerical simulation. The reference trajectory is specified in

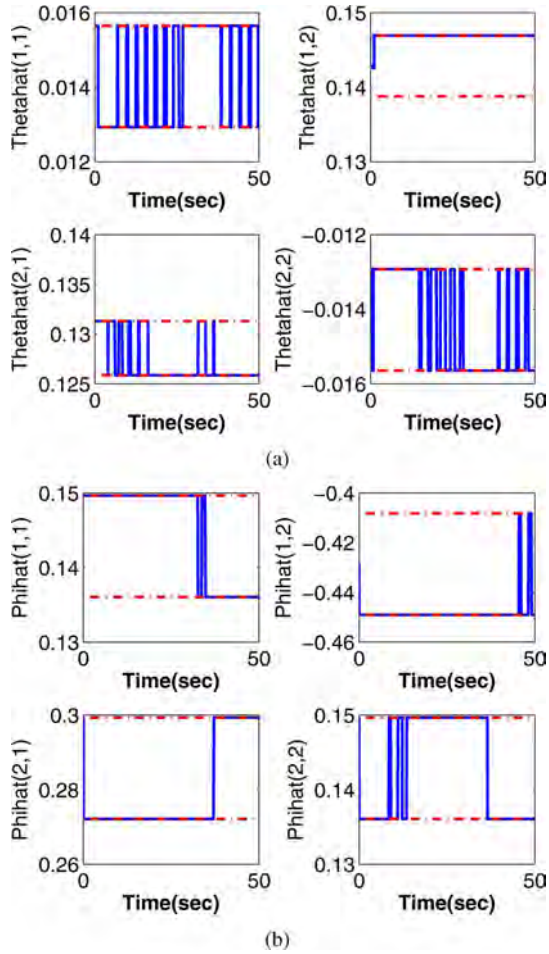


Fig. 7. Case 1(b) adaptive parameter time histories. (a) Case 1(b): Adaptive Parameter $\hat{\theta}$. (b) Case 1(b): Adaptive Parameter $\hat{\phi}$.

Fig. 8. F-16XL external physical characteristics.

TABLE I
TRIM STATE

Parameter	Value
Mach	0.90
Altitude	25,000 ft
Angle-of-Attack	4.61 degrees
Aileron δ_{AA}	0 degrees
Differential Elevon δ_{EA}	0 degrees

terms of ϕ_r, ψ_r, p_r, r_r, $\dot{p}_r$, and $\dot{r}_r$. The control law and the update laws for the adaptive parameters are developed in accordance with the theory developed in the earlier sections. For

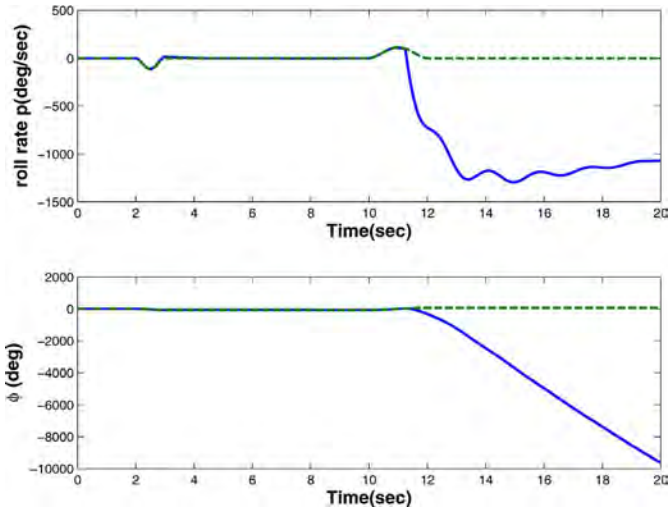


Fig. 9. Case 2(a) Roll rate and Bank Angle for F-16XL.

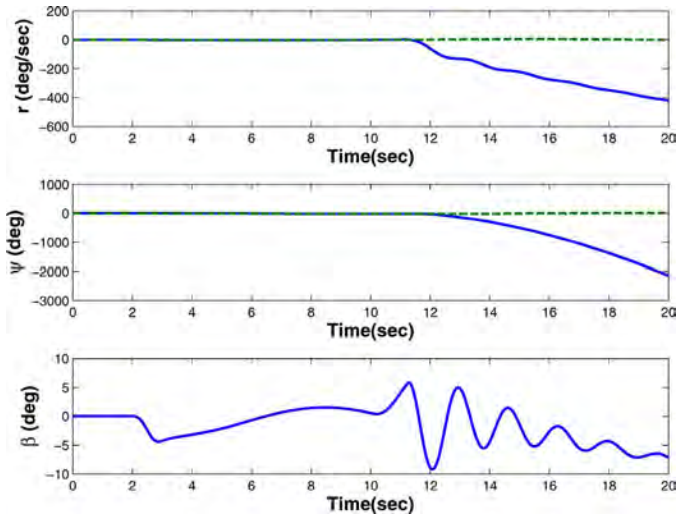


Fig. 10. Case 2(a) Yaw rate, heading angle, and sideslip angle for F-16XL.

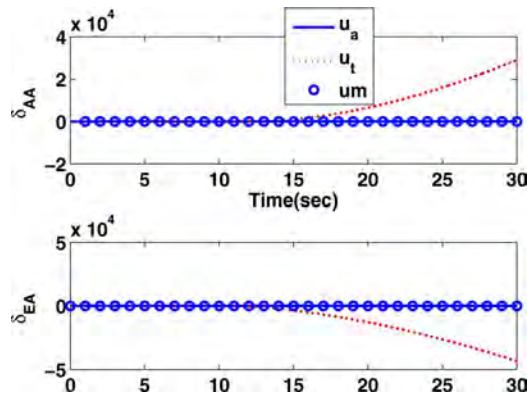


Fig. 11. Case 2(a) Aileron and Differntial Elevon for F-16XL.

brevity only the equations required for incorporating the control law in the simulation are presented here. The tracking errors are defined as

$$e_1 = \phi - \phi_r \tag{123}$$

$$e_2 = \psi - \psi_r. \tag{124}$$

To track the kinematic angles, the closed-loop dynamics are specified as

$$\begin{bmatrix} \dot{e_1} \\ \dot{e_2} \end{bmatrix} + \lambda \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} + K_d \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \mathbf{0}. \tag{125}$$

The design parameters are λ, K_d, γ, σ_1, σ_2, p_1, p_2, s_1, and s_2. The tracking and saturated control are

$$\mathbf{u}_t = -\hat{\theta} \begin{bmatrix} p \\ r \end{bmatrix} + \hat{\Phi}\vartheta \tag{126}$$

$$\mathbf{u}_s = -\hat{\theta} \begin{bmatrix} p \\ r \end{bmatrix} + (1 - u_{mp})\mathrm{Satdc}(\hat{\Phi}\vartheta) \tag{127}$$

where

$$\vartheta \triangleq \begin{bmatrix} \dot{p}_r + \lambda p_r - K_d e_1 \\ \dot{r}_r + \lambda r_r - K_d e_2 \end{bmatrix}. \tag{128}$$

The adaptive laws for the elements of $\hat{\theta}$ and $\hat{\Phi}$ are given by (84)–(85) with

$$\mathbf{e} = \begin{bmatrix} p - p_r \\ r - r_r \end{bmatrix} \tag{129}$$

and

$$\Lambda_x(\hat{\theta}_v - \theta_v) = (\hat{\theta} - \theta) \begin{bmatrix} p \\ r \end{bmatrix} \tag{130}$$

$$\Omega(\hat{\Phi}_v - \Phi_v) = (\hat{\Phi} - \Phi)\vartheta. \tag{131}$$

2) Results and Discussion: Case 2(a) No Switching Control Law For this case, once the control saturates maximum control is applied until the tracking control falls back into limits. Figs. 9–11 show that for the first 10 s the tracking control stays within bounds and the state is bounded. After 10 s, the tracking control required is large and the plant state diverges away from the reference. Notice that without the switching control law the system becomes unstable.

Case 2(b) Switching Control Law In this case the switching control law is implemented. The design constants are chosen as $\gamma = 80$ which gives the value of $\alpha = 0.8582$, $\sigma_1 = 10$, $\sigma_2 = 10$, $s_1 = 0.9$, $s_2 = 0.92$, $p_1 = 0.96$, $p_2 = 0.98$, $\lambda = 10$, and $K_d = 10$. Figs. 12–16 present the simulation results. At 2 s the aircraft is commanded to roll at 112 deg/s to an angle of -60 deg and simultaneously turning to a heading of -20 deg at a yaw rate of 4.2 deg/s. Notice that the tracking control required to perform the maneuver is beyond the position limits specified, so the saturated control is applied since $u_{mp} = 0.9081$. This is greater than the specified s_1. The effect of applying this control is that the aircraft performs the roll at a reduced rate of 80 deg/s. At 2.7 s the tracking control lies within limits and the applied control stays there afterwards. The bank and heading angles settle down to their respective desired values at 10 s.

After 10 s the reference trajectory changes direction yet the system responds accordingly and starts to track closely. The aircraft is commanded to bank 60 deg at the rate of 110 deg/s, while simultaneously turning to 3.5 deg at a yaw rate of 5.4 deg/s. Since the tracking control is outside position limits, the saturated control is applied. As before the roll is performed at a reduced rate of 67.5 deg/s. The bank and heading angles settle to

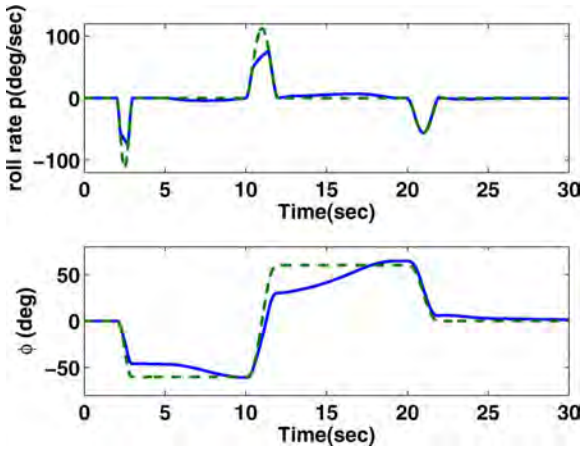


Fig. 12. Case 2(b) Roll rate and bank angle of F-16XL.

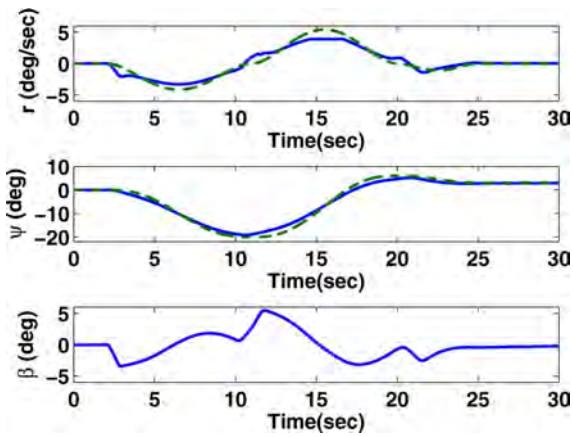


Fig. 13. Case 2(b) Yaw rate, heading, and sidelslip for F-16XL.

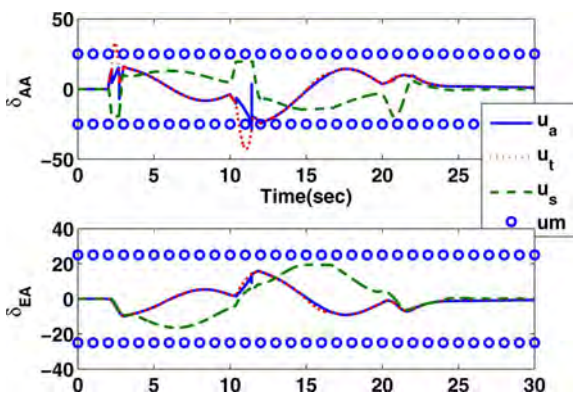


Fig. 14. Case 2(b) Aileron and differential elevon for F-16XL.

their respective values at 20 s, after which the aircraft is commanded to return to wings level, i.e., a 0 deg bank. This is commanded at a rate of 55 deg/s and the tracking control is applied within position limits. It is important to note that during this 30 s time span consistency with the reference is preserved, and control chattering is avoided. Even though sideslip angle β is not directly controlled, it remains within bounds and well behaved throughout the manuever. This is because the system is linear and minimum-phase so the internal dynamics are stable. Further, throughout the maneuver u_{mp} is maintained less than

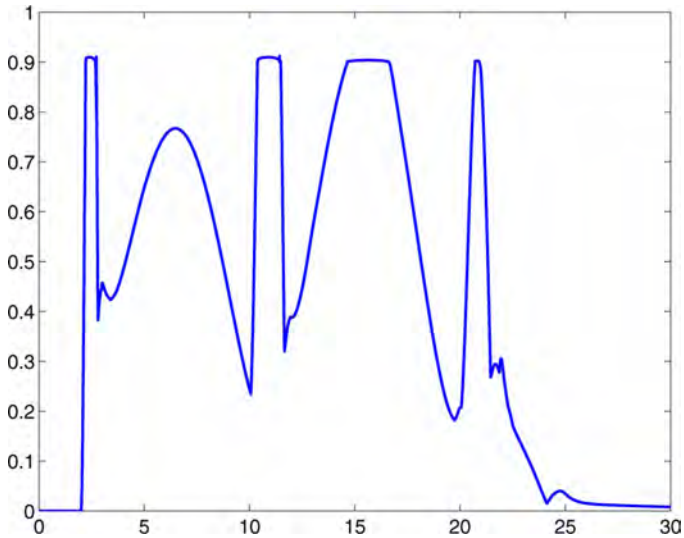


Fig. 15. Case 2(b) $\mathbf{u}_{mp}$ for F-16XL.

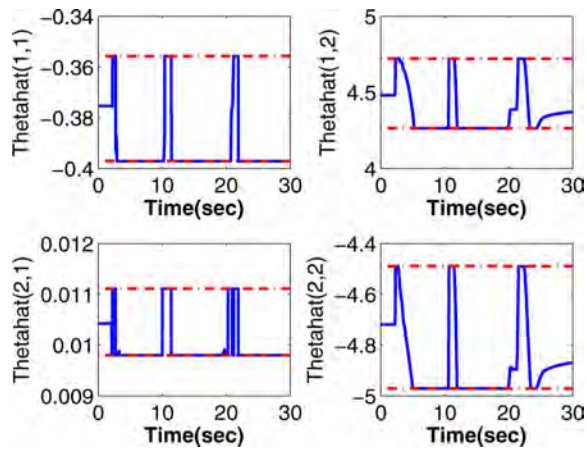


Fig. 16. Case 2(b) Adaptive parameters $\hat{\theta}$ for F-16XL.

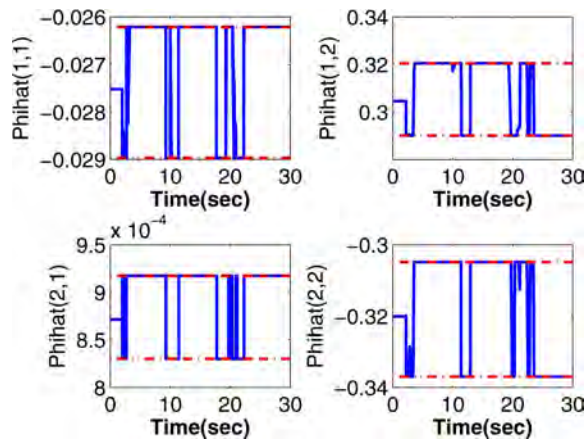


Fig. 17. Case 2(b) Adaptive parameter $\hat{\Phi}$ for F-16XL.

1. The adaptive parameters do not converge to their true values within the duration of the maneuver, because the reference trajectory is not persistently exciting. However, this is immaterial as asymptotic trajectory tracking can be achieved irrespective of parameter convergence.

VIII. Conclusion

Based on the stability proofs and the simulation results presented in the paper, if the control is unsaturated the tracking error asymptotically goes to zero, and all signals in the control scheme are bounded. If the control is saturated, the plant state can only be guaranteed to be bounded and direction consistent with the desired reference. The switching control strategy successfully restricts the state within the DCA. The transition between the tracking control and the stability control is smooth, and the applied control does not show any chattering. The instability protection switching control law can be implemented without prior explicit identification and bookkeeping of the DCA boundary. The control laws developed in this paper are applicable for unstable controllable linear time-invariant square non-singular systems with uncertainty within $\pm 5\%$ of the known nominal model.

Appendix

The F-16XL aircraft lateral/directional model

$$\begin{bmatrix} \dot{\beta} \\ \dot{p} \\ \dot{r} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -0.128 & 0.0803 & -0.997 & 0.035 & 0 \\ -62.867 & -1.306 & 0.296 & 0 & 0 \\ 8.277 & 0.00472 & -0.259 & 0 & 0 \\ 0 & 1 & 0.0806 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \beta \\ p \\ r \\ \phi \\ \psi \end{bmatrix} + \begin{bmatrix} 0.00746 & 0.0344 \\ -37.467 & -35.645 \\ -0.102 & -3.221 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_{AA} \\ \delta_{EA} \end{bmatrix}. \quad (132)$$

All angular quantities are in radians.

Acknowledgment

The authors would like to thank Dr. M. D. Tandale for insightful discussions and comments about this work.

References

[1] T. Hu and Z. Lin, *Control Systems with Actuator Saturation: Analysis and Design*, 1st ed. New York: Birkhauser (Springer), 2001.

[2] T. Hu, L. Qiu, and Z. Lin, "Controllability and stabilization of unstable lti systems with input saturation," in *Proc. IEEE Conf. Decision Control*, 1997, pp. 4498–4503.

[3] T. Hu, Z. Lin, and L. Qiu, "An explicit description of null controllable regions of linear systems with saturating actuators," *Syst. Control Lett.*, vol. 47, no. 1, pp. 65–78, Sep. 2002.

[4] D. Bernstein and A. A. Michel, "Chronological bibliography on saturating actuators," *Int. J. Robust Nonlinear Control*, vol. 5, pp. 375–380, 1995.

[5] S. P. Karason and A. M. Annaswamy, "Adaptive control in the presence of input constraints," *IEEE Trans. Autom. Control*, vol. 39, no. 11, pp. 2325–2330, Nov. 1994.

[6] M. R. Akella, J. L. Junkins, and R. D. Robinett, "Structured model reference adaptive control with actuator saturation limits," in *Proc. AIAA/AAS Astrodynam. Specialist Conf. Exhibit, Collection of Tech. Papers (A98-37348)*, 1998, pp. 10–13.

[7] E. N. Johnson and A. J. Calise, "Limited authority adaptive flight control for reusable launch vehicles," *J. Guid. Control Dyn.*, vol. 26, pp. 906–913, 2003.

[8] E. Lavretsky and N. Hovakimyan, "Positive μ-modification for stable adaptation in the presence of input constraints," in *Proc. Amer. Control Conf.*, 2004, pp. 2545–2550.

[9] D. Li, N. Howakimyan, and C. Cao, "$\mathcal{L}_1$ adaptive controller in the presence of input saturation," presented at the AIAA Guid., Nav., Control Conf. Exhibit, Chicago, IL, 2009.

[10] Y. Hong and B. Yao, "A globally stable high-performance adaptive robust control algorithm with input saturation for precision motion control of linear motor drive systems," *IEEE/ASME Trans. Mechatron.*, vol. 12, no. 2, pp. 198–207, 2007.

[11] N. E. Kahveci and P. A. Ioannou, "Indirect adaptive control for systems with input rate saturation," presented at the Amer. Control Conf., Seattle, WA, 2008.

[12] A. Leonessa, W. M. Haddad, T. Hayakawa, and Y. Morel, "Adaptive control for nonlinear uncertain systems with actuator amplitude and rate saturation constraints," *Int. J. Adapt. Control Signal Process.*, vol. 23, pp. 73–96, 2009.

[13] D. Enns, D. Bugajski, R. Hendrick, and G. Stein, "Dynamic inversion: An evolving methodology for flight control design," *Int. J. Control*, vol. 59, no. 1, pp. 71–91, Jan. 1994.

[14] J. Georgie and J. Valasek, "Evaluation of longitudinal desired dynamics for dynamic-inversion controlled generic reentry vehicles," *J. Guid., Control, Dyn.*, vol. 26, no. 5, pp. 811–819, Sep. 2003.

[15] D. Ito, D. T. Ward, and J. Valasek, "Robust dynamic inversion controller design and analysis for the x-38," presented at the AIAA Guid., Nav., Control Conf. Exhibit, Montreal, QC, Canada, 2001, AIAA-2001-4380.

[16] D. Ito, J. Georgie, J. Valasek, and D. T. Ward, "Re-entry vehicle flight control design guidelines, dynamic inversion," Final Tech. Rep., NASA TP-2002-210771, 2002.

[17] D. B. Doman and A. D. Ngo, "Dynamic inversion based adaptive/reconfigurable control of the x-33 on ascent," *J. Guid., Control, Dyn.*, vol. 25, no. 2, pp. 275–284, Mar. 2002.

[18] W. C. Durham, "Constrained control allocation," *J. Guid., Control, Dyn.*, vol. 16, no. 4, pp. 717–725, Jul. 1993.

[19] M. Tandale and J. Valasek, "Adaptive dynamic inversion control with actuator saturation constraints applied to tracking spacecraft maneuvers," *J. Astronaut. Sci.*, vol. 52, no. 4, pp. 517–530, 2005.

[20] P. J. Antsaklis and A. N. Michel, *A Linear Systems Primer*. New York: Birkhauser Boston, 2007.

[21] R. Bakker and A. Annaswamy, "Stability and robustness properties of a simple adaptive controller," *IEEE Trans. Autom. Control*, vol. 41, no. 9, pp. 1352–1358, Sep. 1996.

John Valasek (S'89–M'95–SM'98) received the B.S. degree in aerospace engineering from California State Polytechnic University, Pomona, in 1986 and the M.S. degree (with honors) and the Ph.D. degree in aerospace engineering from the University of Kansas, Lawrence, in 1991 and 1995, respectively.

From 1985 to 1988, he was a Flight Control Engineer with the Flight Controls Research Group, Aircraft Division, Northrop Corporation, where he worked on the AGM-137 Tri-Services Standoff Attack Missile (TSSAM) Program. He previously held an academic appointment with Western Michigan University during 1995–1997 and was a Summer Faculty Researcher with NASA Langley in 1996 and an AFOSR Summer Faculty Research Fellow with the Air Force Research Laboratory in 1997. Since then, he has been with the Department of Aerospace Engineering, Texas A&M University, College Station, where he is currently Professor of aerospace engineering and Director of the Vehicle Systems and Control Laboratory. He teaches courses on digital control, nonlinear systems, vehicle management systems, cockpit systems and displays, and flight mechanics. His current research interests include machine learning and multi-agent systems, intelligent autonomous control, vision-based navigation systems, and fault-tolerant adaptive control.

Prof. Valasek is a member of the Control Systems Society, the Systems, Man, and Cybernetics Society, the Computational Intelligence Society, and the Education Society. He was formerly an Associate Editor for dynamics and control of the IEEE Transactions on Education (1998–2001).

Maruthi Ram Akella is an Associate Professor with the Aerospace Engineering and Engineering Mechanics Department, The University of Texas at Austin. He has broad interests in dynamical systems and control theory for aero-mechanical systems. His theoretical contributions have found applications in astrodynamics and in the control of space systems and vision-guided robotics. His current research encompasses control theoretic studies of cooperating sensor and robotic agents accounting for measurement time-delays, kinematic constraints, and actuator saturations.

Prof. Akella is an Associate Fellow of the American Institute of Aeronautics and Astronautics (AIAA) and he currently serves as an Associate Editor for the IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS and the *Journal of Guidance, Control, and Dynamics*.

Anshu Siddarth (S'04) is originally from Hyderabad, India. She received the B.Tech. degree in electrical and electronics engineering (with honors) from Jawaharlal Nehru Technological University, Hyderabad, India, in 2006, and the Master of Science degree in aerospace engineering from the Indian Institute of Technology, Madras (IITM), India, in 2008. She is currently pursuing the Ph.D. degree from the Aerospace Engineering Department, Texas A&M University, College Station.

Her current research interests include adaptive control, nonlinear control and singularly perturbed systems.

Ms. Siddarth was a recipient of the "Academic Proficiency Award" in 2006, the "Tata Consultancy Services (TCS)-IITM" "best-in-class" fellowship in 2007, and the Amelia Earhart Fellowship for the 2011–2012 academic year.

Elizabeth Rollins (S'10) received the Magna Cum Laude with a B.S. degree in aerospace engineering from the University of Notre Dame, Notre Dame, IN, in 2007 and the S.M. degree in aeronautics and astronautics from Massachusetts Institute of Technology, Cambridge, in 2009. Currently, she is pursuing the Ph.D. degree in aerospace engineering from Texas A&M University, College Station.

She did research as a Draper Fellow at the Charles Stark Draper Laboratory during her time at MIT. As a member of the Vehicle Systems and Controls Laboratory, Texas A&M University, her current research focus is in the area of control theory and reinforcement learning.

Ms. Rollins was the recipient of an NSF Graduate Research Fellowship (2007). She is a member of Sigma Gamma Tau and Tau Beta Pi, and she was the recipient of the Sigma Gamma Tau Undergraduate Award in the Great Lakes Region (2007).

51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition
07 - 10 January 2013, Grapevine (Dallas/Ft. Worth Region), Texas

AIAA 2013-0875

Approximation of Agent Dynamics Using Reinforcement Learning

Kenton Kirkpatrick* and John Valasek†

Texas A&M University, College Station, Texas 77843-3141

Reinforcement Learning for control of dynamical systems is popular due to the ability to learn control policies without requiring a model of the system being controlled. It can be difficult to learn ideal control policies because it is common to abstract out or ignore completely the dynamics of the agents in the system. In this paper, Reinforcement Learning-based algorithms are developed for learning agents' time dependent dynamics while also learning to control them. Three algorithms are introduced. Sampled-Data Q-learning is an algorithm that learns the optimal sample time for controlling an agent without a prior model. First-Order Dynamics Learning is an algorithm that determines the proper time constants for agents known to have first-order dynamics, while Second-Order Dynamics Learning is an algorithm for learning natural frequencies and damping ratios of second-order systems. The algorithms are demonstrated with numerical simulation. Results presented in this paper show that the algorithms are able to determine information about the system dynamics without resorting to traditional system identification.

I. Introduction

In recent years, Reinforcement Learning (RL) has been an extensively investigated area of research in the field of Machine Learning. It has been a popular tool for solving problems such as dynamical system control, gain scheduling, maze navigation, and game playing. There has been wide success in many of the applications, but researchers have often encountered problems when casting the control of dynamical systems as an RL problem. The state-to-action mapping provided by RL techniques makes the use for control problems attractive, and this is especially the case with Q-learning due to its proven convergence to optimality. RL methods like Q-learning are appealing because they can achieve this mapping experimentally without the need of a model. Although this is indeed the case, implementing these in practice has proven to be very difficult.

Studying the problems associated with this implementation reveals that often the failure to implement RL methods in dynamical systems are not caused but the basic approach of methods like Q-learning. Typically, the problem is either a failure in properly representing the problem or inaccurate function approximation. When choosing to implement the popular Watkins' Q-learning in a dynamical system scenario, it is necessary to realize that the algorithm does not explicitly account for time. Time dependency is often either overlooked or handled outside of the learning process, but accounting for time in the selection of actions (or control) is needed. For instance, when handling sampled-data systems, small changes to sample time can cause drastic changes to the stability of the control policy determined.

One research area that has recently received a lot of attention has been the control of cooperative multi-agent systems through the use of Q-learning-based algorithms. Learning to control multiple agents for the purpose of cooperatively achieving a specified goal is an appealing research topic with high complexity. Some research in this area has involved comparing the effects using Q-learning-based methods to determine joint action selection between different agents to the learning of agent actions independently.[1] Other research has investigated stochastic game extensions to this and treated the system as non-cooperative by having agents

*Graduate Research Assistant, Vehicle Systems & Control Laboratory, Aerospace Engineering Department. Student Member AIAA. *kentonkirk@gmail.com*

†Professor and Director, Vehicle Systems & Control Laboratory, Aerospace Engineering Department. Associate Fellow AIAA. *valasek@tamu.edu*. *Website: vscl.tamu.edu/valasek*

consider only themselves with no knowledge of the existence of other agents.[2] Systems of agents that need to coordinate their actions without knowledge of each other's actions has been determined to be an important area of research for the general application to systems of agents that do not have the ability to communicate with one another.[3,4] Even so, other research has been conducted involving the improvement of Q-learning approaches for determining joint actions through the use of Bayesian inference to estimate strategies.[5]

In most of the research scenarios discussed above, multiple agents are simulated in games that have no dependence on time. Time-dependent agent dynamics cause a fundamental change to the system, and considering the control of time dependencies in multi-agent systems has received little attention. This may be due to the fact that it is difficult to learn control policies for a single time-dependent agent using RL approaches. To address this, a topic of research that needs to be investigated is the learning of an optimal sample time for a sampled-data system. Controlling real continuous systems generally requires computer-based control, so sampling of the continuous system is necessary. Without considering the sample time used, problems arise in attempting to use a policy derived in simulation on an actual hardware experiment. For this scenario, if a model of the dynamics exists it is trivial to determine the best sample time by classical methods. However, here we consider the case where a model is not available and RL is being utilized for its model-free approach. This requires some way to determine the optimal sample time without the use of a model.

One more area of investigation that is needed for the learning of multi-agent system control policies is to learn some approximation of the dynamics. The learning process does not explicitly account for agent time dynamics, so that information is essentially abstracted out of learning. Since the main benefit of RL approaches like Q-learning is to learn a control policy without the need of a model, the benefits of determining a model have been overlooked. It can be very useful for some knowledge of the dynamics to inform the decisions made by the agents, and this is especially true in heterogeneous multi-agent systems. A full model may not be necessary, but some approximation of the individual agent dynamics can be very beneficial for determining global behavior rules.

In this paper, RL-based control approaches are extended to systems with unknown time dynamics. The scope of this paper is limited to the cases of simulated examples where the simulations have time dynamics but the RL agent learning to control the system has no access to that information. The systems considered are all sampled-data systems, and they will exhibit first- or second-order dynamics depending upon the algorithm being investigated. Three algorithms are introduced. One is capable of determining the optimal sample time for these sampled-data systems using RL-based algorithms while simultaneously learning the control policy, and it is named Sampled-Data Q-learning (SDQL). The second is an RL-based algorithm capable of learning some approximation of agent dynamics for a first-order system, and it is called First-Order Dynamics Learning (FODL). The final algorithm is similar to the FODL algorithm and is for second-order systems, called Second-Order Dynamics Learning (SODL). The end result of this research is the ability to learn an optimal sample time for agents, an approximation of agents' time dynamics, and individual agent control policies. This collective result allows for the control of heterogeneous multi-agent systems by means of hierarchical commands provided by a high-level agent with all of the knowledge learned by these algorithms.

This paper is organized as follows. In Section II, the basics of Reinforcement Learning are discussed, with an emphasis on Q-learning. Section III introduces the Sampled-Data Q-learning algorithm and includes simulation results to demonstrate it. The First- and Second-Order Dynamics Learning algorithms are introduced in Section IV with accompanying results, and conclusions and open challenges are discussed in Section V.

II. Q-learning

There are multiple classes of algorithms that fall within the definition of Reinforcement Learning. Currently, the RL algorithms that are most used in research are Temporal-Difference (TD) methods. TD methods are actually a conceptual combination from two other classes of algorithms known as Dynamic Programming (DP) and Monte Carlo.[6] Like Monte Carlo, TD methods use experience through interaction with the system to update the quality of the value function without the need of a model. Like DP, TD methods do not have to wait until the end to improve the value function, but rather update it along the way. This improves convergence time and also makes TD methods usable in online learning. In this section, the TD algorithm known as Q-learning will be discussed, along with the limitations that lead to the development of extended

Q-learning-based algorithms for the use of learning in dynamical sampled-data systems.

Of the various formulations of RL algorithms, Watkins' Q-learning has been the most accepted and utilized algorithm for its proven convergence to the optimal action-value function.[6] Q-learning is a TD method that learns the optimal action-value function in an off-policy manner.[7] This means that the policy used during a learning episode is not necessarily the same as the one that is updated at each timestep. A similar implementation that uses on-policy learning is known as Sarsa, and is often used in the same learning situations as Q-learning.[8,9] The Q-learning algorithm is based upon an action-value update rule that uses a greedy policy to determine a predicted value for the state-action pair at the next (future) timestep.[10,11] The actual action selection may not be done using a greedy policy, and in fact it is typically better for optimality to include some degree of exploration in the policy.[12] The rule used for updating the action-value function is as follows.

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)] \tag{1}$$

In Equation 1, Q is the action-value function, s is the state at the current timestep, a is the action selected for the current timestep using the agent's policy (e.g., ε-greedy), α is the step-size parameter, r is the reward received from the system, s' is the future state (due to taking action a), a' is the action that would be taken using a greedy policy when in state s', and γ is the weight for the future value. The selection of γ affects convergence time, and it is always within the range (0,1). The value of γ can either be kept constant throughout learning, or it can be chosen to vary episodically so that later learning episodes value the future prediction differently than early episodes. The value of α is always in the interval (0,1), and can either be held constant or varied by some user-defined function. In some cases, α is designed to decrease within an episode according to how often the agent revisits the same state, essentially punishing the agent for repeating itself unnecessarily.

The update rule of Equation 1 is the backbone of the famous Watkins' Q-learning algorithm. It is an off-policy TD algorithm with a user-determined policy for selecting actions at each timestep, but uses a fully-greedy policy with the action-value function when updating the action-value function. Rather than utilizing past information to perform this update, this algorithm uses the predicted future state-action pair chosen greedily. The Watkins' Q-learning algorithm is displayed in Algorithm 1.

Algorithm 1 Q-learning[6]

- Initialize $Q(s,a)$ arbitrarily
- Repeat for each episode:
  - Initialize s
  - Repeat for each timestep:
    - Choose a from s using policy derived from $Q(s,a)$ (e.g., ε-Greedy)
    - Take action a, observe r, s'
    - $Q(s,a) \leftarrow Q(s,a) + \alpha\ [\ r + \gamma\ \max_{a'}\ Q(s',a')$ - $Q(s,a)\ ]$
    - $s \leftarrow s'$
  - Until s is terminal

The policy for action selection has a drastic effect on the convergence of the Q-learning algorithm. Some balance of exploration and exploitation is needed to properly learn the full state-space and guarantee convergence to the optimal policy. An ε-greedy policy uses a user-defined probability, ε, that determines whether to choose actions randomly or according to the action-value function each time a new action must be taken. This speeds up the convergence time by reinforcing paths that have already been designated either good or bad while still allowing for new paths to be explored for optimality.[13] In the early episodes the agent is required to explore due to lack of knowledge, regardless of the value assigned to ε. This ε-greedy policy is used as the action selection method for Q-learning in this paper.

III. Sampled-Data Q-learning

When Algorithm 1 is used in dynamical systems, the problem of handling time-dependencies arises. Because it is much faster (and safer) to perform the learning in a computer simulation rather than online with a hardware system, researchers often turn to using dynamics models. The issue of ensuring an accurate model is obvious, but it is often overlooked that using the final control policy on a hardware model results in a sampled-data system. The control policy is handled using a discrete computer, but the policy is learned assuming continuous dynamics. In some cases, the user realizes this and assumes a sampled-data system in the simulated learning, but the Q-learning algorithm will converge to a policy that assumes the same sample time will always be used and that the chosen sample time is best. Here, an attempt to overcome this issue is addressed by wrapping the sample time into the learning process.

A. Sampled-Data Q-learning Algorithm

Incorporating the sample time, denoted T, into the learning process requires determining the value of individual sample times without adversely affecting the stability of the system during an episode. It would therefore be wise to not allow a sample time to change during a single episode. However, to determine optimal action-value functions for a range of sample times requires incorporating it into the state-space. It is therefore necessary to append the state-space with T while not allowing it to be affected by the action-space.

This gives rise to the question of how a particular sample time is to be selected when it cannot be affected by the action-space. It is necessary that a value be associated with each possible selection of T, but T must be held constant throughout an episode. It is therefore proposed that a state-value function for T be determined using Monte Carlo-based learning.[6] The value function can be updated according to an every-visit Monte Carlo method, shown in Equation 2.

$$V_T(T) \leftarrow V_T(T) + \alpha(R - V_T(T)) \tag{2}$$

This update rule will be the basis for Sampled-Data Q-learning. At the beginning of each episode, the sample time value-function, V_T, can be used to select T for the episode according to a user-defined policy. The sample time is appended to the system state vector, s, so that the normal Q-learning update rule will determine separate control policies according to different values of T. When the reward for a given timestep is determined, it is used to update both Q, and the total rewards for the episode are updated. At the end of the episode, V_T is updated by using the average return, R. This causes V_T to be updated such that episodes which experience more positive rewards than negative rewards result in the value associated with that particular T increasing. Likewise, when an episode experiences more negative rewards than positive, the total value for T after the episode ends will have decreased. Using this new sampled-data value function and update rule, the Sampled-Data Q-learning algorithm becomes as shown in Algorithm 2.

B. SDQL Results

For the system described above, a single agent was simulated as a robot that translates forward with a constant speed and rotates the heading angle ψ to change direction. The rotational dynamics are described as a first-order differential equation with time constant τ. The environment the robot is allowed to traverse is a 20m by 20m square with the origin at the center. The governing equations of motion are shown in Equations 3-5, where the subscript c denotes the commanded value.

$$\dot{x} = V\cos\psi \tag{3}$$

$$\dot{y} = V\sin\psi \tag{4}$$

$$\dot{\psi} = (\psi_c - \psi)\tau^{-1} \tag{5}$$

At each timestep, the learner evaluates the current state and chooses the appropriate action based on an ε-greedy policy in a Q-learning scheme. The possible actions are rotate clockwise ($-\Delta\psi$), no rotation ($\Delta\psi = 0$), and rotate counterclockwise($+\Delta\psi$). At each T, the robot is required to take a new action, which corresponds to a new commanded next state. The commanded next action occurs in intervals of 45 degrees. The action-space is shown in Equation 6.

Algorithm 2 Sampled-Data Q-learning (SDQL)

- Initialize $Q(\tilde{s},a)$ arbitrarily
- Initialize $V_T(T)$ arbitrarily
- Repeat for each episode:
  - Choose T using policy derived from $V_T(T)$ (e.g., ε-Greedy)
  - Initialize $R = 0$
  - Initialize s, initialize $\tilde{s}$ by appending T to s
  - Repeat for each sample timestep, T:
    - Choose a from $\tilde{s}$ using policy derived from $Q(\tilde{s},a)$ (e.g., ε-Greedy)
    - Take action a, observe r, $\tilde{s}'$
    - $Q(\tilde{s},a) \leftarrow Q(\tilde{s},a) + \alpha\ [\ r + \gamma\ \max_{a'}\ Q(\tilde{s}',a')\ -\ Q(\tilde{s},a)\ ]$
    - $\tilde{s} \leftarrow \tilde{s}'$
  - Until $\tilde{s}$ is terminal
  - $R = \text{average}(r)$
  - $V_T(T) \leftarrow V_T(T) + \alpha\ [\ R\ -\ V_T(T)\ ]$

$$a \in A = [\ -45^\circ \quad 0^\circ \quad +45^\circ\ ] \tag{6}$$

After simulating this problem using the SDQL algorithm, the result is a determined best sample time and a control policy based on that sample time. The values for the individual sample times after 10,000 learning episodes are shown in Table 1.

Table 1. SDQL Robot Result

$T(sec)$	V_T
0.01	29.1
0.02	29.9
0.03	2.4
0.04	42.9
0.05	51.4
0.06	73.4
0.07	2271.8
0.08	14.7
0.09	29.2
0.10	120.8

The control policy has the ability to control the robot to move from randomly initialized points to the goal within a tolerance of $\pm 1m$. The sample time with the maximum value determined by V_T of $T = 0.07$ sec is in this simulation of the resulting Q function. Figures 1-6 demonstrate the ability to control the robot to the goal using this learned function for the determined sample time.

Figures 1-3 show that the robot is able to guide itself to the goal from an initial point in quadrant 1. The state time history shown in Figure 2 shows that the robot is able to guide itself there in under 8 seconds of simulated real-time. The commanded heading angle changes at each $T = 0.07$ sec are shown in Figure 3.

For further demonstration of the ability to control the robot, an initial condition beginning in quadrant 3 was tested. These results are shown in Figures 4-6. Figure 5 shows that the robot is able to reach the goal in this case in approximately 15 seconds, and the commanded heading angle history is shown in Figure 6.

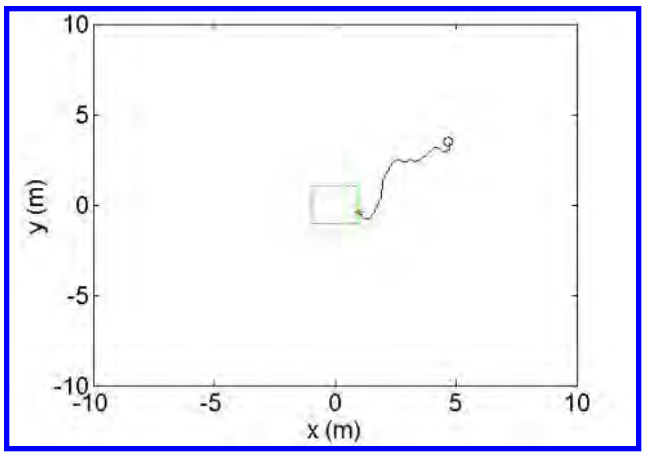


Figure 1. Simulation of Robot - Q1 Initial Condition

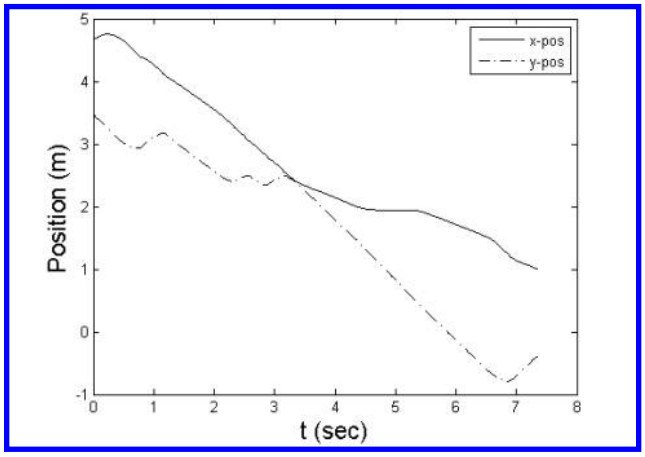


Figure 2. State History of Robot - Q1 Initial Condition

IV. System Dynamics Approximation

When using Q-learning to determine a control policy for dynamical systems, the benefit of not needing to have a model of the dynamics can lead to neglecting the dynamics completely. Learning a control policy for these systems is necessary, but often it is desired to also determine some approximation of the dynamics. This is especially important when dealing with the control of heterogeneous multi-agent systems since coordinating the agents requires knowing how the individual agents respond differently in time to similar action inputs.

A. First-Order Dynamics Learning

The simplest agent dynamics to represent are first-order dynamics. In a stable first-order system, the dynamics are described by the differential equation shown in Equation 7. Given the command value s_c, the time constant τ is needed to fully describe this differential equation.

$$\tau\dot{s} + s = s_c \tag{7}$$

The solution to this ODE is shown in Equation 8. This solution can be used to determine the next state, but τ is required to do so. This makes learning an approximate time constant all that is needed to determine the time behavior of the agent. Given the current state, s, the predicted next state, s^*, can be approximated using the current guess of the time constant, τ, and the sample time period, T, according to Equation 8.

$$s^* = se^{-T/\tau} + (1 - e^{-T/\tau})s_c \tag{8}$$

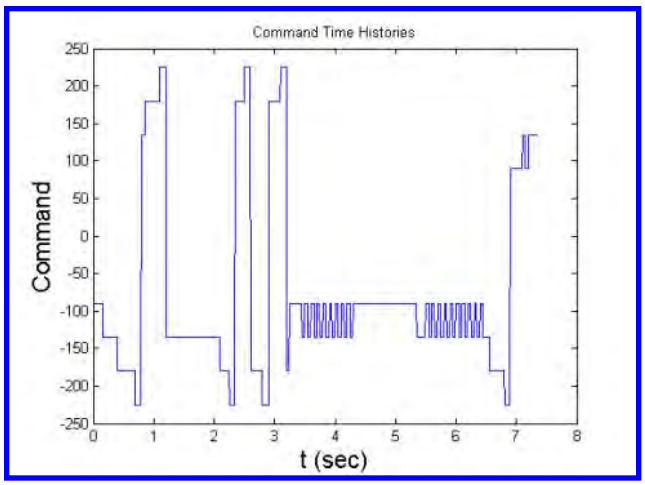


Figure 3. Command History of Robot - Q1 Initial Condition

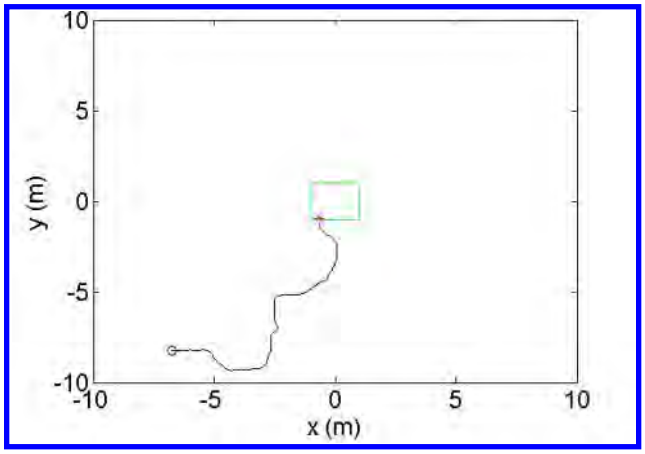


Figure 4. Simulation of Robot - Q3 Initial Condition

Equation 8 can be used to approximate a state transition for a single 1-dimensional state. Learning to approximate the time constant requires determining a reward that is a function of the current estimate of the sample time. By estimating the next state using Equation 8 and the current estimate of the time constant, the reward can be shaped by the error between the measured true next state and the estimated next state. The reward function used is shown in Equation 9.

$$r = ||s^* - s'||_2 \tag{9}$$

If more than one state dimension is to be approximated, multiple time constants would be needed. For instance, if this method were applied to a robot traversing a 2-D space with first-order dynamics in forward translation and rotation, one might want to know the state transition dynamics of both $\tau_{forward}$ and τ_{rotate} independently as they would most likely have different dynamics. Time constants for each state-action pair would be determined based on whether the robot were moving forward or rotating. To learn a time constant, a Monte Carlo learning formulation similar to the SDQL algorithm can be used. Algorithm 3 can shows how this can be accomplished.[14]

Algorithm 3 was used for the same example shown in Section III.B. After the 10,000 learning episodes completed, the FODL algorithm was able to successfully converge to the proper value of the time constant of $\tau = 0.2$ sec. Table 2 shows the V_1 values associated with each time constant, τ.

Over the course of the 10,000 learning episodes, the value associated with τ evolved as shown in Figure 7. As can be seen, the value associated with $\tau = 0.2$ became the maximum value early in the learning process. As learning episodes continue, the value function reinforces this as the best estimate of the time constant.

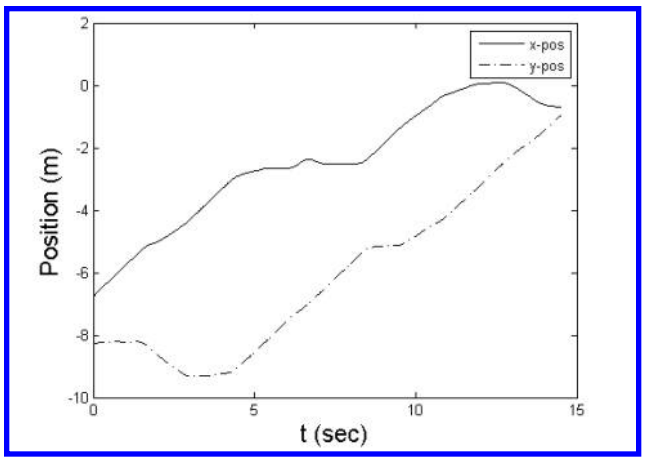


Figure 5. State History of Robot - Q3 Initial Condition

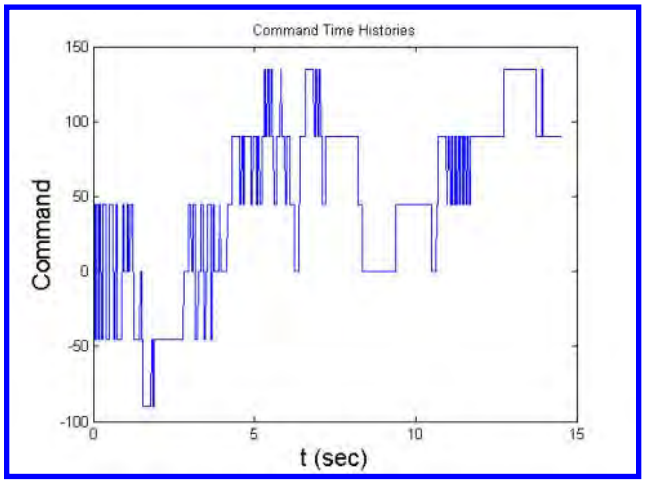


Figure 6. Command History of Robot - Q3 Initial Condition

B. Second-Order Dynamics Learning

In general, real world systems are more prone to exhibit second-order behavior rather than first-order. Second-order systems are the simplest systems that exhibit overshoot and oscillations, so higher-order systems can be approximated using second-order models.[15] It is therefore necessary to learn approximate models of second-order systems to approximate any systems that are higher than first-order. The second-order dynamics can be described by Equation 10.

$$\frac{d^2s}{dt^2} + 2\zeta\omega_n\frac{ds}{dt} + \omega_n^2 s = \omega_n^2 s_c \tag{10}$$

To approximate a second-order system requires determining 2 parameters: natural frequency and damping ratio. The learning framework used to determine these parameters is the same as for the first-order case, but the value function and the state approximation equations are different. The value function, here called V_2, is a function of the frequency and damping ratio. The state prediction equation is the solution to Equation 10 after a time period of T. This solution is dependent on the current approximation of damping ratio. For the case of $\zeta = 0$, the solution is as shown in Equation 11.

$$s^* = s_c + \frac{\dot{s}}{\omega_n}\sin(\omega_n T) + (s - s_c)\cos(\omega_n T) \tag{11}$$

For the case of $0 < \zeta < 1$, the solution is as shown in Equation 12.

Algorithm 3 First-Order Dynamics Learning (FODL)

- Determine T and $Q(s,a)$ for system (e.g., Sampled-Data Q-learning)
- Initialize $V_1(s,a,\tau)$ arbitrarily
- Repeat for each episode:
  - Initialize s, append T to s
  - Repeat for each timestep:
    - Choose a from s using greedy policy derived with $Q(s,a)$
    - Choose τ using policy derived from $V_1(s,a,\tau)$ (e.g., ε-Greedy)
    - Predict next state, s^*, with s,a, and τ using first-order approximations
    - Take action a, observe actual next state, s'
    - Observe r shaped from observed s' and predicted s^*
    - $V_1(s,a,\tau) \leftarrow V_1(s,a,\tau) + \alpha\ [\ r$ - $V_1(s,a,\tau)\ ]$
    - $s \leftarrow s'$
  - Until s is terminal

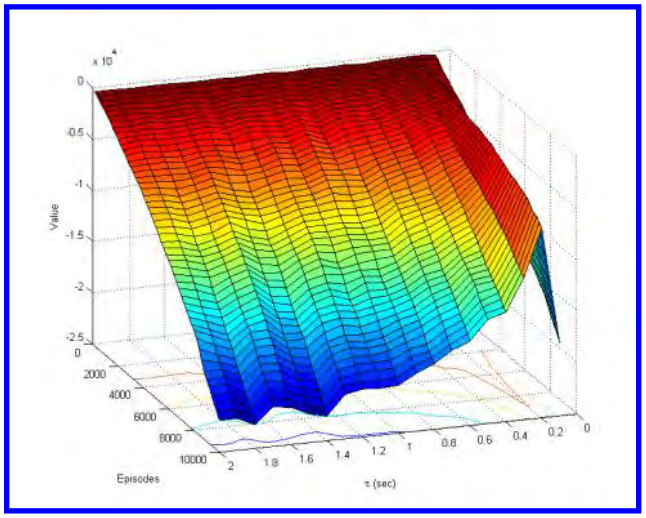


Figure 7. Time Constant Value History

$$s^* = s_c + \frac{\zeta}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n T}\left(s - s_c + \frac{\dot{s}}{\zeta\omega_n}\right)\sin(\omega_n T\sqrt{1-\zeta^2}) + e^{-\zeta\omega_n T}(s - s_c)\cos(\omega_n T\sqrt{1-\zeta^2}) \tag{12}$$

For the case of $\zeta = 1$, the solution is as shown in Equation 13.

$$s^* = s_c + (s - s_c)e^{-\omega_n T} + (\dot{s} + \omega_n s - \omega_n s_c)Te^{-\omega_n T} \tag{13}$$

And for the case of $\zeta > 1$, the solution is as shown in Equation 14.

Table 2. FODL Robot Value Function

$\tau(sec)$	V_1
0.1	-1.79×10^4
0.2	-0.65×10^4
0.3	-1.12×10^4
0.4	-1.44×10^4
0.5	-1.49×10^4
0.6	-1.68×10^4
0.7	-1.75×10^4
0.8	-1.86×10^4
0.9	-1.94×10^4
1.0	-2.03×10^4
1.1	-2.05×10^4
1.2	-2.03×10^4
1.3	-2.07×10^4
1.4	-2.26×10^4
1.5	-2.22×10^4
1.6	-2.14×10^4
1.7	-2.11×10^4
1.8	-2.27×10^4
1.9	-2.19×10^4
2.0	-2.19×10^4

$$s^* = s_c + \left(\frac{\dot{s} + (2\zeta\omega_n - z_1)s - s_c z_2}{z_2 - z_1} \right) e^{-z_1 T} - \left(\frac{\dot{s} + (2\zeta\omega_n + z_2)s - s_c z_1}{z_2 - z_1} \right) e^{-z_2 T} \tag{14}$$

where

$$z_1 = \omega_n(\zeta - \sqrt{\zeta^2 - 1})$$
$$z_2 = \omega_n(\zeta + \sqrt{\zeta^2 - 1})$$

To learn the second-order parameters, the FODL algorithm can be adjusted for learning the natural frequency and damping ratio rather than the time constant. This alternate version of dynamics learning, called Second-Order Dynamics Learning, is shown in Algorithm 4.

To demonstrate the SODL algorithm, the robot example from before was altered to have second-order dynamics in the heading angle equation of motion. For this example, the natural frequency of the robot heading angle was set to $\omega_n = 6$ rad/sec and the damping ratio is $\zeta = 0.8$. After 10,000 episodes of learning using Algorithm 4, the value of V_2 determined a maximum value associated with the correct frequency and damping ratio. Table 3 shows the final values, and they are plotted in Figure 8.

The way that the V_2 function is formulated implies that the value determined is for the *combination* of the variables ω and ζ rather than designating values for each separately. This is done because it is both of these variables together that determines the behavior of a system, and not either one separately. However, if one were to want to see how they are valued individually then the average values can be determined. If the value function entries for each instance of a particular ω are averaged together, an estimate of the value for that frequency is determined. Likewise, the same can be done for the damping ratio. Figures 9-10 show how the average values for the individual parameters evolve over time.

Algorithm 4 Second-Order Dynamics Learning (SODL)

- Determine T and $Q(s,a)$ for system (e.g., Sampled-Data Q-learning)
- Initialize $V_2(s,a,\omega_n,\zeta)$ arbitrarily
- Repeat for each episode:
  - Initialize s, append T to s
  - Repeat for each timestep:
    - Choose a from s using greedy policy derived with $Q(s,a)$
    - Choose ω_n and ζ using policy derived from $V_2(s,a,\omega_n,\zeta)$ (e.g., ε-Greedy)
    - Predict next state s^* with s, a, ω_n, and ζ using ζ-dependent solution
    - Take action a, observe actual next state s'
    - Observe r shaped from observed s' and predicted s^*
    - $V_2(s,a,\omega_n,\zeta) \leftarrow V_2(s,a,\omega_n,\zeta) + \alpha\ [\ r$ - $V_2(s,a,\omega_n,\zeta)\ ]$
    - $s \leftarrow s'$
  - Until s is terminal

Table 3. V_2 after 10,000 Episodes

	$\omega = 2$	$\omega = 4$	$\omega = 6$	$\omega = 8$	$\omega = 10$
$\zeta = 0.4$	−3220	−3107	−2412	−2671	−5130
$\zeta = 0.8$	−3450	−2610	−1466	−2606	−4110
$\zeta = 1.2$	−3283	−2883	−2393	−2323	−4092
$\zeta = 1.6$	−3008	−2855	−3180	−2940	−3155
$\zeta = 2.0$	−3240	−3602	−2795	−2576	−3484

V. Conclusions and Open Challenges

The work presented in this paper has shown that Reinforcement Learning-based techniques can be adapted to not only learn control policies for agents, but also learn an approximation of the agents' dynamics. Several conclusions can be drawn from these results. The Sampled-Data Q-learning algorithm is capable of determining longer sample times that still allow for successful control of the agent. The First-Order Dynamics Learning algorithm is capable of determining the time constants that best model the dynamics of the states for an individual agent with first-order dynamics. The Second-Order Dynamics Learning algorithm is capable of determining the best combination of natural frequency and damping ratio to model the second-order dynamics of the states for an agent.

There are a number of challenges for future research efforts. First, after learning the dynamics of agents it can be shown that in a hierarchical multiagent system the supervisory agents can use the dynamics information to determine commands to lower-level agents. Adaptation of these algorithms to stochastic systems is also possible. Another open problem is determining a means to either demonstrate the Second-Order Dynamics Learning algorithm's capability to approximate higher-order systems, or investigate alternate algorithms for approximating higher-order dynamics.

Acknowledgment

This work was sponsored (in part) by the Air Force Office of Scientific Research, USAF, under grant/contract number FA9550-08-1-0038. The technical monitor is Dr. Fariba Fahroo. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government.

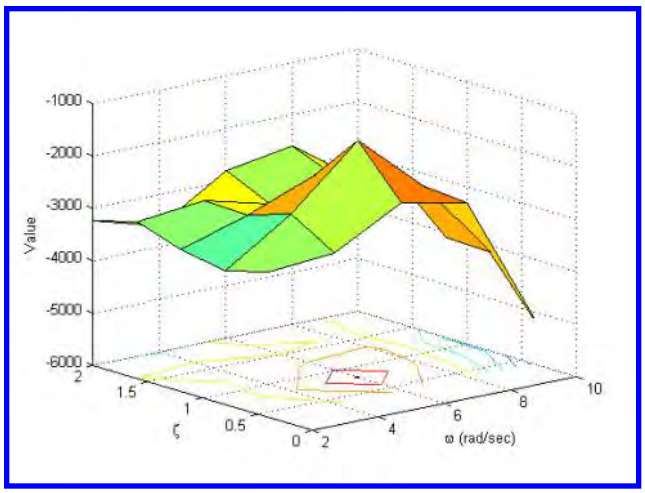


Figure 8. V_2 after 10,000 Episodes

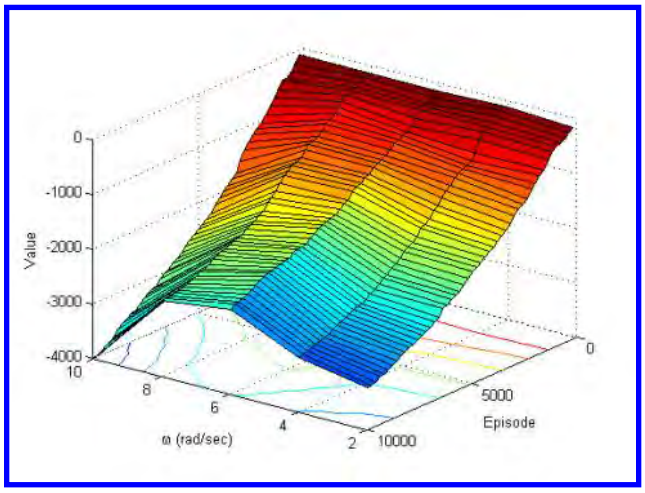


Figure 9. Natural Frequency Value History

References

[1]C. Claus and C. Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of National Conference on Artificial Intelligence*, AAAI-98, pages 746–752, 1998.

[2]J. Hu and M.P. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *Proceedings of the 15th International Conference on Machine Learning*, pages 242–250, Madison, WI, 1998.

[3]M. Lauer and M. Riedmiller. An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 535–542, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[4]S. Kapetanakis and D. Kudenko. Reinforcement learning of coordination in cooperative multi-agent systems. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence*, AAAI-02, pages 326–331, 2002.

[5]G. Tesauro. Extending q-learning to general adaptive multiagent systems. In *Advances in Neural Information Processing Systems*, volume 16, Vancouver and Whistler, Canada, 8-13 December 2003.

[6]R. Sutton and A. Barto. *Reinforcement Learning: An Introductions*. The MIT Press, Cambridge, MA, 1998.

[7]C. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992.

[8]K. Kirkpatrick and J. Valasek. Reinforcement learning for characterizing hysteresis behavior of shape memory alloys. *Journal of Aerospace Computing, Information, and Communication*, 6(3):227–238, March 2009.

[9]K. Kirkpatrick and J. Valasek. Active length control of shape memory alloy wires using reinforcement learning. *Journal of Intelligent Material Systems and Structures*, 22(14):1595–1604, September 2011.

[10]J. Valasek, M. Tandale, and J. Rong. A reinforcement learning - adaptive control architecture for morphing. *Journal of Aerospace Computing, Information, and Communication*, 2(5):174–195, April 2005.

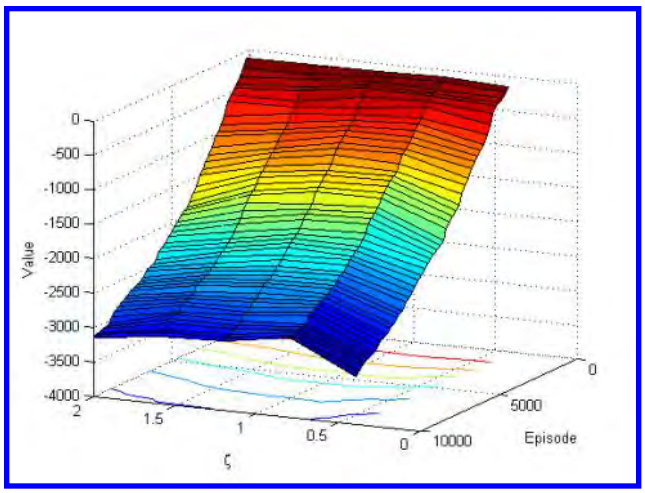


Figure 10. Damping Ratio Value History

[11]J. Valasek, J. Doebbler, M. D. Tandale, and A. J. Meade. Improved adaptive-reinforcement learning control for morphing unmanned air vehicles. *IEEE Transactions on Systems, Man, and Cybernetics: Part B*, 38(4):1014–1020, August 2008.

[12]A. Lampton. *Discretization and Approximation Methods for Reinforcement Learning of Highly Reconfigurable Systems*. PhD thesis, Texas A&M University, October 2009.

[13]S. Whiteson, M. E. Taylor, and P. Stone. Empirical studies in action selection with reinforcement learning. *Adaptive Behavior*, 15:33–50, 2007.

[14]K. Kirkpatrick and J. Valasek. Reinforcement learning control with time dependent agent dynamics. In F. L. Lewis and D. Liu, editors, *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*. Wiley.

[15]Gene F. Franklin, Michael L. Workman, and Dave Powell. *Digital Control of Dynamic Systems*. 1997.

2013 American Control Conference (ACC)
Washington, DC, USA, June 17-19, 2013

A Constructive Stabilization Approach for Open-Loop Unstable Non-Affine Systems*

Anshu Narang-Siddarth[1] and John Valasek[2]

***Abstract*— This paper focuses on the stabilization of non-affine-in-control systems that are open-loop unstable. The main result of the paper is a general method for constructing feedback stabilization of all non-affine systems. The synthesis procedure is based on concepts of feedback passivation, and is extended for non-affine systems by deriving sufficient conditions for passivity. The developments and essential ideas of the proposed technique are validated via simulation.**

I. INTRODUCTION

Consider the core problem of developing stabilizing controllers for *non-affine* systems of the form

$$\underline{\Sigma} : \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state, $\mathbf{u} \in \mathbb{R}^m$ is the control input, and $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a sufficiently smooth vector field. Assume that

(A1) The unforced dynamics of $\underline{\Sigma}$ in (1), namely $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{0}) \triangleq \mathbf{f}_0(\mathbf{x})$ is unstable.

In general the function $\mathbf{f}(\mathbf{x}, \mathbf{u})$ is not monotonic in the control and $\frac{\partial \mathbf{f}}{\partial \mathbf{u}}$ may be singular at the origin. Thus $\underline{\Sigma}$ cannot be stabilized using either dynamic-inversion [1] or the modeling error compensation technique [2]. Furthermore, non-affine systems of the form given in (1) cannot be stabilized using a fixed-gain static compensator. To see this behaviour consider the system

$$\dot{x} = x - 2u^3. \tag{2}$$

It is open-loop unstable and satisfies Assumption (A1). Suppose the control takes the form $u = K(t)x$ in (2). Then the resulting closed-loop dynamics become $\dot{x} = x - 2K^3x^3$ that has the following equilibrium solutions:

$$x_* = \begin{cases} 0 & \text{for all } K \\ \pm\frac{1}{\sqrt{2K^3}} & \text{for } K > 0. \end{cases} \tag{3}$$

The bifurcation map (Fig. 1) illustrates that the origin remains unstable except for $K = \infty$. Furthermore, the system has three equilibrium solutions for positive values of the feedback gain that converge to the origin as $K \to \infty$. This behaviour indicates that only an infinite control effort can stabilize the origin.

*This work was supported in part by the U.S. Air Force Office of Scientific Research under contract FA9550-08-1-0038 with technical monitor Dr. Fariba Fahroo.

[1]A. Siddarth is a Post-Doctoral Research Associate in the Vehicle Systems & Control Laboratory, Aerospace Engineering Department, Texas A&M University, College Station, TX 77843-3141, USA anshun1@tamu.edu

[2]J. Valasek is Professor and Director, Vehicle Systems & Control Laboratory, Aerospace Engineering Department, Texas A&M University, College Station, TX 77843-3141, USA valasek@tamu.edu

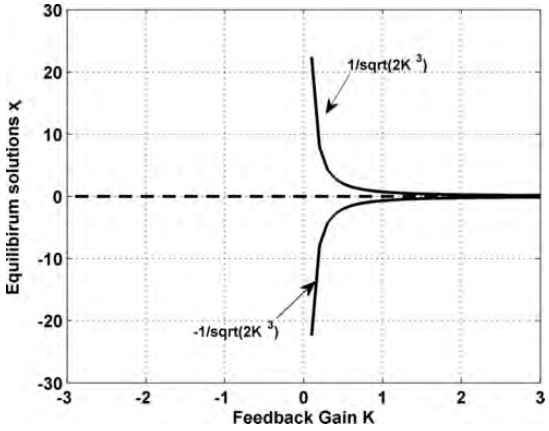


Fig. 1. Stable (solid lines) and unstable (broken line) equilibrium solutions of (2) with $u = Kx$

An alternative solution to regulate (2) is to switch feedback gains in accordance with the current state. As an example, suppose the feedback gain for (2) was initialized to $K(t = 0) = 1.5$. Then from (3) the state would stabilize to either $x_{steady} = -0.384$ or $x_{steady} = 0.384$ depending on its initial condition. Next, if the feedback gain is switched to $K(t > t_{steady}) = 2$ the state would stabilize at $x_{steady} = \pm 0.25$. Hence, increasing the gain successively the origin can be stabilized through a finite control input. The fundamental problem with this switching scheme is that analytic determination of the switching curves for general systems of the form (1) requires substantial system knowledge and offline processing. Additionally, the switching conditions and the number of control switches depend on the initial condition and the control form, leading to a system specific design [3], [4].

In this paper the construction of an analytic state-feedback control law is pursued. The major contribution of the paper is a theoretical result which shows that under mild conditions a control law of the form $\mathbf{u}(\mathbf{x}) = \boldsymbol{\alpha}(\mathbf{x}) + \boldsymbol{\nu}(\mathbf{x})$ globally stabilizes a large class of non-affine systems. The function $\boldsymbol{\alpha}(\mathbf{x})$ converts an open-loop unstable system into a stable system in the Lyapunov sense, and $\boldsymbol{\nu}(\mathbf{x})$ is constructed to bring about the necessary energy dissipation for globally stabilizing the origin. The design procedure presented here is based on the ideas of *feedback passivation* introduced in [5] for control-affine systems. The general concept is to use state-feedback to render the system passive and then employ well-established results for stabilizing passive systems. Toward this end, the fundamental question to be

answered is *when is a general nonlinear system passive?* The well-known Kalman-Yacubovitch-Popov lemma [6] and its nonlinear counterparts derived by Hill and Moylan [7] answer this question for linear and affine-in-control systems respectively. Sufficient conditions for passivity of non-affine systems and their relationship with the existing necessary conditions [8] are derived for the first time in this paper, in Section II. The main result for stabilization of general multiple-input systems posed as sufficiency conditions is derived in Section III. The theoretical results are verified with simulation examples in Section IV. Closing remarks are discussed in Section V.

II. Passive Systems

In this section the sufficiency conditions under which a nonlinear system can be considered passive are derived. Consider

$$\Sigma_1 : \begin{array}{l} \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \\ \mathbf{y} = \mathbf{h}(\mathbf{x}, \mathbf{u}) \end{array} \tag{4}$$

with state-space $X = \mathbb{R}^n$, set of input values $U = \mathbb{R}^m$, and set of output values $Y = \mathbb{R}^m$. The set $\mathcal{U}$ of admissible inputs consists of all U-valued piece-wise continuous functions defined on $\mathbb{R}$. The functions $\mathbf{f}(.)$ and $\mathbf{h}(.)$ are continuously differentiable maps defined on the open subset $O \subset \mathbb{R}^n$. It is assumed that these vector fields are smooth mappings, with at least one equilibrium. Without loss of generality the origin is chosen as the equilibrium of Σ_1, that is, $\mathbf{f}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$ and $\mathbf{h}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$. In order to derive conditions for Σ_1 to be passive two important definitions are reviewed and presented below.

Definition II.1. *The system Σ_1 is said to be passive if there exists a positive semi-definite storage function $V(\mathbf{x})$ that satisfies $V(\mathbf{0}) = 0$ and for any $\mathbf{u} \in \mathcal{U}$ and initial condition $\mathbf{x}_0 \in X$*

$$V(\mathbf{x}) - V(\mathbf{x}_0) \leq \int_0^t \mathbf{y}^T(s)\mathbf{u}(s)ds. \tag{5}$$

If the storage function is C^r times continuously differentiable with $r \geq 1$ then (5) is equivalent to

$$\dot{V} \leq \mathbf{y}^T\mathbf{u}. \tag{6}$$

Definition II.1 is the mathematical analog of stating that the amount of energy stored in a passive system is less than or equal to the energy being input. For convenience define the vector fields

$$\mathbf{f}_0(\mathbf{x}) \triangleq \mathbf{f}(\mathbf{x}, \mathbf{0}) \in \mathbb{R}^n \tag{7a}$$

$$\mathbf{h}_0(\mathbf{x}) \triangleq \mathbf{h}(\mathbf{x}, \mathbf{0}) \in \mathbb{R}^m \tag{7b}$$

where $\mathbf{f}_0(\mathbf{x})$ represents the open-loop dynamics of Σ_1 while $\mathbf{h}_0(\mathbf{x})$ is the output of Σ_1 at zero-input. Using (7) and the fact that the vector fields in Σ_1 are smooth, (4) is equivalently represented as

$$\dot{\mathbf{x}} = \mathbf{f}_0(\mathbf{x}) + \mathbf{g}(\mathbf{x}, \mathbf{u})\mathbf{u} \tag{8a}$$

$$\mathbf{h}(\mathbf{x}, \mathbf{u}) = \mathbf{h}_0(\mathbf{x}) + \mathbf{j}(\mathbf{x}, \mathbf{u})\mathbf{u} \tag{8b}$$

where the following identities have been used:

$$\mathbf{f}(\mathbf{x}, \mathbf{u}) - \mathbf{f}_0(\mathbf{x}) = \left(\int_0^1 \frac{\partial \mathbf{f}(\mathbf{x}, \gamma)}{\partial \boldsymbol{\gamma}}\Big|_{\boldsymbol{\gamma} = \theta \mathbf{u}} d\theta \right) \mathbf{u}(\mathbf{x}) \triangleq \mathbf{g}(\mathbf{x}, \mathbf{u})\mathbf{u} \tag{9}$$

$$\mathbf{h}(\mathbf{x}, \mathbf{u}) - \mathbf{h}_0(\mathbf{x}) = \left(\int_0^1 \frac{\partial \mathbf{h}(\mathbf{x}, \gamma)}{\partial \boldsymbol{\gamma}}\Big|_{\boldsymbol{\gamma} = \theta \mathbf{u}} d\theta \right) \mathbf{u}(\mathbf{x}) \triangleq \mathbf{j}(\mathbf{x}, \mathbf{u})\mathbf{u}. \tag{10}$$

The vector fields $\mathbf{g}(\mathbf{x}, \mathbf{u})$ and $\mathbf{j}(\mathbf{x}, \mathbf{u})$ defined above capture the effect of the control input on the motion of the dynamical system states and the output. Notice that for control-affine systems these vector fields will be independent of the control input vector. Using smoothness of the vector $\mathbf{g}(\mathbf{x}, \mathbf{u})$, (8a) can be further decomposed as

$$\dot{\mathbf{x}} = \mathbf{f}_0(\mathbf{x}) + \mathbf{g}_0(\mathbf{x})\mathbf{u} + \sum_{i=1}^{m} u_i \left[\mathbf{R}_i(\mathbf{x}, \mathbf{u})\mathbf{u} \right] \tag{11}$$

with $\mathbf{R}_i(\mathbf{x}, \mathbf{u}) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n \times m}$, being a smooth map for $1 \leq i \leq m$ and

$$\mathbf{g}_i^0(\mathbf{x}) = \mathbf{g}_i(\mathbf{x}, \mathbf{0}) = \frac{\partial \mathbf{f}}{\partial u_i}(\mathbf{x}, \mathbf{0}) \in \mathbb{R}^n; \quad 1 \leq i \leq m \tag{12a}$$

$$\mathbf{g}_0(\mathbf{x}) = \frac{\partial \mathbf{f}}{\partial \mathbf{u}}(\mathbf{x}, \mathbf{0}) = \left[\mathbf{g}_1^0(\mathbf{x}), \ldots, \mathbf{g}_m^0(\mathbf{x}) \right] \in \mathbb{R}^{n \times m} \tag{12b}$$

The vector field $\mathbf{g}_i^0(\mathbf{x})$ defines the influence of the input u_i on the system about the origin and is collected for all inputs under the vector $\mathbf{g}_0(\mathbf{x})$.

The next definition gives the necessary conditions for an input/output nonlinear system Σ_1 to be passive. In the following and the rest of the paper, the expression

$$\mathcal{L}_{\mathbf{f}_0} V = \left\langle \frac{\partial V}{\partial \mathbf{x}}, \mathbf{f}_0(\mathbf{x}) \right\rangle \tag{13}$$

represents the Lie derivative of the $C^r (r \geq 1)$ functional $V : \mathbb{R}^n \to \mathbb{R}$ along the vector field $\mathbf{f}_0(\mathbf{x})$. Additionally, the standard notation $ad_{\underline{f}_0}^k \underline{g}_i^0$ is used for Lie bracket.

Definition II.2. *[8]. Let $\Omega_1 \triangleq \{\mathbf{x} \in \mathbb{R}^n : \mathcal{L}_{\mathbf{f}_0} V(\mathbf{x}) = 0\}$. Necessary conditions for Σ_1 to be passive with a C^2 positive semi-definite storage function V are*

(i) $\mathcal{L}_{\mathbf{f}_0} V(\mathbf{x}) \leq 0$,
(ii) $\mathcal{L}_{\mathbf{g}_0} V(\mathbf{x}) = \mathbf{h}_0^T(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega_1$,
(iii) $\sum_{i=1}^{n} \frac{\partial^2 f_i}{\partial \mathbf{u}^2}(\mathbf{x}, \mathbf{0}).\frac{\partial V}{\partial x_i} \leq \mathbf{j}^T(\mathbf{x}, \mathbf{0}) + \mathbf{j}(\mathbf{x}, \mathbf{0}) \quad \forall \mathbf{x} \in \Omega_1$,

where $f_i(\mathbf{x}, \mathbf{u})$ is the ith component of the vector function $\mathbf{f}(\mathbf{x}, \mathbf{u})$.

For a positive-definite storage function property (i) is analogous to Lyapunov's condition $\dot{V} \leq 0$ for bounded stability. The other conditions in Definition II.2 follow directly from Definition II.1 by noticing that the difference $\frac{\partial V}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}, \mathbf{u}) - \mathbf{h}^T(\mathbf{x}, \mathbf{u})\mathbf{u}$ attains its maximum at $\mathbf{u} = \mathbf{0}$ on the set Ω_1.

The following theorem completes Definition II.2 by presenting the sufficiency conditions required for a system Σ_1 to be passive.

Theorem 1. *Let V be a C^1 positive semi-definite function. A system Σ_1 is passive if there exists some functions $q : \mathbb{R}^n \to$*

$\mathbb{R}^k$, $W : \mathbb{R}^n \to \mathbb{R}^{k\times m}$ *and* $H : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{k\times m}$, *for some integer* k *such that*

(i) $\mathcal{L}_{\mathbf{f}_0} V(\mathbf{x}) = -\frac{1}{2} q^T(\mathbf{x}) q(\mathbf{x})$,
(ii) $\mathcal{L}_{\mathbf{g}_0} V(\mathbf{x}) = \mathbf{h}_0^T(\mathbf{x}) - q^T(\mathbf{x}) W(\mathbf{x})$,
(iii) $\frac{1}{2}\left[W(\mathbf{x}) + H(\mathbf{x},\mathbf{u})\right]^T \left[W(\mathbf{x}) + H(\mathbf{x},\mathbf{u})\right]$
$= \frac{1}{2}\left[\mathbf{j}(\mathbf{x},\mathbf{u})^T + \mathbf{j}(\mathbf{x},\mathbf{u})\right] - \mathcal{L}_{\mathbf{R}(\mathbf{x},\mathbf{u})} V$,
(iv) $W^T(\mathbf{x}) H(\mathbf{x},\mathbf{u}) + H^T(\mathbf{x},\mathbf{u}) W(\mathbf{x})$ *is positive-definite.*

In the conditions above

$$\mathcal{L}_{\mathbf{R}(\mathbf{x},\mathbf{u})} V = \left[\mathcal{L}_{\mathbf{R}_1(\mathbf{x},\mathbf{u})} V, \cdots, \mathcal{L}_{\mathbf{R}_m(\mathbf{x},\mathbf{u})} V\right]^T \in \mathbb{R}^{m\times m}$$

Proof. The proof follows the developments given in [9]. Assuming functions $q(\mathbf{x})$, $W(\mathbf{x})$, and $H(\mathbf{x},\mathbf{u})$ exist, then along the solutions of Σ_1

$$\begin{aligned}\dot{V} \leq & \dot{V} + \frac{1}{2}\left[W(\mathbf{x})\mathbf{u} + q(\mathbf{x})\right]^T \left[W(\mathbf{x})\mathbf{u} + q(\mathbf{x})\right] \\ & + \frac{1}{2}\mathbf{u}^T\Big[W^T(\mathbf{x}) H(\mathbf{x},\mathbf{u}) + H^T(\mathbf{x},\mathbf{u}) W(\mathbf{x})\Big]\mathbf{u} \\ & + \frac{1}{2}\mathbf{u}^T H^T(\mathbf{x},\mathbf{u}) H(\mathbf{x},\mathbf{u})\mathbf{u} \end{aligned} \tag{14}$$

through property (iv) of Theorem 1. Rearrange (14) to get

$$\begin{aligned}\dot{V} \leq & \mathcal{L}_{\mathbf{f}_0} V + \mathcal{L}_{\mathbf{g}_0} V\mathbf{u} + \mathbf{u}^T \mathcal{L}_{\mathbf{R}(\mathbf{x},\mathbf{u})} V\mathbf{u} + \frac{1}{2} q^T(\mathbf{x}) q(\mathbf{x}) \\ & + \frac{1}{2}\left[q^T(\mathbf{x}) W(\mathbf{x})\mathbf{u} + \mathbf{u}^T W^T(\mathbf{x}) q(\mathbf{x})\right] \\ & + \frac{1}{2}\left[W(\mathbf{x}) + H(\mathbf{x},\mathbf{u})\right]^T \left[W(\mathbf{x}) + H(\mathbf{x},\mathbf{u})\right]. \end{aligned} \tag{15}$$

Using properties (i) through (iii) given in Theorem 1, (15) becomes

$$\begin{aligned}\dot{V} &\leq \mathbf{h}_0^T(\mathbf{x})\mathbf{u} + \frac{1}{2}\mathbf{u}^T\left[\mathbf{j}(\mathbf{x},\mathbf{u})^T + \mathbf{j}(\mathbf{x},\mathbf{u})\right]\mathbf{u} \\ &\leq \mathbf{y}^T\mathbf{u}. \end{aligned} \tag{16}$$

Thus, comparing (16) with (6) it is concluded that Σ_1 is passive and $V(\mathbf{x})$ is the storage function. This completes the proof. □

Remark 1. *Notice on the set* Ω_1 *defined in Definition II.2 that properties (i) through (iii) of Theorem 1 become exactly the necessary conditions for passivity. Thus, Theorem 1 plays the role of the generalized KYP lemma for non-affine systems on the set* Ω_1.

For an affine Σ_1, Theorem 1 becomes exactly the nonlinear version of the KYP lemma derived by Hill and Moylan [7]. This result is summarized in the following corollary.

Corollary 1. *Let* V *be a* C^1 *positive semi-definite function. A system*

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{f}_0(\mathbf{x}) + \mathbf{g}_0(\mathbf{x})\mathbf{u} \\ \mathbf{y} &= \mathbf{h}_0(\mathbf{x}) + \mathbf{j}(\mathbf{x})\mathbf{u}\end{aligned}$$

is passive if and only if

(i) $\mathcal{L}_{\mathbf{f}_0} V = -\frac{1}{2} q^T(\mathbf{x}) q(\mathbf{x})$,
(ii) $\mathcal{L}_{\mathbf{g}_0} V = \mathbf{h}_0^T(\mathbf{x}) - q^T(\mathbf{x}) W(\mathbf{x})$,
(iii) $\frac{1}{2} W^T(\mathbf{x}) W(\mathbf{x}) = \frac{1}{2}\left[j(\mathbf{x})^T + j(\mathbf{x})\right]$.

III. Control Synthesis for Stabilization

This section addresses control design for general non-affine systems. Suppose the control is decomposed as

$$\mathbf{u}(\mathbf{x}) = \boldsymbol{\alpha}(\mathbf{x}) + \boldsymbol{\nu}(\mathbf{x}) \tag{17}$$

and the first component $\boldsymbol{\alpha}(\mathbf{x})$ is used to ensure the non-affine system under consideration is passive through input $\boldsymbol{\nu}(\mathbf{x})$ for some dummy output $\mathbf{y}(\mathbf{x})$. Then asymptotic stabilization is guaranteed through pure output feedback under zero-state detectability conditions [6]. The construction and proof that the choice of control in (17) asymptotically stabilizes a non-affine system is the focus of this section.

A. Control Synthesis for Multi-Input Non-Affine Systems

This result is an algorithm for stabilizing non-affine systems of the form

$$\underline{\Sigma} : \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{u}); \quad \mathbf{x}(0) = \mathbf{x}_0 \tag{1}$$

with state-space $X = \mathbb{R}^n$ and set of input values $U = \mathbb{R}^m$. The set $\mathcal{U}$ of admissible inputs consists of all U-valued piecewise continuous functions defined on $\mathbb{R}$. The vector field $\mathbf{f}(.)$ is a continuously differentiable map defined on the open subset $O \subset \mathbb{R}^n$. Without loss of generality, the origin is chosen as the equilibrium of $\underline{\Sigma}$. The control algorithm is detailed in the following four steps.

Step 1: Define vector fields for the system under study:

$$\begin{aligned}\underline{f}_0(\mathbf{x}) &= \mathbf{f}(\mathbf{x}, \boldsymbol{\alpha}(\mathbf{x})) \in \mathbb{R}^n \\ \underline{g}(\mathbf{x}, \boldsymbol{\nu}(\mathbf{x})) &= \int_0^1 \frac{\partial \mathbf{f}\left(\mathbf{x}, \boldsymbol{\alpha}(\mathbf{x}) + \boldsymbol{\gamma}\right)}{\partial \boldsymbol{\gamma}}\bigg|_{\boldsymbol{\gamma}=\theta\nu} d\theta \quad \in \mathbb{R}^{n\times m} \\ \underline{g}_i^0 &= \underline{g}_i(\mathbf{x}, \mathbf{0}) \in \mathbb{R}^n.\end{aligned} \tag{18}$$

Under the influence of the control in (17) and the definitions above, $\underline{\Sigma}$ becomes

$$\dot{\mathbf{x}} = \underline{f}_0(\mathbf{x}) + \underline{g}(\mathbf{x}, \boldsymbol{\nu}(\mathbf{x}))\boldsymbol{\nu}(\mathbf{x}). \tag{19}$$

Notice that the vector field $\underline{f}_0(\mathbf{x})$ is the closed-loop dynamics with control input $\boldsymbol{\alpha}(\mathbf{x})$, unlike $f_0(\mathbf{x})$ defined in (7a). Further, $\underline{f}_0(\mathbf{x})$ is independent of $\boldsymbol{\nu}(\mathbf{x})$ in (19). This allows separate construction of $\boldsymbol{\alpha}(\mathbf{x})$ independent of $\boldsymbol{\nu}(\mathbf{x})$ and is exploited in the following step.

Step 2: Construct $\boldsymbol{\alpha}(\mathbf{x})$ to ensure $\underline{f}_0(\mathbf{x})$ is stable in the Lyapunov sense. This step ensures the energy of the system remains bounded. Note that during construction of $\boldsymbol{\alpha}(\mathbf{x})$ the control function $\boldsymbol{\nu}(\mathbf{x}) = \mathbf{0}$ identically.

Step 3: Define a dummy output $\mathbf{h}(\mathbf{x}, \boldsymbol{\nu}(\mathbf{x})) \in \mathbb{R}^m$ for the system in (19) to ensure that it becomes passive through the input $\boldsymbol{\nu}(\mathbf{x})$, or equivalently, satisfies Definition II.1 or Theorem 1 with a positive-definite storage function.

Step 4: Finally, construct the control function $\boldsymbol{\nu}(\mathbf{x})$ to satisfy

$$\boldsymbol{\nu}(\mathbf{x}) = -\mathbf{h}(\mathbf{x}, \boldsymbol{\nu}(\mathbf{x})). \tag{20}$$

From previous work on stabilization of passive systems [8] it is known that if the dynamics (19) along with the output definition in Step 3 is zero-state detectable, then the control

given in (20) globally asymptotically stabilizes the system. Zero-state detectability states that if the output is identically zero, then the state vector approaches the origin in time. The detectability properties of $\underline{\Sigma}$ can be verified through accessibility type conditions summarized below.

Definition III.1. *[8] Suppose the system $\underline{\Sigma}$ is passive with $C^r(r \geq 1)$ storage function V, which is positive definite and proper. Then, $\underline{\Sigma}$ is zero-state detectable if $\Omega \cap S = \{0\}$ where the distribution*

$$D = span\left\{ad^k_{\underline{f}_0}\underline{g}^0_i : 0 \leq k \leq n-1, 1 \leq i \leq m\right\}$$

and two sets Ω and S, associated with D be defined as

$$\Omega = \left\{\mathbf{x} \in N \subseteq \mathbb{R}^n : \mathcal{L}^k_{\underline{f}_0} V(\mathbf{x}) = 0, k = 1, \ldots, r\right\}, \tag{21}$$

$$S = \left\{\mathbf{x} \in N \subseteq \mathbb{R}^n : \mathcal{L}^k_{\underline{f}_0} \mathcal{L}_\tau V(\mathbf{x}) = 0, \forall \tau \in D, k = 0, 1, \ldots, r-1\right\} \tag{22}$$

The following proposition proves that $\underline{\Sigma}$ described in (1) and equivalently in (19) is asymptotically stabilized by the proposed control algorithm.

Proposition 1. *Suppose V is a C^2 positive-definite Lyapunov function and the functions $\boldsymbol{\alpha}(\mathbf{x})$ and $\boldsymbol{\nu}(\mathbf{x})$ satisfy Steps 1-4 with output $\mathbf{h}(\mathbf{x}, \boldsymbol{\nu}(\mathbf{x})) = [\mathcal{L}_{\underline{g}} V]^T$. If $\Omega \cap S = \{0\}$ then the control $\mathbf{u}(\mathbf{x}) = \boldsymbol{\alpha}(\mathbf{x}) + \boldsymbol{\nu}(\mathbf{x})$ asymptotically stabilizes the system $\underline{\Sigma}$.*

Proof. Asymptotic stabilization is shown using LaSalle's invariant principle and Lyapunov's direct method. The rate of change of the Lyapunov function about the trajectories of $\underline{\Sigma}$ given in (19) is

$$\dot{V} = \mathcal{L}_{\underline{f}_0} V + \mathcal{L}_{\underline{g}} V \boldsymbol{\nu}(\mathbf{x}). \tag{23}$$

Then, through construction of $\boldsymbol{\alpha}(\mathbf{x})$

$$\dot{V} \leq \mathcal{L}_{\underline{g}} V \boldsymbol{\nu}(\mathbf{x}). \tag{24}$$

By Definition II.1 and Theorem 1 (24) is passive with the output $\mathbf{y} = \left(\mathcal{L}_{\underline{g}} V\right)^T$. With $\Omega \bigcap S = \{0\}$, this passive system is zero-state detectable and $\underline{\Sigma}$ is asymptotically stabilized by input $\boldsymbol{\nu}(\mathbf{x}) = -\mathcal{L}_{\underline{g}} V$. This completes the proof. □

Proposition 1 is a powerful result that guarantees asymptotic stabilization for all non-affine nonlinear systems. This method of control synthesis is general and relies upon separate construction of stiffness and damping functions $\boldsymbol{\alpha}(\mathbf{x})$ and $\boldsymbol{\nu}(\mathbf{x})$ respectively. The construction of $\boldsymbol{\nu}(\mathbf{x})$ has received considerable attention in the literature under the label 'passivity-based control'. The requirements of zero-state detectability is a consequence of employing pure output feedback for passive systems [8], [10], [11] which can be relaxed by use of other methods for control of open-loop stable systems.

IV. Numerical Example: One-Dimensional Non-Affine Unstable Dynamics

The purpose of this section is to verify the theoretical developments through an open-loop unstable non-affine system. The example considered is a polynomial system of degree three given in (2). The control law for this example was developed through analytic root solving techniques in [12]. Here an alternate control law formulation is presented to globally stabilize the origin.

1) Controller Design: The feedback control of the form (17) is constructed in four steps.

Step 1: The vector fields corresponding to definitions (18) for the system given in (2) are:

$$\underline{f}_0(x) = x - 2\alpha^3(x) \tag{25a}$$

$$\underline{g}\,(x, \nu(x)) = -6\alpha^2(x) - 6\alpha(x)\nu(x) - 2\nu^2(x) \tag{25b}$$

$$\underline{g}^0(x) = -6\alpha^2(x). \tag{25c}$$

Step 2: The following choice for function $\alpha(x)$ ensures Lyapunov stability of $\underline{f}_0(x)$:

$$\alpha(x) = \begin{cases} \frac{1}{\sqrt[3]{2}} x & \text{if } |x| \geq 1; \\ -\frac{1}{\sqrt[3]{2}} & \text{if } -1 < x < 0; \\ 0 & \text{if } x = 0\ ; \\ \frac{1}{\sqrt[3]{2}} & \text{if } 0 < x < 1. \end{cases} \tag{26}$$

Using $\alpha(x)$ defined above the dynamics $\underline{f}_0(x)$ become

$$\underline{f}_0(x) = \begin{cases} x - x^3 & \text{if } |x| \geq 1; \\ x + 1 & \text{if } -1 < x < 0; \\ 0 & \text{if } x = 0\ ; \\ x - 1 & \text{if } 0 < x < 1. \end{cases} \tag{27}$$

Note that $\underline{f}_0(x)$ described in (27) has three stable fixed points: $x = -1$, $x = 0$, and $x = 1$. Thus the dynamics of the system (2) are rendered stable for all time.

Steps 3 & 4: These steps construct the control input $\nu(x)$ that enforces stability of the origin. Control laws for such a class of system have been addressed by passivity-based methods. Following the formulation given in [10] control input $\nu(x)$ is constructed as

$$\nu(x) = -\frac{\gamma(x)\mathcal{L}_{g^0} V(x)}{1 + |\mathcal{L}_{g^0} V(x)|^2} \tag{28}$$

where $\gamma(x) = \frac{\beta}{1+x^2(1+4+36\alpha^2(x))^2}$, and $\mathcal{L}_{g_0} V(x)$ is the Lie derivative of $V(x)$ along $[0; \underline{g}^0(x)]$. The design parameter $0 < \beta < 1$ bounds the control input.

Proposition 1 guarantees that the control input $\alpha(x) + \nu(x)$ asymptotically stabilizes an open-loop unstable stable system if $\Omega \bigcap S = \{0\}$. A routine calculation shows that $\mathcal{L}_{\underline{f}_0} V(x) = 0$ for $\Omega = \{-1, 0, 1\}$. Additionally,

$$0 = \mathcal{L}_{\underline{g}^0} V(x) = -6x\alpha^2(x) \tag{29}$$

$$0 = \mathcal{L}_{[\underline{f}_0, \underline{g}^0]} V(x) \tag{30}$$

is satisfied for $x = 0$ so $\Omega \bigcap S = \{0\}$ for all $x \in \mathbb{R}$. *Hence it can be concluded that the control form $\alpha(x) + \nu(x)$ globally asymptotically stabilizes the origin*. Reference [12] designed $u = \sqrt[3]{x}$ as the control law for the prescribed system using inversion, which only locally regulates the system (2).

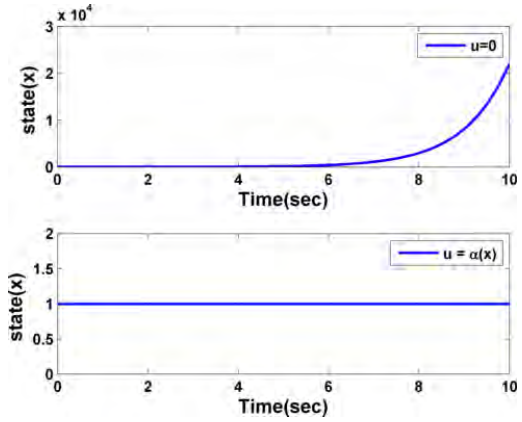


Fig. 2. System response of (2) for $u = 0$ and $u = \alpha(x)$

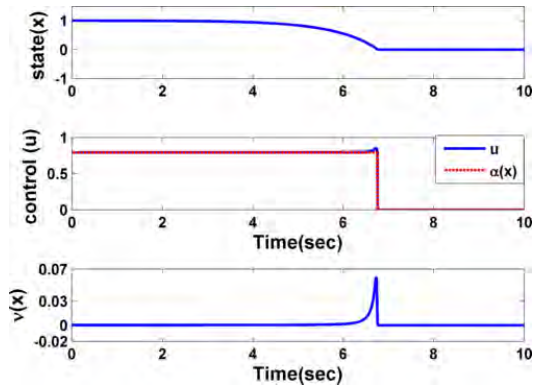


Fig. 3. Closed-loop system response of (2) and control effort

2) Results and Discussion: The proposed control law given in (26) and (28) was validated in simulation with the design parameter β set to 0.9. The initial condition was chosen as $x(0) = 1$. The behaviour of the open-loop system and the system with control input $u = \alpha(x)$ is presented in Fig. 2. As expected the open-loop behaviour is unstable and the system with $u = \alpha(x)$ stays at $x = 1$ for all time. The closed-loop response is shown in Fig. 3. The initial magnitude of the control input $\nu(x)$ is small (specifically $\nu(x) = 0.00029$) but greater than zero to ensure the state of the system becomes less than 1. It is difficult to see but in the figure at time $t = 2$seconds the state is $x(2) = 0.993$. The control is dominated by $\alpha(x)$ since the dynamics $\underline{f}_0(x)$ inherently push the system toward the origin. By construction in (28) the magnitude of $\nu(x)$ increases when the state nears the origin so as to asymptotically regulate the dynamics. This is consistent with earlier conclusions that high-gain feedback is required to stabilize the origin. Thereafter the control is turned off and the system stays at the origin for all future time. Note that the discontinuous nature of the control is an artifact of the choice of $\alpha(x)$.

V. Conclusions

In this paper a design procedure for analytic construction of control laws for unstable non-affine systems was proposed, and sufficiency conditions for passivity were derived. This work also extended well-established control law design procedures for stable non-affine-in-control systems to unstable non-affine systems, without requiring the control influence to be non-singular throughout the domain of interest.

The proposed control laws are real-time implementable and unlike some switching schemes proposed in literature, do not require immense offline processing. The algorithm is general and can stabilize systems of the form $\dot{x} = x - 2xu^4$ by appropriate design of $\alpha(x)$. Owing to the energy-based concept that is utilized for construction of the control, the results obtained are consistent with the physics of the problem and do not violate system constraints. Numerical examples illustrate that the nature of the control function $\alpha(\mathbf{x})$ is continuous but not differentiable. It is interesting to note that [13][Corollary 5.8.8] proved that any nonlinear system whose linear counterparts are unstable cannot be locally C^1 stabilizable. Importantly, the control laws derived here arrive at this well-known result without making any prior assumptions about the nature of the vector fields.

References

[1] N. Hovakimyan, E. Lavretsky, and A. Sasane, "Dynamic inversion for non-affine-in-control systems via time-scale separation part i," *Journal of Dynamical and Control Systems*, vol. 13, no. 4, pp. 451–465, 2007.

[2] J. Xiang, Y. Li, and W. Wei, "Stabilization of a class of non-affine systems via modelling error compensation," *IET Control Theory and Applications*, vol. 2, pp. 108–116, 2008.

[3] R. Burridge, A. Rizzi, and D. Koditschek, "Sequential composition of dynamically dexterous robot behaviours," *The International Journal of Robotics Research*, vol. 18, pp. 534–555, 1999.

[4] A. Leonessa, W. M. Haddad, and V. Chellaboina, "Nonlinear system stabilization via hierarchical switching control," *IEEE Transactions on Automatic Control*, vol. 46, pp. 17–28, 2001.

[5] C. I. Brynes, A. Isidori, and J. C. Willems, "Passivity, feedback equivalence, and the global stabilization of minimum phase nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 36, no. 11, pp. 1228–1240, 1991.

[6] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Upper Saddle River, NJ: Prentice Hall, December 2001.

[7] D. Hill and P. Moylan, "The stability of nonlinear dissipative systems," *IEEE Transactions on Automatic Control*, vol. 21, no. 5, pp. 708–711, 1976.

[8] W. Lin, "Feedback stabilization of general nonlinear control systems: A passive system approach," *Systems & Control Letters*, vol. 25, no. 1, pp. 41–52, 1995.

[9] R. Sepulchre, M. Janković, and P. Kokotović, *Constructive nonlinear control*, ser. Communications and control engineering. New York, NY: Springer, 1997. [Online]. Available: http://books.google.com/books?id=JKIeAQAAIAAJ

[10] W. Lin, "Global asymptotic stabilization of general nonlinear systems with stable free dynamics via passivity and bounded feedback," *Automatica*, vol. 32, no. 6, pp. 915–924, 1996.

[11] A. S. Shiriaev and A. L. Fradkov, "Stabilization of invariant sets for nonlinear non-affine systems," *Automatica*, vol. 36, no. 11, pp. 1709–1715, 2000.

[12] E. Moulay and W. Perruquetti, "Stabilization of nonaffine systems: A constructive method for polynomial systems," *IEEE Transactions on Automatic Control*, vol. 50, no. 4, pp. 520–526, 2005.

[13] E. D. Sontag, *Mathematical Control Theory: Deterministic Finite Dimensional Systems*, 2nd ed. New York: Springer, 1998, iSBN 0-387-984895.

Necessary Conditions for Feedback Passivation of Nonaffine-in-Control Systems*

Anshu Narang-Siddarth† John Valasek‡

Abstract

It is well understood that an open-loop Lyapunov stable nonaffine-in-control nonlinear system can be asymptotically stabilized through feedback. But stabilizing an open-loop *unstable* nonaffine system remains an open research question. This paper derives the necessary conditions required to render a general open-loop unstable nonlinear system passive through static feedback. It is shown that this is possible only if the system under consideration has relative degree one and is weakly minimum phase through an appropriate output definition. Unlike feedback passivation for affine-in-control nonlinear systems this result is not sufficient. The developments and the essential ideas of the paper are verified for a continuously stirred tank reactor.

1 Introduction.

This paper revisits the problem of stabilizing systems of the following form:

$$\underline{\Sigma} : \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{1.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state, $\mathbf{u} \in \mathbb{R}^m$ is the control input and $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is sufficiently smooth. Throughout the paper it is assumed that a control Lyapunov function for (1.1) exists which sufficiently ensures that the dynamical model given in $\underline{\Sigma}$ is asymptotically controllable [1]. Balakrishnan [2] proved that any such controllable nonlinear system could be transformed into the following *affine* form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \equiv \mathbf{f}_1(\mathbf{x}) + \mathbf{f}_2(\mathbf{x})\mathbf{u}, \tag{1.2}$$

which inspired many of the of nonlinear control techniques that we know today such as feedback linearization, gain-scheduling, sliding-mode control, backstepping and more recently forwarding. However, it is difficult to find a change of coordinates that leads to the linear form given in (1.2). Moreover if such a transformation exists, the resultant set of coordinates may be abstract mathematical quantities and/or lead to discontinuous vector fields that are not desirable from a control standpoint.

Artstein [3] proved that a stabilizable control for continuous time-invariant nonaffine systems of the form (1.1) exists if and only if the Lyapunov function $V(\mathbf{x})$ satisfies

$$\inf_{\mathbf{u}} \nabla V(\mathbf{x})\mathbf{f}(\mathbf{x}, \mathbf{u}) < 0. \tag{1.3}$$

The intuitive idea behind (1.3) is that there exists some sort of 'energy' measure of the states that diminishes along suitably chosen paths and the control input must be chosen to force the system to approach a minimal-energy configuration. This condition is a special case of the Hamilton-Jacobi-Bellman equation [4][Ch.6, Sec.6.3] with time-invariant objective function. It is well-known that this partial differential equation may not always have a solution. Moreover, if a solution exists it may not be unique. This was discussed in Artstein's work and he suggested that nonaffine systems in general cannot be stabilized with continuous feedback. Motivated by Artstein's conclusions, Jayawardhana [5] used pulse-width modulated control signals to stabilize non-interacting mechanical systems.

The fact that discontinuous control cannot be employed for most physical systems has motivated several researchers to explore other feedback solution methods for nonaffine systems. Moulay [6] augmented convexity requirement on the argument of (1.3) to provide sufficiency conditions for the existence of stabilizing continuous controls and developed constructive control laws for a class of single-input second and third order polynomial systems. Lin [7],[8] explored passivity-based methods for smooth open-loop Lyapunov stable nonaffine systems. The central idea of the passivity-based approach is to take advantage of the smoothness of the nonlinear vector field and represent it as a linear combination of affine and nonaffine parts. Upon doing so the controller is designed by assuming that the affine part dominates the closed-loop system stability, and the higher-order terms are always upper-bounded for all admissible states and control inputs.

The control design methods discussed so far pro-

*This work was supported in part by the U.S. Air Force Office of Scientific Research under contract FA9550-08-1-0038 with technical monitor Dr. Fariba Fahroo.

†Postdoctoral Research Associate, Vehicle Systems & Controls Laboratory, Aerospace Engineering, Texas A&M University, College Station, TX 77843-3141, asiddarth@aero.tamu.edu (Corresponding Author)

‡Professor and Director, Vehicle Systems & Controls Laboratory, Aerospace Engineering, Texas A&M University, College Station, TX 77843-3141, valasek@tamu.edu

vide constructive forms for the control variable. But in order to consider higher-order unstable systems several approximation and numerical methods have been explored. The intuitive idea has been to indirectly stabilize the system by varying the control derivative. In order to do so the nonaffine problem given in (1.1) is augmented with the control input dynamics such that the resulting dynamics

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{1.4a}$$

$$\tau \dot{\mathbf{u}} = \boldsymbol{\nu} \tag{1.4b}$$

become *affine* in the input vector $\boldsymbol{\nu}$. The time-constant τ is appropriately chosen such that the control input dynamics evolve faster than the dynamical system under consideration. Hovakimyan [9] designed the new input vector using dynamic inversion motivated by the observation that for a single-state single-input system the following input vector

$$\nu = -\operatorname{sgn}\left(\frac{\partial f}{\partial u}\right)(f(x,u) + ax)\,; \quad a > 0, \quad \frac{\partial f}{\partial u} > 0 \tag{1.5}$$

globally asymptotically stabilizes the system. But (1.5) assumes non-singular control influence which is quite restrictive and not satisfied in general. Similar assumptions were also made in [10], [11] and [12].

Motivated by Sontag's universal formula for affine systems [13] one is lead to the natural question: *Assuming that a control Lyapunov function exists for the dynamic system given in (1.1). Can a constructive control law design procedure be formulated to asymptotically stabilize an unstable nonaffine system?* In [14] the authors pursued this research problem and presented a control design procedure based on feedback passivation introduced in [15] for control-affine systems. The general concept was to use state-feedback to render the system passive and then employ well-established results for stabilizing passive systems. Reference [14] developed sufficiency conditions for passivity and presented a novel method for construction of control laws without making any assumptions about the nature of the control influence. In this paper, the important conditions under which *the dynamical model given in (1.1) can be rendered passive through state-feedback* are derived.

The paper is organized as follows. Section 2 presents the necessary mathematical preliminaries and conditions under which (1.1) can be rendered passive are derived in Section 3. A continuously stirred tank reactor example is studied in Section 4 and conclusions are presented in Section 5.

2 Preliminaries

Consider the following nonlinear dynamical system:

$$\Sigma_1 : \begin{array}{l} \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \\ \mathbf{y} = \mathbf{h}(\mathbf{x}, \mathbf{u}) \end{array} \tag{2.6}$$

with state-space $X = \mathbb{R}^n$, set of input values $U = \mathbb{R}^m$ and set of output values $Y = \mathbb{R}^m$. The set $\mathcal{U}$ of admissible inputs consists of all U-valued piecewise continuous functions defined on $\mathbb{R}$. The functions $\mathbf{f}(.)$ and $\mathbf{h}(.)$ are continuously differentiable maps defined on the open subset $O \subset \mathbb{R}^n$. It is assumed that these vector fields are smooth mappings with at least one equilibrium. Without loss of generality, the origin is chosen as the equilibrium of Σ_1, that is $\mathbf{f}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$ and $\mathbf{h}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$.

DEFINITION 2.1. *A system Σ_1 is said to be passive if there exists a storage function $V(\mathbf{x})$ that satisfies $V(\mathbf{0}) = 0$ and for any $\mathbf{u} \in \mathcal{U}$, $\forall t \geq 0$ and initial condition $\mathbf{x}_0 \in X$*

$$V(\mathbf{x}(t)) - V(\mathbf{x}_0) \leq \int_0^t \mathbf{y}^T(s)\mathbf{u}(s)ds. \tag{2.7}$$

If the storage function is C^r times continuously differentiable with $r \geq 1$ then differentiating both sides of (2.7) gives

$$\dot{V} \leq \mathbf{y}^T\mathbf{u}. \tag{2.8}$$

Definition 2.1 is the mathematical analog of stating that the amount of energy stored in a passive system is less than or equal to the energy being input. The next definition gives the necessary conditions for an input/output nonlinear system Σ_1 to be passive. For convenience, define the following vector fields

$$\mathbf{f}_0(\mathbf{x}) \triangleq \mathbf{f}(\mathbf{x}, \mathbf{0}) \in \mathbb{R}^n, \tag{2.9a}$$

$$\mathbf{h}_0(\mathbf{x}) \triangleq \mathbf{h}(\mathbf{x}, \mathbf{0}) \in \mathbb{R}^m. \tag{2.9b}$$

In the above definitions $\mathbf{f}_0(\mathbf{x})$ represents the open-loop dynamics of the dynamical system Σ_1 while $\mathbf{h}_0(\mathbf{x})$ is the output of Σ_1 at zero-input. Using these introduced notations and the fact that the vector fields in Σ_1 are smooth, the nonlinear dynamical system is equivalently represented as

$$\dot{\mathbf{x}} = \mathbf{f}_0(\mathbf{x}) + \mathbf{g}(\mathbf{x}, \mathbf{u})\mathbf{u} \tag{2.10a}$$

$$\mathbf{h}(\mathbf{x}, \mathbf{u}) = \mathbf{h}_0(\mathbf{x}) + \mathbf{j}(\mathbf{x}, \mathbf{u})\mathbf{u}, \tag{2.10b}$$

where the following identities have been used:

$$\mathbf{f}(\mathbf{x}, \mathbf{u}) - \mathbf{f}_0(\mathbf{x}) = \left(\int_0^1 \frac{\partial \mathbf{f}(\mathbf{x}, \gamma)}{\partial \boldsymbol{\gamma}}\bigg|_{\boldsymbol{\gamma} = \theta\mathbf{u}} d\theta\right)\mathbf{u}(\mathbf{x}) \triangleq \mathbf{g}(\mathbf{x}, \mathbf{u})\mathbf{u} \tag{2.11}$$

$$\mathbf{h}(\mathbf{x}, \mathbf{u}) - \mathbf{h}_0(\mathbf{x}) = \left(\int_0^1 \frac{\partial \mathbf{h}(\mathbf{x}, \gamma)}{\partial \boldsymbol{\gamma}}\bigg|_{\boldsymbol{\gamma} = \theta\mathbf{u}} d\theta\right)\mathbf{u}(\mathbf{x}) \triangleq \mathbf{j}(\mathbf{x}, \mathbf{u})\mathbf{u}. \tag{2.12}$$

Hence the vector fields $\mathbf{g}(\mathbf{x},\mathbf{u})$ and $\mathbf{j}(\mathbf{x},\mathbf{u})$ capture the effect of the control input on the motion of the dynamical system states and the output. Recall, for control-affine systems these vector fields are independent of the control input vector. Using smoothness of the vector $\mathbf{g}(\mathbf{x},\mathbf{u})$, (2.10a) can be further decomposed as

$$\dot{\mathbf{x}} = \mathbf{f}_0(\mathbf{x}) + \mathbf{g}_0(\mathbf{x})\mathbf{u} + \sum_{i=1}^{m} u_i \left[\mathbf{R}_i(\mathbf{x},\mathbf{u})\mathbf{u}\right] \tag{2.13}$$

with $\mathbf{R}_i(\mathbf{x},\mathbf{u}) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n\times m}$, being a smooth map for $1 \le i \le m$ and

$$\mathbf{g}_i^0(\mathbf{x}) = \mathbf{g}_i(\mathbf{x},\mathbf{0}) = \frac{\partial \mathbf{f}}{\partial u_i}(\mathbf{x},\mathbf{0}) \in \mathbb{R}^n; \quad 1 \le i \le m \tag{2.14}$$

$$\mathbf{g}_0(\mathbf{x}) = \frac{\partial \mathbf{f}}{\partial \mathbf{u}}(\mathbf{x},\mathbf{0}) = \left[\mathbf{g}_1^0(\mathbf{x}),\ldots,\mathbf{g}_m^0(\mathbf{x})\right] \in \mathbb{R}^{n\times m} \tag{2.15}$$

The vector field $\mathbf{g}_i^0(\mathbf{x})$ defines the influence of input u_i on the system about the origin and is collected for all inputs under the vector $\mathbf{g}_0(\mathbf{x})$.

For convenience let $V : \mathbb{R}^n \to \mathbb{R}$ be a $C^r (r \ge 1)$ storage function and the expression

$$\mathcal{L}_{\mathbf{f}_0} V = \left\langle \frac{\partial V}{\partial \mathbf{x}}, \mathbf{f}_0(\mathbf{x}) \right\rangle \tag{2.16}$$

represent the Lie derivative of the functional V along the vector field $\mathbf{f}_0(\mathbf{x})$.

DEFINITION 2.2. *[7]. Let* $\Omega_1 \triangleq \{\mathbf{x} \in \mathbb{R}^n : \mathcal{L}_{\mathbf{f}_0} V(\mathbf{x}) = 0\}$. *Necessary conditions for* Σ_1 *to be passive with a* C^2 *storage function* V *are*

(i) $\mathcal{L}_{\mathbf{f}_0} V(\mathbf{x}) \le 0$,

(ii) $\mathcal{L}_{\mathbf{g}_0} V(\mathbf{x}) = \mathbf{h}_0^T(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega_1$,

(iii) $\sum_{i=1}^{n} \frac{\partial^2 f_i}{\partial \mathbf{u}^2}(\mathbf{x},\mathbf{0}).\frac{\partial V}{\partial x_i} \le 2\mathbf{j}^T(\mathbf{x},\mathbf{0}) \quad \forall \mathbf{x} \in \Omega_1$,

where $f_i(\mathbf{x},\mathbf{u})$ *is the ith component of the vector function* $\mathbf{f}(\mathbf{x},\mathbf{u})$.

If the storage function was positive-definite property (i) would be analogous to Lyapunov's condition $\dot{V} \le 0$ for bounded stability. The other conditions in Definition 2.2 follow directly from Definition 2.1 by noticing that the difference $\frac{\partial V}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x},\mathbf{u}) - \mathbf{h}^T(\mathbf{x},\mathbf{u})\mathbf{u}$ attains its maximum at $\mathbf{u} = \mathbf{0}$ on the set Ω_1.

3 Feedback Equivalence to a Passive System/Feedback Passivation

In this section the conditions under which the following system

$$\Sigma_2 : \begin{array}{l} \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{u}) \\ \mathbf{y} = \mathbf{h}(\mathbf{x}) \end{array} \tag{3.17}$$

is feedback equivalent to a passive system with positive definite storage function $V(\mathbf{x})$ are derived. These conditions are developed to exploit the following interesting stabilizing property of passive systems. Assume that Σ_2 is passive and zero-state observable. This means that if the output $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ is zero, then the state is identically zero. With this property the following theorem states that a passive system is globally stabilized purely by output feedback.

THEOREM 3.1. *[16][theorem 14.4] If* Σ_2 *is*

(i) passive with a radially unbounded positive definite storage function and

(ii) zero-state observable

then the origin $\mathbf{x} = \mathbf{0}$ *can be globally stabilized by* $\mathbf{u} = -\phi(\mathbf{y})$, *where* ϕ *is any locally Lipschitz function such that* $\phi(\mathbf{0}) = \mathbf{0}$ *and* $\mathbf{y}^T\phi(\mathbf{y}) > 0$ *for all* $\mathbf{y} \ne \mathbf{0}$.

The control in Theorem 3.1 has been formulated to ensure the passivity condition in Definition 2.1 holds globally. Then the zero-state observable property helps conclude that the origin is the largest invariant set and hence the global equilibrium of the closed-loop system. In order to use this powerful result for control design, conditions under which systems can be made passive need to be studied. The first result toward this end, studies the relative degree of a passive system. Relative degree of a system is the number of times the output must be differentiated for the input to appear explicitly. The following definition expresses this condition using Lie derivatives.

DEFINITION 3.1. *The system* Σ_2 *is said to have a relative degree* $(r_1, r_2, \ldots, r_m)$ *at a point* $(\mathbf{x}_0, \mathbf{u}_0)$ *if:*

(i) $\frac{\partial}{\partial u}\left[\mathcal{L}_{\mathbf{f}}^k h_i(\mathbf{x})\right] = 0$ *for all* $1 \le i \le m$, $\mathbf{x}$ *in the neighbourhood of* $\mathbf{x}_0$ *and all* $\mathbf{u}$ *in the neighbourhood of* $\mathbf{u}_0$ *and all* $k < r_i$,

(ii) $\frac{\partial}{\partial \mathbf{u}}\left[\mathcal{L}_{\mathbf{f}}^{r_i} h_i(\mathbf{x})\right]\big|_{(\mathbf{x}_0,u_0)} \ne 0$.

Note the relative degree of a nonlinear system is a local concept defined about the point $(\mathbf{x}_0, \mathbf{u}_0)$ and also depends on the domain of control. This dependence is a result of the non-affinity of the system. Next a lemma is derived that will help determine the relative degree of Σ_2.

LEMMA 3.1. *The origin belongs to the set* Ω_1 *given in Definition 2.2.*

Proof. Consider the open-loop system Σ_2. The necessary condition for passivity with positive definite storage function is

$$\mathcal{L}_{\mathbf{f}_0} V(\mathbf{x}) \le 0.$$

This indicates that the system is stable in the Lyapunov sense. By Lasalle's theorem [17] it is known that the state of this open-loop system will enter the set $\{\mathbf{x} \in \mathbb{R}^n : \mathcal{L}_{\mathbf{f}_0} V(\mathbf{x}) = 0\}$. This is exactly the set Ω_1 in Definition 2.2. This result also can be shown by Barbalat's lemma [18].

Further, the set Ω_1 contains the invariant sets of the system. Since origin is the fixed-point of the system Σ_1, it is concluded that it belongs to the set Ω_1. This completes the proof. □

The next theorem analyzes the relative degree of the passive system Σ_2.

THEOREM 3.2. *Suppose Σ_2 is passive with a C^2 storage function V which is positive definite. If $\mathbf{g}_0(\mathbf{0})$ and $\frac{\partial h}{\partial \mathbf{x}}(\mathbf{0})$ have full rank, then Σ_2 has relative degree $(1, 1, \ldots, 1)$ at $(\mathbf{x} = 0, \mathbf{u} = 0)$.*

Proof. The relative degree of Σ_2 is one if $\left[\frac{\partial \dot{\mathbf{y}}}{\partial \mathbf{u}}\right](\mathbf{0}, \mathbf{0})$ is non-singular, or

$$\frac{\partial \dot{\mathbf{y}}}{\partial \mathbf{u}}(\mathbf{0}, \mathbf{0}) = \left\{ \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \mathbf{g}_0(\mathbf{x}) + \frac{\partial}{\partial \mathbf{u}} \left[\frac{\partial \mathbf{h}}{\partial \mathbf{x}} \left[\sum_{i=1}^{m} u_i \mathbf{R}_i(\mathbf{x}, \mathbf{u}) \right] \mathbf{u} \right] \right\}_{(\mathbf{0},\mathbf{0})} \tag{3.18}$$
$$= \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \mathbf{g}_0(\mathbf{0})$$
$$= \mathcal{L}_{\mathbf{g}_0} \mathbf{h}(\mathbf{0})$$

is $m \times m$ and non-singular. Hence conditions for which (3.18) holds true need to be determined. This is carried out in the following two steps.

First, since Σ_2 is passive it satisfies the necessary conditions given in Definition 2.2. Further, property (ii) in Definition 2.2 is defined only for set Ω_1. From Lemma 3.1 it is known that origin belongs to the set Ω_1 and hence

$$\frac{\partial}{\partial \mathbf{x}} \left[\mathbf{g}_0^T(\mathbf{x}) \frac{\partial V}{\partial \mathbf{x}} \right] \mathbf{g}_0(\mathbf{x}) = \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \mathbf{g}_0(\mathbf{x}) \tag{3.19}$$

is satisfied at $\mathbf{x} = \mathbf{0}$. Differentiating and using the fact that $\frac{\partial V}{\partial \mathbf{x}}(\mathbf{0}) = 0$ in (3.19)

$$\mathbf{g}_0^T(\mathbf{0}) \frac{\partial^2 V}{\partial \mathbf{x}^2}(\mathbf{0}) \mathbf{g}_0(\mathbf{0}) = \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \mathbf{g}_0(\mathbf{0}). \tag{3.20}$$

The rest of the proof proceeds similar to Proposition 2.44 given in [19]. The Hessian $\frac{\partial^2 V}{\partial \mathbf{x}^2}(\mathbf{0})$ is symmetric positive definite by properties of the storage function and can be factored as $R^T R$ with some matrix R. Then,

$$\mathbf{g}_0^T(\mathbf{0}) R^T R(\mathbf{0}) \mathbf{g}_0(\mathbf{0}) = \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \mathbf{g}_0(\mathbf{0}). \tag{3.21}$$

Since $\frac{\partial \mathbf{h}}{\partial \mathbf{x}}(\mathbf{0}) = \mathbf{g}_0^T(\mathbf{0}) R^T R(\mathbf{0})$ is assumed to be full rank, $R\mathbf{g}_0(\mathbf{0})$ has full rank. Hence it is concluded that $\frac{\partial \mathbf{h}}{\partial \mathbf{x}} \mathbf{g}_0(\mathbf{0})$ is $m \times m$ and full rank. This completes the proof. □

REMARK 3.1. *For an affine system, the conditions of Definition 2.2 are satisfied for all control inputs. Since the relative degree for an affine system does not depend on input, Theorem 3.2 consequently reduces to Proposition 2.44 [19].*

The next result examines the nature of the zero dynamics of Σ_2.

THEOREM 3.3. *Suppose Σ_2 is passive with a C^2 storage function V which is positive definite. If $\mathbf{g}_0(\mathbf{0})$ and $\frac{\partial \mathbf{h}}{\partial \mathbf{x}}(\mathbf{0})$ have full rank, then zero dynamics of Σ_2 locally exist about $(\mathbf{x} = \mathbf{0}, \mathbf{u} = \mathbf{0})$ and is weakly minimum phase.*

Proof. From Theorem 3.2, Σ_2 has a well-defined relative degree and local zero dynamics exist. Let the set $\Omega_2 = \{\mathbf{x} \in \mathbf{R}^n : \mathbf{h}(\mathbf{x}) = \mathbf{0}\}$ define the points on the zero-output manifold. By definition of Σ_2 this set contains the origin. By Lemma 3.1 origin is also contained in the set Ω_1. Thus, in order to study the local nature of the zero dynamics about the origin, only those state trajectories that fall in the intersection set $\Omega_2 \bigcap \Omega_1$ need to be considered. On these set of points properties (i) through (ii) of Definition 2.2 hold. Hence,

$$\begin{aligned} \dot{V} &= \mathcal{L}_{\mathbf{f}(\mathbf{x},\mathbf{u})} V \\ &= \mathcal{L}_{\mathbf{f}_0} V + \mathcal{L}_{\mathbf{g}_0} V \mathbf{u} + \mathbf{u}^T \mathcal{L}_{\mathbf{R}(\mathbf{x},\mathbf{u})} V \mathbf{u} \\ &= \mathbf{u}^T \mathcal{L}_{\mathbf{R}(\mathbf{x},\mathbf{u})} V \mathbf{u}. \end{aligned} \tag{3.22}$$

By Definition 2.1, for passive systems $\dot{V} \leq \mathbf{y}^T \mathbf{u}$. Furthermore, this condition becomes $\dot{V} \leq 0$ on the set $\Omega_2 \bigcap \Omega_1$. This inference along with condition (3.22) implies that the origin is Lyapunov stable and hence zero dynamics is weakly minimum phase. This completes the proof.□

Theorems 3.2 and 3.3 together give the necessary conditions for feedback equivalence to a passive system. This result is summarized by the following theorem.

THEOREM 3.4. *Suppose $\mathbf{g}_0(\mathbf{0})$ and $\frac{\partial \mathbf{h}}{\partial \mathbf{x}}(\mathbf{0})$ have full rank. The necessary conditions for transforming Σ_2 into a passive system with C^2 positive definite storage function V using static state-feedback compensation are:*

(i) Σ_2 has relative degree $\{1, 1, \ldots, 1\}$ and

(ii) is weakly minimum phase

Proof. From Theorem 3.2 and Theorem 3.3 it is known that the resulting system will have relative degree $(1, 1, \ldots)$ with weakly minimum phase zero dynamics. Further, it is well understood that relative degree and zero dynamics are invariant under static feedback [20][Lemma 2.4]. Hence the conditions in the proof follow. □

Theorem 3.4 extends the powerful feedback equivalence approach to general nonlinear systems. It provides necessary conditions for a system to be made passive by feedback under mild restrictions. The equivalent theorem for affine systems derived in [15] shows that Theorem 3.4 is also sufficient for feedback passivity. But the topological and nonlinear nature of nonaffine systems hinders this result to be sufficient.

4 Application to Continuously Stirred Tank Reactor

The purpose of this section is to show how conditions given by Theorem 3.4 can be applied to test whether or not a system can be rendered passive through static-feedback. The nonaffine system under consideration is a constant volume reactor and the objective is to stabilize the system about its equilibrium by adjusting the coolant flow rate. The system [21] is represented as

$$\dot{x}_1 = 1 - x_1 - a_0 x_1 \exp^{(-10^4/x_2)} \tag{4.23a}$$

$$\dot{x}_2 = 350 - x_2 + a_1 x_1 \exp^{(-10^4/x_2)} + a_3 u(1 - \exp^{(-a_2/u)})(350 - x_2) \tag{4.23b}$$

and where $0 < x_1 < 1$ is the concentration of the tank in mol/l, $x_2 > 350$ is the temperature of the tank in $^\circ K$ and $u \geq 0$ is the coolant flow rate in mol/min. The system parameters [21] are given in Table 1. The control influence in (4.23) is nonlinear in the control and not monotonic in any variable. This trend is presented in Fig. 1. Owing to this nonlinear behaviour previous studies have used neural-network based control designs to stabilize the concentration of the reactor [21], [22].

4.1 Test for Feedback Passivation The *first step* is to cast the system into form of Σ_2. However,

Table 1: Continuously stirred tank reactor model parameters

Parameter	Value
a_0	$7.2 \times 10^{10} min^{-1}$
a_1	1.44×10^{13}
a_2	6.987×10^2
a_3	0.01

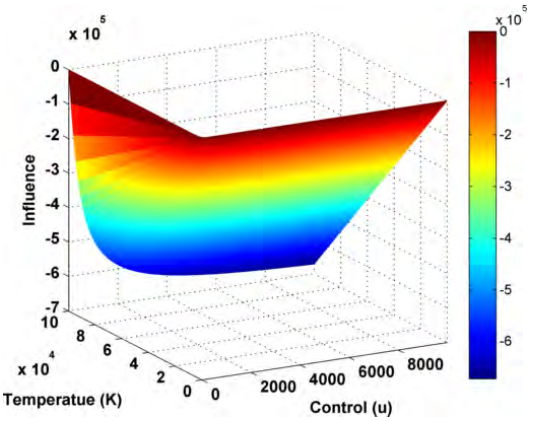


Figure 1: Control influence of the continuously stirred tank reactor

the origin is not the equilibrium of the system given in (4.23). The equilibrium solutions are obtained by solving the following transcendental equations:

$$0 = 1 - x_{1*} - a_0 x_{1*} \exp^{(-10^4/x_{2*})} \tag{4.24a}$$

$$0 = 350 - x_{2*} + a_1 x_{1*} \exp^{(-10^4/x_{2*})}. \tag{4.24b}$$

Rewrite the concentration as $x_{1*} = 1/(1 + a_0 \exp^{(-10^4/x_{2*})})$ and solve for roots of

$$0 = 350 - x_{2*} + \exp^{(-10^4/x_{2*})}[350a_0 + a_1 - a_0 x_{2*}]. \tag{4.25}$$

The algebraic equation given in (4.25) has a unique root $x_{2*} = 549.01257025^\circ K$. Using (4.24) the unique root for concentration is $x_{1*} = 0.001128849277 mol/l$. Define the states $e_1 = x_1 - x_{1*}$ and $e_2 = x_2 - x_{2*}$ to shift the equilibrium to origin. Routine calculation gives the following system:

$$\dot{e}_1 = c - e_1 - a_0(x_{1*} + e_1) \exp^{(-10^4/(x_{2*}+e_2))} \tag{4.26a}$$

$$\dot{e}_2 = d - e_2 + a_1(e_1 + x_{1*}) \exp^{(-10^4/(x_{2*}+e_2))} + a_3 u(1 - \exp^{(-a_2/u)})(350 - x_{2*} - e_2) \tag{4.26b}$$

$$y = e_2 \tag{4.26c}$$

where $c = a_0 x_{1*} \exp^{(-10^4/x_{2*})}$ and $d = -a_1 x_{1*} \exp^{(-10^4/x_{2*})}$ and a dummy output has been defined. Thus comparing (4.26) with Σ_2 gives the following vector field definitions

$$\mathbf{f}_0(\mathbf{e}) = \begin{bmatrix} c - e_1 - a_0(x_{1*} + e_1) \exp^{(-10^4/(x_{2*}+e_2))} \\ d - e_2 + a_1(e_1 + x_{1*}) \exp^{(-10^4/(x_{2*}+e_2))} \end{bmatrix}, \tag{4.27}$$

$$\mathbf{g}_0(\mathbf{e}) = \begin{bmatrix} 0 \\ a_3(350 - x_{2*} - e_2) \end{bmatrix}, \tag{4.28}$$

and

$$\mathbf{h}_0(\mathbf{e}) = e_2. \tag{4.29}$$

In the *second* step necessary conditions given in Theorem 3.4 are verified. Notice with the output defined by (4.29) property (i) is satisfied. Furthermore, $\mathbf{g}_0(\mathbf{0}) = [0, a_3(350 - x_{2*})]^T$ and $\frac{\partial \mathbf{h}_0}{\partial \mathbf{e}}(\mathbf{0}) = [0, 1]$ have full rank. Finally, to verify that the (4.26) is weakly minimum phase set the output $y = e_2 = 0$ and its derivative $\dot{e}_2 = 0$. This leaves the following internal dynamics

$$\dot{e}_1 = a_0 x_{1*} \exp^{(-10^4/x_{2*})} - e_1 - a_0(x_{1*} + e_1) \exp^{(-10^4/x_{2*})} \tag{4.30}$$

which equivalently becomes a linear homogeneous differential equation

$$\dot{e}_1 = -e_1 - a_0 e_1 \exp^{(-10^4/x_{2*})}. \tag{4.31}$$

From Table 1 and properties of exponential function, it can be concluded that the internal dynamics given in (4.31) are exponentially stable and the continuously stirred tank reactor given in (4.23) can be rendered passive through feedback.

5 Conclusions

In this paper necessary conditions for analytical construction of control for unstable nonaffine systems were derived. This work extended the applicability of the well-established feedback passivation control law design procedures to unstable nonaffine systems. Furthermore, the results presented do not require the control influence to be non-singular throughout the domain of interest. These conditions along with results given in [8] can be employed for asymptotic stabilization of a general nonaffine system with static compensation unlike some of the switching schemes [23] that require immense off-line processing.

References

[1] E. D. Sontag, *Mathematical Control Theory: Deterministic Finite Dimensional Systems*. New York: Springer, 2nd ed., 1998. ISBN 0-387-984895.

[2] A. Balakrishnan, "On the controllability of a nonlinear system," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 55, pp. 465–468, 1966.

[3] Z. Artstein, "Stabilization with relaxed controls," *Nonlinear Analysis, Theory, Methods & Applications*, vol. 7, no. 11, pp. 1163–1173, 1983.

[4] F. L. Lewis and V. L. Syrmos, *Optimal Control*. New York: John Wiley & Sons Inc, 2nd ed., 1995. ISBN 0-471-03378-2.

[5] B. Jayawardhana, "Noninteracting control of nonlinear systems based on relaxed control," in *Proceedings of 49th IEEE Conference on Decision and Control*, (Atlanta), December 2010.

[6] E. Moulay and W. Perruquetti, "Stabilization of non-affine systems: A constructive method for polynomial systems," *IEEE Transactions on Automatic Control*, vol. 50, no. 4, pp. 520–526, 2005.

[7] W. Lin, "Feedback stabilization of general nonlinear control systems: A passive system approach," *Systems & Control Letters*, vol. 25, no. 1, pp. 41–52, 1995.

[8] W. Lin, "Global asymptotic stabilization of general nonlinear systems with stable free dynamics via passivity and bounded feedback," *Automatica*, vol. 32, no. 6, pp. 915–924, 1996.

[9] N. Hovakimyan, E. Lavretsky, and A. Sasane, "Dynamic inversion for non-affine-in-control systems via time-scale separation part i," *Journal of Dynamical and Control Systems*, vol. 13, no. 4, pp. 451–465, 2007.

[10] J. D. Boškoviċ, L. Chen, and R. K. Mehra, "Adaptive control design for nonaffine models arising in flight control," *Journal of Guidance, Control and Dynamics*, vol. 27, no. 2, pp. 209–217, 2004.

[11] H. Du and X. Chen, "Nn-based output feedback adaptive variable structure control for a class of non-affine nonlinear systems: A nonseparation principle design," *Neurocomputing*, vol. 72, pp. 2009–2016, 2009.

[12] R. Ghasemi, M. Menhaj, and A. Afsar, "Output tracking controller for non-affine nonlinear systems with nonlinear output: Fuzzy adaptive approach," in *Proceedings of the 7th Asian Control Conference*, pp. 1073–1078, IEEE, 2009.

[13] E. D. Sontag, "A 'universal' construction of artstein's theorem on nonlinear stabilization," *Systems & Control Letters*, vol. 13, pp. 117–123, 1989.

[14] A. Siddarth and J. Valasek, "A constructive stabilization approach for open-loop unstable non-affine systems," in *Proceedings of American Control Conference*, June 2013.

[15] C. I. Brynes, A. Isidori, and J. C. Willems, "Passivity, feedback equivalence, and the global stabilization of minimum phase nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 36, no. 11, pp. 1228–1240, 1991.

[16] H. K. Khalil, *Nonlinear Systems*. Upper Saddle River, NJ: Prentice Hall, 3 ed., December 2001.

[17] J. Slotine and W. Li, *Applied nonlinear control*. Upper Saddle River, NJ: Prentice Hall, 1991.

[18] P. Ioannou and J. Sun, *Robust Adaptive Control*. Upper Saddle River, NJ: Prentice Hall Inc., 2003.

[19] R. Sepulchre, M. Janković, and P. Kokotović, *Constructive nonlinear control*. Communications and control engineering, New York, NY: Springer, 1997.

[20] H. Sira-Ramirez, "Sliding regimes in general non-linear systems: a relative degree approach," *International*

Journal of Control, vol. 50, no. 4, pp. 1487–1506, 1989.

[21] S. Ge, C. Hang, and T. Zhang, "Nonlinear adaptive control using neural networks and its application to CSTR systems," *Journal of Process Control*, vol. 9, no. 4, pp. 313–323, 1999.

[22] S. S. Ge and J. Zhang, "Neural-network control of nonaffine nonlinear system with zero dynamics by state and output feedback," *IEEE Transactions on Neural Networks*, vol. 14, no. 4, pp. 900–918, 2003.

[23] A. Leonessa, W. M. Haddad, and V. Chellaboina, "Nonlinear system stabilization via hierarchical switching control," *IEEE Transactions on Automatic Control*, vol. 46, no. 1, pp. 17–28, 2001.

AIAA 2013-5234

Nonlinear Adaptive Dynamic Inversion Applied to a Generic Hypersonic Vehicle

Elizabeth Rollins* and John Valasek†
Texas A&M University, College Station, TX 77843-3141
Jonathan A. Muse‡ and Michael A. Bolender§
U.S. Air Force Research Laboratory, Wright-Patterson Air Force Base, OH 45433

Flight control of hypersonic vehicles is challenging because of the wide range of operating conditions encountered and certain aspects unique to high speed flight. A particular safety concern in hypersonic flight is the risk of an inlet unstart, which not only produces a significant decrease in thrust but also results in a change to the aerodynamics and thus can lead to the loss of the vehicle. Previous work on control design for hypersonic vehicles often uses linearized or simplified nonlinear dynamical models of the vehicle, and very little work has been done on recovering from unstart events. Using a generic hypersonic vehicle as a control design and simulation model, this paper develops a nonlinear adaptive dynamic inversion control architecture with a control allocation scheme to track realistic flight path angle trajectories. A robustness analysis is performed on the initial control architecture design, which shows that the control architecture is able to handle time delays, perturbations in stability derivatives, and reduced control surface effectiveness. The control architecture then is evaluated for its ability to handle inlet unstart. Simulation results presented in the paper demonstrate that the approach achieves desired tracking performance while being robust to the particular uncertainties and inlet unstart conditions studied.

I. Introduction

The design of control architectures for hypersonic vehicles is a current area of research in the field of controls. One particular safety concern in hypersonic flight is the risk of an inlet unstart, which can lead to a decrease in thrust and the possible loss of the vehicle. There are three main reasons that cause a hypersonic airbreathing engine to unstart: (1) a flow to the inlet that is slower than the required Mach number for the engine to operate, (2) an altered flow that no longer passes through the throat of the engine because of reasons such as exceeding the limits on angle-of-attack (α) and sideslip angle (β), and (3) an increase in the back pressure in the engine that causes the shock wave to move ahead of the throat [1]. A control architecture for a hypersonic vehicle must be capable of maintaining the aircraft on a controlled trajectory in the event of an inlet unstart.

Many of the previous control designs for hypersonic vehicles have involved the use of linearized models of the aircraft instead of the full nonlinear equations of motion [2], [3], [4], [5]. Annaswamy, et.al. created adaptive controllers for hypersonic vehicles; however, the controllers are designed based on linearized models of the aircraft dynamics and require gain-scheduling for their implementation [2],[3]. Groves, et.al. implemented control designs based on linear models of a hypersonic vehicle for setpoint and regulator tracking [4]. Bolender, et.al. designed adaptive control laws for an experimental hypersonic vehicle based on a linearized model of the longitudinal dynamics of the vehicle [5].

In terms of work with nonlinear models for control design, Johnson, et.al. applied a neural network-based adaptive control architecture to a model of the X-33 vehicle for the generation of ascent and abort trajectories

*Graduate Student, Vehicle Systems & Controls Laboratory, AIAA Student Member, *erollins@tamu.edu.*
†Professor and Director, Vehicle Systems & Controls Laboratory, AIAA Associate Fellow, *valasek@tamu.edu, http://vscl.tamu.edu/valasek.*
‡Research Aerospace Engineer, AFRL/RQQA, and AIAA Member.
§Senior Research Engineer, AFRL/RQQA, and AIAA Associate Fellow.

as well as the control of the aircraft [6]. Fiorentini, et.al. [7] and Parker, et.al. [8] both used simplified nonlinear models of a hypersonic vehicle in their control design that exhibited good tracking performance but a slow response. While Parker, et.al. designed an approximate feedback linearization controller, the controller in that paper is not adaptive; however, a case study of their approximate feedback linearization controller showed that the controller was robust to mild plant parameter variations in the moment of inertia I_{yy}, the vehicle length, and the mass of the vehicle [9].

This paper presents a design of a nonlinear adaptive dynamic inversion controller for a Generic Hypersonic Vehicle (GHV). Because the dynamic equations for the GHV are inherently nonlinear and the aerodynamic and control derivatives for the aircraft have significant uncertainty associated with them, a nonlinear adaptive dynamic inversion control architecture was selected as the preferred control architecture. The design of the control architecture for the GHV involved the nonlinear equations of motion for the vehicle. When the reference trajectories were generated for the simulation, the adaptive dynamic inversion control architecture was altered to use altitude rate $\dot{h}$ instead of α, but the control architecture still used the nonlinear equation for $\dot{h}$ in its design.

The paper is organized as follows. Section II provides a brief overview of the GHV and the proposed control architecture. Section III contains the derivations of the general adaptive dynamic inversion equations that are used in the control architecture. Sections IV and V show how the nonlinear dynamic equations for the GHV are transformed into the form given in Section III for the control architecture. Representative simulation results and a robustness analysis of the nonlinear adaptive dynamic inversion control architecture are given in Section VI. Section VII describes the development of equations to allow realistic trajectories to be generated for the GHV simulation, and the simulation results for these realistic trajectories are shown in Section VIII. Finally, the conclusion and future extensions of this work are given in Section IX.

II. Control Structure for the GHV

The Generic Hypersonic Vehicle (GHV), as shown in Figure 1, is an academic hypersonic aircraft vehicle model created at the Air Force Research Laboratory as a platform for studying control laws. The GHV plant simulation is implemented using a Simulink model that contains the nonlinear, 6-DOF equations of motion for an inelastic hypersonic vehicle. The aerodynamic and thrust forces and moments acting on the vehicle are modeled using look-up tables; the tables for the aerodynamic forces and moments were generated based on computational fluid dynamics data using shock-expansion methods with a viscous correction.

Using two elevons and two ruddervators, it is desired to control angle-of-attack (α), sideslip angle (β), and aerodynamic bank angle (μ). It was decided to command the aerodynamic bank angle (μ) instead of the bank attitude angle (ϕ) in order to ensure that the dynamic inversion is singular only at $\beta = \pm 90$ deg. Figure 2 shows a diagram of the GHV system with the adaptive dynamic inversion controllers. To simplify the process of designing a nonlinear adaptive dynamic inversion control architecture, it is assumed that the aircraft states can be divided into two timescale categories, which are the fast states, which consist of the angular rates p, q, and r as noted in [10], and the slow states, which consist of the angles α, β, and μ. An adaptive dynamic inversion controller first must translate α, β, and μ commands into commands for the body axis rates p, q, and r, which then are passed into another adaptive dynamic inversion controller that determines the corresponding control surface deflections to achieve the desired p, q, and r commands.

The following three sections will describe the equations found in the inversion blocks in Figure 2. See Reference [11] for a description of the equations that are contained in the GHV simulation. For the equations derived in Sections IV and V, the flat, nonrotating earth assumption [12, p. 43] is made. It is acceptable to make this assumption in this case because while the vehicle is flying fast enough for the round rotating Earth effects to be significant, the time scale of the controlled dynamics are sufficiently fast to neglect them.

III. General Adaptive Dynamic Inversion Equations

This section contains the derivation of the control laws for two cases of the adaptive nonlinear dynamic inversion controller. The first case involves dynamic equations containing the same number of controls and controlled variables, and the second case deals with dynamic equations with a greater number of controls than controlled variables. It should be noted in both cases, the general nonlinear equation of the system is assumed to be affine in the control, which is reasonable for small deflection angles.

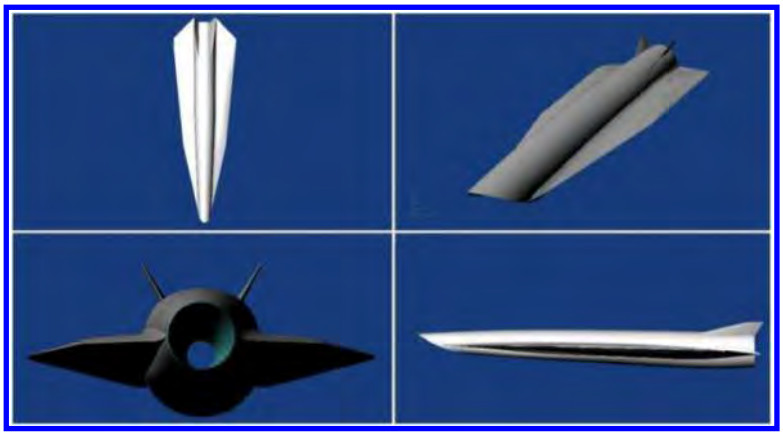


Figure 1. The Generic Hypersonic Vehicle (GHV).

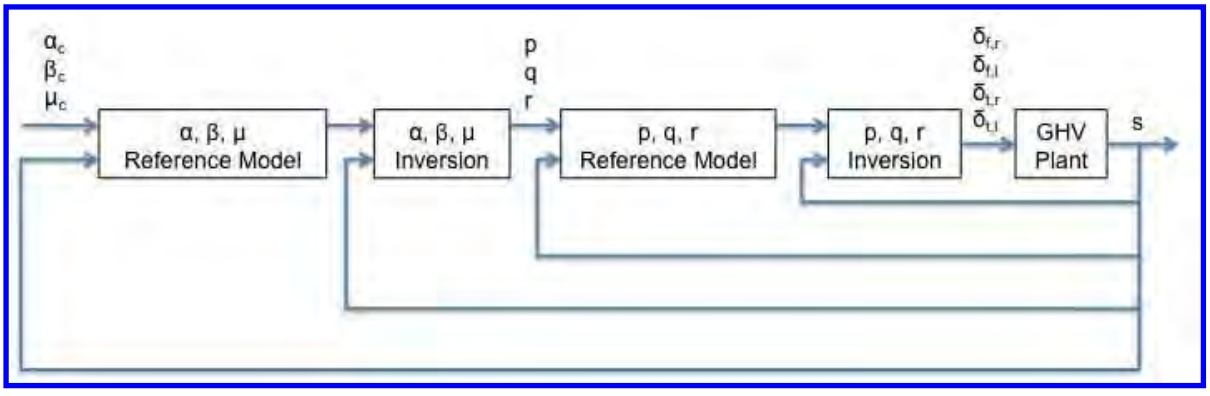


Figure 2. Diagram of the nonlinear adaptive dynamic inversion control architecture for the GHV.

A. Case with Equal Number of Controls and Controlled Variables

Consider a general nonlinear equation of a system in the form

$$\dot{x} = f(x) + g(x)u \tag{1}$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^n$ is the control, and $f(x) : \mathbb{R}^n \mapsto \mathbb{R}^n$ and $g(x) : \mathbb{R}^n \mapsto \mathbb{R}^n$ are locally Lipschitz continuous. It is assumed that $g(x)$ is nonsingular for all $x \in \mathbb{R}^n$. Suppose that the desired reference dynamics for the system are given by

$$\dot{x}_m = Ax_m + Br \tag{2}$$

where $x_m \in \mathbb{R}^n$ is the model state, $r \in \mathbb{R}^n$ is a bounded reference signal, $A \in \mathbb{R}^{n\times n}$ is Hurwitz, and $B \in \mathbb{R}^{n\times n}$. The equation for the error between the reference model and the actual system is

$$e = x_m - x. \tag{3}$$

Taking the time derivative of equation (3) results in

$$\dot{e} = \dot{x}_m - \dot{x} = \dot{x}_m - f(x) - g(x)u. \tag{4}$$

If the control u is chosen to be

$$u = [g(x)]^{-1}[\dot{x}_m - \hat{f}(x) + Ke - \nu] \tag{5}$$

where $\hat{f}(x) : \mathbb{R}^n \mapsto \mathbb{R}^n$ is a model of the plant dynamics, $K \in \mathbb{R}^{n\times n}$ such that $K = K^T > 0$ are the gains on the tracking errors, and $\nu \in \mathbb{R}^n$ is a pseudo-control signal, then substituting equation (5) into equation (4) produces the error dynamics

$$\dot{e} = -f(x) + \hat{f}(x) - Ke + \nu. \tag{6}$$

Defining the error between the model and the actual system as $\Delta = \hat{f}(x) - f(x)$, equation (6) becomes

$$\dot{e} = -Ke + \Delta + \nu. \tag{7}$$

In this paper, it is assumed that Δ can be represented in the form $\Delta = W^T\beta(x; d)$, where $W \in \mathbb{R}^{p\times n}$ is a set of unknown weights, and $\beta \in \mathbb{R}^{p\times 1}$ is a set of known basis functions composed of the states x and a vector d of bounded continuous exogenous inputs. Using this representation for Δ, ν is chosen to be $\nu = -\widehat{W}^T\beta(x; d)$, where $\widehat{W} \in \mathbb{R}^{p\times n}$ is an estimate of the weights. With these definitions, equation (7) can be written as

$$\dot{e} = -Ke - \widetilde{W}^T\beta(x; d) \tag{8}$$

where $\widetilde{W} = \widehat{W} - W$ is the weight estimation error.

The stability of the closed loop system under these assumptions can be established using a candidate Lyapunov function of the form

$$V = e^Te + tr(\widetilde{W}^T{\Gamma_W}^{-1}\widetilde{W}) \tag{9}$$

where $\Gamma_W \in \mathbb{R}^{p\times p}$ with $\Gamma_W = {\Gamma_W}^T > 0$. In order to determine the adaptation law for the parameters in W and to prove that the error between the states of the actual system and the reference model will converge, first, the derivative of equation (9) along the system trajectories is taken, which gives the result

$$\dot{V} = 2e^T\dot{e} + 2tr(\widetilde{W}^T{\Gamma_W}^{-1}\dot{\widehat{W}}^T). \tag{10}$$

Substituting equation (8) into equation (10) produces

$$\dot{V} = -2e^TKe - 2e^T\widetilde{W}^T\beta(x; d) + 2tr(\widetilde{W}^T{\Gamma_W}^{-1}\dot{\widehat{W}}^T). \tag{11}$$

Applying the trace identity that $a^Tb = tr(ba^T)$, equation (11) is determined to be

$$\dot{V} = -2e^TKe + 2tr(\widetilde{W}^T({\Gamma_W}^{-1}\dot{\widehat{W}}^T - \beta(x; d)e^T)). \tag{12}$$

Then, by choosing $\dot{\widehat{W}}$ as

$$\dot{\widehat{W}} = \Gamma_W Proj(\widehat{W}, \beta(x;d)e^T) \tag{13}$$

where Proj represents the projection operator, which is used to maintain specified bounds on the weights [13], $\dot{V}$ can be upper bounded as

$$\dot{V} \leq -2e^T Ke \leq 0 \tag{14}$$

which implies that e is bounded. Because r is bounded by definition above, x_m is bounded. Since e and x_m are bounded, x is bounded. Consequently, $\beta(x;d)$ is bounded as well. In order to use Barbalat's lemma [14] to complete the proof, the second derivative of equation (9) along the system trajectories is taken, which gives the result

$$\ddot{V} = -4e^T K\dot{e}. \tag{15}$$

Substituting equation (8) into equation (15) produces

$$\ddot{V} = -4e^T K(-Ke - \widetilde{W}^T\beta(x;d)). \tag{16}$$

Because e, $\widetilde{W}$, and $\beta(x;d)$ are bounded as proved above, $\ddot{V}$ is bounded, and therefore $\dot{V}$ is uniformly continuous.

Because V is lower bounded, $\dot{V}$ is negative semi-definite, and $\dot{V}$ is uniformly continuous, by Barbalat's lemma $\dot{V} \to 0$ as $t \to \infty$, and thus $e \to 0$ as $t \to \infty$ as desired.

B. Case with a Greater Number of Controls Than Controlled Variables

Specifically for the GHV, the form of the general adaptive dynamic inversion controller in the previous subsection applies to the α, β, and μ inversion component, in which the number of inputs to the system (α, β, μ) is equal to the number of outputs (p, q, r). However, in the p, q, r inversion component, the number of inputs to the system (p, q, r) is not the same as the number of outputs ($\delta_{f,r}$, $\delta_{f,l}$, $\delta_{t,r}$, $\delta_{t,l}$). The fact that the number of outputs is greater than the number of inputs requires a control allocation scheme to be integrated into the inversion control law. To frame the problem in general terms, consider the given nonlinear equation of a system in the form

$$\dot{x} = f(x) + g(x)\Lambda u \tag{17}$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the control, $f(x) : \mathbb{R}^n \mapsto \mathbb{R}^n$ and $g(x) : \mathbb{R}^n \mapsto \mathbb{R}^{n\times m}$ are locally Lipschitz continuous, and $\Lambda \in \mathbb{R}^{m\times m}$ is a constant unknown positive definite matrix. It is assumed that $g(x)$ is full rank for all $x \in \mathbb{R}^n$. Suppose that the desired dynamics of the closed loop system are given by

$$\dot{x}_m = Ax_m + Br \tag{18}$$

where $x_m \in \mathbb{R}^n$ is the model state, $r \in \mathbb{R}^m$ is the bounded reference signal, $A \in \mathbb{R}^{n\times n}$ is Hurwitz, and $B \in \mathbb{R}^{n\times m}$.

The derivation of the control law and the adaptive laws, including one for the unknown control effectiveness matrix Λ, proceeds similarly to the derivation in Subsection A. The equation for the error between the reference model and the actual system is

$$e = x_m - x. \tag{19}$$

Taking the time derivative of equation (19) results in

$$\dot{e} = \dot{x}_m - \dot{x} = \dot{x}_m - f(x) - g(x)\Lambda u. \tag{20}$$

The desired final form for $\dot{e}$ is

$$\dot{e} = -Ke - \widetilde{W}^T\beta(x;d) + g(x)\widetilde{\Lambda}u \tag{21}$$

which is the same as the final form for $\dot{e}$ in Subsection A, except for the final term $g(x)\widetilde{\Lambda}u$. With the appropriate choice of adaptive law for $\widehat{\Lambda}$, the choice of the above final form for $\dot{e}$ will allow the stability of the system to be proven. In order to derive this desired form of $\dot{e}$, first the term $g(x)\widehat{\Lambda}u$ is added and subtracted from equation (20), where $\widehat{\Lambda} \in \mathbb{R}^{m\times m}$ is an estimate of the control effectiveness matrix, and the error equation becomes

$$\dot{e} = \dot{x}_m - f(x) - g(x)\Lambda u + g(x)\widehat{\Lambda}u - g(x)\widehat{\Lambda}u. \tag{22}$$

Let $\widetilde{\Lambda} = \widehat{\Lambda} - \Lambda$, which is the estimation error of the control effectiveness matrix. Then, equation (22) simplifies to

$$\dot{e} = \dot{x}_m - f(x) - g(x)\widehat{\Lambda}u + g(x)\widetilde{\Lambda}u. \tag{23}$$

Because of the fact that the number of controls is greater than the number of controlled variables in this case, there sometimes are infinite choices for u at any instant in time. In order to determine a specific control law for the system, a constrained optimization problem is solved in which the cost function $J = u^T Q u$, where $Q \in \mathbb{R}^{m\times m}$ with $Q = Q^T > 0$, will be minimized, subject to the constraint $g(x)\widehat{\Lambda}u = \ell$, which must be satisfied at all times. The cost function is chosen to be $J = u^T Q u$ so that the control effort will be minimized, which consequently can be used to reduce the amount of trim drag during flight. it is assumed by this formulation of the problem that the control surfaces do not have position limits, and as a result, sufficient control power will always exist. By choosing the term ℓ in the constraint equation to be

$$\ell = \dot{x}_m - \hat{f}(x) + Ke - \nu \tag{24}$$

where $\hat{f}(x) : \mathbb{R}^n \mapsto \mathbb{R}^n$ is an estimate of the plant dynamics, $K \in \mathbb{R}^{n\times n}$ with $K = K^T > 0$ contains the gains on the errors, and $\nu \in \mathbb{R}^n$ is a pseudo-control signal, the constraint $g(x)\widehat{\Lambda}u = \ell$ will ensure that when the derived control law for this second case is substituted into the equation for $\dot{e}$, and the equation for the error dynamics is simplified, the first two terms of equation (21) will appear in the resulting equation for $\dot{e}$ as desired. For simplicity in the control law derivation, equation (24) will not be substituted into the constraint equation at the present time.

To derive the control law, first the constraint must be included in the cost function to form the augmented cost function

$$\bar{J} = u^T Q u + \lambda^T (g(x)\widehat{\Lambda}u - \ell) \tag{25}$$

where $\lambda \in \mathbb{R}^n$ is a Lagrange multiplier. The necessary conditions for minimizing $\bar{J}$ are given by

$$\frac{\partial \bar{J}}{\partial \lambda} = g(x)\widehat{\Lambda}u - \ell = 0 \tag{26}$$

$$\frac{\partial \bar{J}}{\partial u} = 2Qu + \widehat{\Lambda}^T g^T(x)\lambda = 0. \tag{27}$$

Rearranging terms in equation (27) results in

$$u = -\frac{1}{2}Q^{-1}\widehat{\Lambda}^T g^T(x)\lambda. \tag{28}$$

Substituting equation (28) into equation (26) and solving for λ produces the equation

$$\lambda = -2(g(x)\widehat{\Lambda}Q^{-1}\widehat{\Lambda}^T g^T(x))^{-1}\ell. \tag{29}$$

Finally, substituting equation (29) back into equation (28) results in the control law

$$u = Q^{-1}\widehat{\Lambda}^T g^T(x)(g(x)\widehat{\Lambda}Q^{-1}\widehat{\Lambda}^T g^T(x))^{-1}\ell. \tag{30}$$

In order for the control law given in equation (30) to be continuous, Q and $g(x)\widehat{\Lambda}Q^{-1}\widehat{\Lambda}^T g^T(x)$ must be invertible. The projection bounds that will be applied in the adaptive law for Λ must ensure that $\widehat{\Lambda}$ remains invertible. It should be noted that for the case where the number of controls equals the number of controlled variables, the control solution is unique, and the control law in equation (30) simplifies to

$$u = [g(x)]^{-1}[\dot{x}_m - \hat{f}(x) + Ke - \nu] \tag{31}$$

which is the control law that was chosen in Subsection A.

Continuing with the derivation of $\dot{e}$, let $\Delta = \hat{f}(x) - f(x)$. Substituting equation (30), equation (24), and into equation (23) produces the equation

$$\dot{e} = -Ke + \Delta + \nu + g(x)\widetilde{\Lambda}u. \tag{32}$$

Again, assume that Δ can be represented in the form $\Delta = W^T\beta(x;d)$, where $W \in \mathbb{R}^{p\times n}$ is a set of unknown weights, and $\beta \in \mathbb{R}^{p\times 1}$ is a set of known basis functions composed of the states x and a vector d

of bounded continuous exogenous inputs. Also, the representation for ν is chosen to be $\nu = -\widehat{W}^T\beta(x;d)$, where $\widehat{W} \in \mathbb{R}^{p\times n}$. Then, equation (32) can be written as

$$\dot{e} = -Ke - \widetilde{W}^T\beta(x;d) + g(x)\widetilde{\Lambda}u \tag{33}$$

where $\widetilde{W} = \widehat{W} - W$, which is the weight estimation error.

As in Subsection A, a Lyapunov analysis needs to be performed in order to determine the adaptive laws for $\widehat{\Lambda}$ and $\widehat{W}$ and to prove that the error between the states of the actual system and the reference model will converge. Given the candidate Lyapunov function

$$V = e^Te + tr(\widetilde{W}^T{\Gamma_W}^{-1}\widetilde{W}) + tr(\widetilde{\Lambda}{\Gamma_\Lambda}^{-1}\widetilde{\Lambda}) \tag{34}$$

where $\Gamma_W \in \mathbb{R}^{p\times p}$ with $\Gamma_W = {\Gamma_W}^T > 0$, and $\Gamma_\Lambda \in \mathbb{R}^{m\times m}$ with $\Gamma_\Lambda = {\Gamma_\Lambda}^T > 0$, the derivative of equation (34) along the system trajectories is taken, which results in the equation

$$\dot{V} = 2e^T\dot{e} + 2tr(\widetilde{W}^T{\Gamma_W}^{-1}\dot{\widehat{W}}^T) + 2tr(\widetilde{\Lambda}{\Gamma_\Lambda}^{-1}\dot{\widehat{\Lambda}}^T). \tag{35}$$

Substituting equation (33) into equation (35) produces

$$\dot{V} = -2e^TKe - 2e^T\widetilde{W}^T\beta(x;d) + 2e^Tg(x)\widetilde{\Lambda} + 2tr(\widetilde{W}^T{\Gamma_W}^{-1}\dot{\widehat{W}}^T) + 2tr(\widetilde{\Lambda}{\Gamma_\Lambda}^{-1}\dot{\widehat{\Lambda}}^T) \tag{36}$$

and by applying the trace identity that $a^Tb = tr(ba^T)$ to equation (36), the equation for $\dot{V}$ becomes

$$\dot{V} = -2e^TKe + 2tr(\widetilde{W}^T({\Gamma_W}^{-1}\dot{\widehat{W}}^T - \beta(x;d)e^T)) + 2tr(\widetilde{\Lambda}({\Gamma_\Lambda}^{-1}\dot{\widehat{\Lambda}}^T + ue^Tg(x))). \tag{37}$$

Let the equation for $\dot{\widehat{W}}$ in this case be the same as equation (13), and let $\dot{\widehat{\Lambda}}$ be

$$\dot{\widehat{\Lambda}} = \Gamma_\Lambda Proj(\widehat{\Lambda}, -ue^Tg(x)). \tag{38}$$

In this case, the final equation for $\dot{V}$ is is upper bounded by

$$\dot{V} \leq -2e^TKe \leq 0 \tag{39}$$

which implies that e is bounded. Since r is bounded by definition above and e is bounded, x_m is bounded, and thus x is bounded. Consequently, $g(x)$ and $\beta(x;d)$ are bounded as well. In order to use Barbalat's lemma to complete the proof, the second derivative of equation (34) along the system trajectories is taken, which gives the result

$$\ddot{V} = -4e^TK\dot{e}. \tag{40}$$

Substituting equation (33) into equation (40) produces

$$\ddot{V} = -4e^TK(-Ke - \widetilde{W}^T\beta(x;d) + g(x)\widetilde{\Lambda}u). \tag{41}$$

It should be noted that u is bounded because all of the the signals found in u, which is given by equations (30) and (24) are bounded. Thus, because e, $\widetilde{W}$, $\beta(x;d)$, $g(x)$, $\widetilde{\Lambda}$, and u are bounded as proved above, $\ddot{V}$ is bounded, and therefore $\dot{V}$ is uniformly continuous.

Finally, Barbalat's lemma can be applied. Because V is lower bounded, $\dot{V}$ is negative semi-definite, and $\dot{V}$ is uniformly continuous, by Barbalat's lemma $\dot{V} \to 0$ as $t \to \infty$, and thus $e \to 0$ as $t \to \infty$ as desired.

IV. P, Q, R Inversion Controller

The first designed controller was the inversion controller for the angular body rates of the GHV since these variables are linked directly to the control surface deflections, which control the vehicle. The reference inputs to the controller are the commanded angular body rates p_c, q_c and r_c, and the output states of the controller are the control surface deflections $\delta_{f,r}$, $\delta_{f,l}$, $\delta_{t,r}$, and $\delta_{t,l}$. Therefore, in this case, equation (17) represents the current system. In order for the adaptive dynamic inversion controller to be designed for the

angular body rates, $f(x)$ and $g(x)$ must be determined from the general nonlinear equations for $\dot{p}$, $\dot{q}$, and $\dot{r}$, which in vector-matrix form, are

$$[J]\frac{d\omega_{B,I}}{dt}\bigg|_B + \omega_{B,I} \times J\omega_{B,I} = M_{aero} + M_T \tag{42}$$

where

$$[J] = \begin{bmatrix} J_x & 0 & -J_{xz} \\ 0 & J_y & 0 \\ -J_{xz} & 0 & J_z \end{bmatrix} \tag{43}$$

and

$$\omega_{B,I} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{44}$$

Substituting these equations into equation (42) and simplifying produces the result

$$\begin{bmatrix} J_x & 0 & -J_{xz} \\ 0 & J_y & 0 \\ -J_{xz} & 0 & J_z \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} -J_{xz}pq + (J_z - J_y)qr \\ (J_x - J_z)pr + J_{xz}(p^2 - r^2) \\ J_{xz}qr + (J_y - J_x)pq \end{bmatrix} = M_{aero} + M_T. \tag{45}$$

Therefore, the nonlinear equations for the angular body accelerations can be written as

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} J_x & 0 & -J_{xz} \\ 0 & J_y & 0 \\ -J_{xz} & 0 & J_z \end{bmatrix}^{-1} \left(- \begin{bmatrix} -J_{xz}pq + (J_z - J_y)qr \\ (J_x - J_z)pr + J_{xz}(p^2 - r^2) \\ J_{xz}qr + (J_y - J_x)pq \end{bmatrix} + M_{aero} + M_T \right). \tag{46}$$

After having determined the nonlinear equations for the angular body accelerations, the next step is to write those equations in the form of equation (17). In order to accomplish this task, the terms related to the control surfaces, which will form $g(x)$, must be extracted from equation (46). The control surfaces terms are included in the aerodynamic moment terms M_A, which are modeled as

$$M_{aero} = \begin{bmatrix} L_A \\ M_A \\ N_A \end{bmatrix} = \begin{bmatrix} \bar{q}SbC_\ell \\ \bar{q}S\bar{c}C_m \\ \bar{q}SbC_n \end{bmatrix} \tag{47}$$

where

$$\begin{aligned} C_\ell &= C_{\ell,baseline} + \Delta C_{\ell,surfaces} + \frac{b}{2V_T}\left(C_{\ell_p}p\right) \\ C_m &= C_{m,baseline} + \Delta C_{m,surfaces} + \frac{\bar{c}}{2V_T}\left(C_{m_q}q + C_{m_{\dot{\alpha}}}\dot{\alpha}\right) \\ C_n &= C_{n,baseline} + \Delta C_{n,surfaces} + \frac{b}{2V_T}\left(C_{n_r}r\right) \end{aligned} \tag{48}$$

and

$$\begin{aligned} \Delta C_{i,surfaces} = {} & \Delta C_{i,\delta_{f,r}}(M,\alpha,\beta,\delta_{f,r}) + \Delta C_{i,\delta_{f,l}}(M,\alpha,\beta,\delta_{f,l}) \\ & + \Delta C_{i,\delta_{t,r}}(M,\alpha,\beta,\delta_{t,r}) + \Delta C_{i,\delta_{t,l}}(M,\alpha,\beta,\delta_{t,l}) \end{aligned} \tag{49}$$

for $i = \ell$, m, n.

As seen in equation (48), the moment coefficients are comprised of three parts. The baseline term is the moment coefficient for the base airframe, while the second and third terms adjust for the effects on the moment coefficients due to the control surfaces and damping, respectively. In equation (48), the first and third terms do not depend on the control surfaces; therefore, those two terms belong to the $f(x)$ term in equation (17). In order to determine $g(x)$, the second term in each equation in equation (48) must be examined to determine what portion of the term is control-dependent and thus belongs in $g(x)$. For this particular control design for the GHV, it is assumed that a linear approximation with respect to the control

surface deflection δ can be made for each of the terms in equation (49). The linear approximation can be expressed as

$$\begin{aligned}\Delta C_{i,\delta_s}(M,\alpha,\beta,\delta_s) &= C_{i,\delta_s}(M,\alpha,\beta,[\delta_s=0]) + \left.\frac{\partial C_{i,\delta_s}}{\partial \delta_s}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_s \\ &\text{for } i = N, Y, A, \ell, m, n \\ &\text{and } \delta_s = \delta_{f,r}, \delta_{f,l}, \delta_{t,r}, \delta_{t,l}.\end{aligned} \tag{50}$$

In this paper, it is assumed that all interactions between each control surface are negligible, which at high Mach numbers is approximately true. Deflections of the right and left control surfaces will generate summative forces and moments in the XZ-plane of symmetry of the aircraft, whereas in the other planes, the deflections will generate canceling forces and moments. In equation form, for both the flaps and the tail control surfaces, the relationships between right and left elevon deflections are expressed as

$$\begin{aligned} C_{N,\delta_{f,r}} &= C_{N,\delta_{f,l}} & -C_{Y,\delta_{f,r}} &= C_{Y,\delta_{f,l}} \\ C_{A,\delta_{f,r}} &= C_{A,\delta_{f,l}} & -C_{\ell,\delta_{f,r}} &= C_{\ell,\delta_{f,l}} \\ C_{m,\delta_{f,r}} &= C_{m,\delta_{f,l}} & -C_{n,\delta_{f,r}} &= C_{n,\delta_{f,l}} \end{aligned} \tag{51}$$

and the relationships between right and left rudder deflections are expressed similarly as

$$\begin{aligned} C_{N,\delta_{t,r}} &= C_{N,\delta_{t,l}} & -C_{Y,\delta_{t,r}} &= C_{Y,\delta_{t,l}} \\ C_{A,\delta_{t,r}} &= C_{A,\delta_{t,l}} & -C_{\ell,\delta_{t,r}} &= C_{\ell,\delta_{t,l}} \\ C_{m,\delta_{t,r}} &= C_{m,\delta_{t,l}} & -C_{n,\delta_{t,r}} &= C_{n,\delta_{t,l}}. \end{aligned} \tag{52}$$

Consequently, in equation (50), the term where $\delta_s = 0$ can be written for the combined effect of both the right and left control surfaces collectively as

$$C_{i,\delta_f}(M,\alpha,\beta,[\delta_{f,r}=0,\delta_{f,l}=0]) = \begin{cases} 2C_{i,\delta_{f,r}}(M,\alpha,\beta,[\delta_{f,r}=0]) & \text{for } i = N, A, m \\ 0 & \text{for } i = Y, \ell, n \end{cases} \tag{53}$$

$$C_{i,\delta_t}(M,\alpha,\beta,[\delta_{t,r}=0,\delta_{t,l}=0]) = \begin{cases} 2C_{i,\delta_{t,r}}(M,\alpha,\beta,[\delta_{t,r}=0]) & \text{for } i = N, A, m \\ 0 & \text{for } i = Y, \ell, n. \end{cases} \tag{54}$$

Given equations (50), (53), and (54), equation (49) can be rewritten for $i = \ell, m, n$ as

$$\begin{aligned}\Delta C_\ell = &\left.\frac{\partial C_{\ell,\delta_{f,r}}}{\partial \delta_{f,r}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{f,r} + \left.\frac{\partial C_{\ell,\delta_{f,l}}}{\partial \delta_{f,l}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{f,l} \\ &+ \left.\frac{\partial C_{\ell,\delta_{t,r}}}{\partial \delta_{t,r}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{t,r} + \left.\frac{\partial C_{\ell,\delta_{t,l}}}{\partial \delta_{t,l}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{t,l}\end{aligned} \tag{55}$$

$$\begin{aligned}\Delta C_m = &\, 2C_{m,\delta_{f,r}}(M,\alpha,\beta,[\delta_{f,r}=0]) + 2C_{m,\delta_{t,r}}(M,\alpha,\beta,[\delta_{t,r}=0]) \\ &+ \left.\frac{\partial C_{m,\delta_{f,r}}}{\partial \delta_{f,r}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{f,r} + \left.\frac{\partial C_{m,\delta_{f,l}}}{\partial \delta_{f,l}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{f,l} \\ &+ \left.\frac{\partial C_{m,\delta_{t,r}}}{\partial \delta_{t,r}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{t,r} + \left.\frac{\partial C_{m,\delta_{t,l}}}{\partial \delta_{t,l}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{t,l}\end{aligned} \tag{56}$$

$$\begin{aligned}\Delta C_n = &\left.\frac{\partial C_{n,\delta_{f,r}}}{\partial \delta_{f,r}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{f,r} + \left.\frac{\partial C_{n,\delta_{f,l}}}{\partial \delta_{f,l}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{f,l} \\ &+ \left.\frac{\partial C_{n,\delta_{t,r}}}{\partial \delta_{t,r}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{t,r} + \left.\frac{\partial C_{n,\delta_{t,l}}}{\partial \delta_{t,l}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{t,l}.\end{aligned} \tag{57}$$

Since the first two terms of equation (56) are for fixed values of δ_s, they constitute bias terms and therefore belong in the $f(x)$ portion of equation (17). As a result, only the terms represented by $\left.\frac{\partial C_{i,\delta_s}}{\partial \delta_s}\right|_{M,\alpha,\beta \text{ constant}}$ in equations (55), (56), and (57) belong in the $g(x)$ term in equation (17).

To complete the analysis of the terms in equation (46), the effect of the center of gravity shift must be accounted for in the nonlinear equations for the angular body accelerations. The shift of a set of moments from a given reference point to the center of gravity is given by the equation

$$M_{cg} = M_{aero} - r_{cg/aero} \times F_{aero} \tag{58}$$

and in this particular simulation, $r_{cg/aero}$ is defined to be $\begin{bmatrix} x_{cg} & 0 & 0 \end{bmatrix}^T$. In the simulation, F_{aero} is calculated similarly to M_{aero} in equation (47) above, which means that F_{aero} has the form

$$F_{aero} = \begin{bmatrix} X_A \\ Y_A \\ Z_A \end{bmatrix} = \begin{bmatrix} -\bar{q}SC_A \\ \bar{q}SC_Y \\ -\bar{q}SC_N \end{bmatrix}. \tag{59}$$

Therefore, given equation (59) and the definition of $r_{cg/aero}$, equation (58) can be written as

$$M_{cg} = M_{aero} - \begin{bmatrix} x_{cg} \\ 0 \\ 0 \end{bmatrix} \times \begin{bmatrix} -\bar{q}SC_A \\ \bar{q}SC_Y \\ -\bar{q}SC_N \end{bmatrix} \tag{60}$$

$$M_{cg} = M_{aero} - \begin{bmatrix} 0 \\ -\bar{q}SC_N x_{cg} \\ -\bar{q}SC_Y x_{cg} \end{bmatrix} \tag{61}$$

where M_{aero} is defined in equation (47). It should be noted that the terms C_N and C_Y in equation (61) can be written like the moment coefficients in equations (55), (56), and (57) as

$$\begin{aligned} \Delta C_N = {} & 2C_{N,\delta_{f,r}}(M,\alpha,\beta,[\delta_{f,r}=0]) + 2C_{N,\delta_{t,r}}(M,\alpha,\beta,[\delta_{t,r}=0]) \\ & + \left.\frac{\partial C_{N,\delta_{f,r}}}{\partial \delta_{f,r}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{f,r} + \left.\frac{\partial C_{N,\delta_{f,l}}}{\partial \delta_{f,l}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{f,l} \\ & + \left.\frac{\partial C_{N,\delta_{t,r}}}{\partial \delta_{t,r}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{t,r} + \left.\frac{\partial C_{N,\delta_{t,l}}}{\partial \delta_{t,l}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{t,l} \end{aligned} \tag{62}$$

$$\begin{aligned} \Delta C_Y = {} & \left.\frac{\partial C_{Y,\delta_{f,r}}}{\partial \delta_{f,r}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{f,r} + \left.\frac{\partial C_{Y,\delta_{f,l}}}{\partial \delta_{f,l}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{f,l} \\ & + \left.\frac{\partial C_{Y,\delta_{t,r}}}{\partial \delta_{t,r}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{t,r} + \left.\frac{\partial C_{Y,\delta_{t,l}}}{\partial \delta_{t,l}}\right|_{M,\alpha,\beta \text{ constant}} \Delta\delta_{t,l} \end{aligned} \tag{63}$$

Similarly to the moment coefficients as shown above, since the first two terms of equation (62) are for fixed values of δ_s, they constitute bias terms and therefore belong in the $f(x)$ portion of equation (17). As a result, only the terms represented by $\left.\frac{\partial C_{i,\delta_s}}{\partial \delta_s}\right|_{M,\alpha,\beta \text{ constant}}$ in equations (62) and (63) belong in the $g(x)$ term in equation (17).

Having examined all of the terms in the nonlinear equations for the angular body accelerations, equation (46) can be written in the final form of equation (17) as

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} J_x & 0 & -J_{xz} \\ 0 & J_y & 0 \\ -J_{xz} & 0 & J_z \end{bmatrix}^{-1} \left(- \begin{bmatrix} -J_{xz}pq + (J_z - J_y)qr \\ (J_x - J_z)pr + J_{xz}(p^2 - r^2) \\ J_{xz}qr + (J_y - J_x)pq \end{bmatrix} + M_T + \bar{q}SG + \bar{q}SH \begin{bmatrix} \delta_{f,r} \\ \delta_{f,l} \\ \delta_{t,r} \\ \delta_{t,l} \end{bmatrix} \right) \tag{64}$$

where

$$G = \begin{bmatrix} b\Big(C_{\ell,baseline} + \frac{b}{2V_T}(C_{\ell_p}p)\Big) \\ \bar{c}\Big(C_{m,baseline} + \frac{\bar{c}}{2V_T}(C_{m_q}q + C_{m_{\dot{\alpha}}}\dot{\alpha}) + 2C_{m,\delta_{f,r}}(\delta_{f,r}=0) + 2C_{m,\delta_{t,r}}(\delta_{t,r}=0)\Big) \\ -2C_{N,\delta_{f,r}}(\delta_{f,r}=0)x_{cg} - 2C_{N,\delta_{t,r}}(\delta_{t,r}=0)x_{cg} \\ b\Big(C_{r,baseline} + \frac{b}{2V_T}(C_{n_r}r)\Big) \end{bmatrix} \tag{65}$$

and

$$H = \begin{bmatrix} b\frac{\partial C_\ell}{\partial \delta_{f,r}} & b\frac{\partial C_\ell}{\partial \delta_{f,l}} & b\frac{\partial C_\ell}{\partial \delta_{t,r}} & b\frac{\partial C_\ell}{\partial \delta_{t,l}} \\ \Big(\bar{c}\frac{\partial C_m}{\partial \delta_{f,r}} - x_{cg}\frac{\partial C_N}{\partial \delta_{f,r}}\Big) & \Big(\bar{c}\frac{\partial C_m}{\partial \delta_{f,l}} - x_{cg}\frac{\partial C_N}{\partial \delta_{f,l}}\Big) & \Big(\bar{c}\frac{\partial C_m}{\partial \delta_{t,r}} - x_{cg}\frac{\partial C_N}{\partial \delta_{t,r}}\Big) & \Big(\bar{c}\frac{\partial C_m}{\partial \delta_{t,l}} - x_{cg}\frac{\partial C_N}{\partial \delta_{t,l}}\Big) \\ \Big(b\frac{\partial C_n}{\partial \delta_{f,r}} - x_{cg}\frac{\partial C_Y}{\partial \delta_{f,r}}\Big) & \Big(b\frac{\partial C_n}{\partial \delta_{f,l}} - x_{cg}\frac{\partial C_Y}{\partial \delta_{f,l}}\Big) & \Big(b\frac{\partial C_n}{\partial \delta_{t,r}} - x_{cg}\frac{\partial C_Y}{\partial \delta_{t,r}}\Big) & \Big(b\frac{\partial C_n}{\partial \delta_{t,l}} - x_{cg}\frac{\partial C_Y}{\partial \delta_{t,l}}\Big) \end{bmatrix}. \tag{66}$$

It should be noted that the partial derivatives in equation (64) are taken with respect to a constant value of M, α, and β from the current flight condition and that the control surface bias terms, where $\delta_s = 0$, are evaluated at a constant value of M, α and β from the current flight condition as well.

Given equation (64), which is now in the form of equation (17), the adaptive dynamic inversion controller can be constructed using equations (13), (24), (30), and (38).

V. α, β, μ Inversion Controller

As with the p, q, r inversion controller, equations for $\dot{\alpha}$, $\dot{\beta}$, and $\dot{\mu}$ must be determined in order for the adaptive dynamic inversion controller to be constructed. It should be noted that for this section, S_x will represent $\sin(x)$, C_x will represent $\cos(x)$, and T_x will represent $\tan(x)$, where x is an angle. The derivations for $\dot{\alpha}$ and $\dot{\beta}$ are based on the derivations for those terms on pages 110-112 in Reference [15]. The starting point of the derivations is the basic force equations in the stability axes under the flat Earth assumption, which are

$${}^b\dot{v}_{rel} = (1/m)F_{A,T} + g - \omega_{b/e} \times v_{rel}. \tag{67}$$

Taking the time derivative of the relative velocity in the wind axes instead of in the body axes and converting the right hand side of equation (67) to the wind axes produces the result

$$m\dot{V}_T = F_T C_{\alpha+\alpha_T} C_\beta - D - mgS_\gamma \tag{68}$$

$$m\dot{\beta}V_T = -F_T C_{\alpha+\alpha_T} S_\beta - C + mg(C_\alpha S_\beta S_\theta + C_\beta S_\phi C_\theta - S_\alpha S_\beta C_\phi C_\theta) - mV_T(pS_\alpha - rC_\alpha) \tag{69}$$

$$m\dot{\alpha}V_T C_\beta = -F_T S_{\alpha+\alpha_T} - L + mg(S_\alpha S_\theta + C_\alpha C_\phi C_\theta) + mV_T(-pS_\beta C_\alpha + qC_\beta - rS_\beta S_\alpha) \tag{70}$$

where D, L, and C represent drag, lift, and cross-wind force, respectively, in the wind axes.

In order to simplify equations (69) and (70) and to express them in terms of μ, which is one of the commanded states, the gravity terms in those equations are transformed using relationships given by the following direction cosine matrices from Chapter 4 of Reference [12] as

$$T_{W,H}(\mu,\gamma,\chi) = T^T_{B,W}(0,-\alpha,\beta)T_{B,H}(\phi,\theta,\psi) \tag{71}$$

where W represents the wind axes, B represents the body axes, and H represents the local horizon axes. Each direction cosine matrix has the general form

$$T_{2,1}(\theta_x,\theta_y,\theta_z) = \begin{bmatrix} C_{\theta_y}C_{\theta_z} & C_{\theta_y}S_{\theta_z} & -S_{\theta_y} \\ S_{\theta_x}S_{\theta_y}C_{\theta_z} - C_{\theta_x}S_{\theta_z} & S_{\theta_x}S_{\theta_y}S_{\theta_z} + C_{\theta_x}C_{\theta_z} & S_{\theta_x}C_{\theta_y} \\ C_{\theta_x}S_{\theta_y}C_{\theta_z} + S_{\theta_x}S_{\theta_z} & C_{\theta_x}S_{\theta_y}S_{\theta_z} - S_{\theta_x}C_{\theta_z} & C_{\theta_x}C_{\theta_y} \end{bmatrix}. \tag{72}$$

as shown on page 9 of Reference [16]. By examining the elements of the matrices in equation (71), the following relationships involving μ and γ were determined to be

$$T_{W,H}(2,3) = S_\mu C_\gamma = C_\alpha S_\beta S_\theta + C_\beta S_\phi C_\theta - S_\alpha S_\beta C_\phi C_\theta \tag{73}$$

$$T_{W,H}(3,3) = C_\mu C_\gamma = S_\alpha S_\theta + C_\alpha C_\phi C_\theta \tag{74}$$

which can be substituted into equations (69) and (70) in the gravity terms.

Additionally, the thrust force F_T terms are converted into the wind frame and expressed in terms of the vector $\begin{bmatrix} F_{T_x} & F_{T_y} & F_{T_z} \end{bmatrix}^T$, which is given in the body frame. The transformation of the F_T terms results in

$$\begin{aligned} \begin{bmatrix} F_T C_{\alpha+\alpha_T} C_\beta \\ -F_T C_{\alpha+\alpha_T} S_\beta \\ -F_T S_{\alpha+\alpha_T} \end{bmatrix} &= \begin{bmatrix} C_\alpha C_\beta & S_\beta & S_\alpha C_\beta \\ -C_\alpha S_\beta & C_\beta & -S_\alpha S_\beta \\ -S_\alpha & 0 & C_\alpha \end{bmatrix} \begin{bmatrix} F_{T_x} \\ F_{T_y} \\ F_{T_z} \end{bmatrix} \\ &= \begin{bmatrix} F_{T_x} C_\alpha C_\beta + F_{T_y} S_\beta + F_{T_z} S_\alpha C_\beta \\ -F_{T_x} C_\alpha S_\beta + F_{T_y} C_\beta - F_{T_z} S_\alpha S_\beta \\ -F_{T_x} S_\alpha + F_{T_z} C_\alpha \end{bmatrix}. \end{aligned} \tag{75}$$

Finally, the forces D, C, and L must be written in terms of the corresponding forces in the stability axes, which can be calculated directly from information in the model, as

$$\begin{aligned} \begin{bmatrix} D \\ C \\ L \end{bmatrix} &= \begin{bmatrix} C_\beta & S_\beta & 0 \\ -S_\beta & C_\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} D_s \\ Y_s \\ L_s \end{bmatrix} \\ &= \begin{bmatrix} D_s C_\beta + Y_s S_\beta \\ -D_s S_\beta + Y_s C_\beta \\ L_s \end{bmatrix}. \end{aligned} \tag{76}$$

It is assumed that the D_s terms are absorbed into the thrust terms in equation (75).

Substituting equations (73), (74), (75), and (76) into equations (69) and (70) gives the final equations for $\dot{\beta}$ and $\dot{\alpha}$ to be

$$\dot{\beta} = \frac{1}{mV_T}\left((Y_s + F_{T_y})C_\beta + mgS_\mu C_\gamma - F_{T_x} C_\alpha S_\beta - F_{T_z} S_\alpha S_\beta\right) + (pS_\alpha - rC_\alpha) \tag{77}$$

$$\dot{\alpha} = \frac{1}{mV_T C_\beta}\left(-L_s + mgC_\mu C_\gamma - F_{T_x} S_\alpha + F_{T_z} C_\alpha\right) + (-pC_\alpha T_\beta + q - rS_\alpha T_\beta). \tag{78}$$

Now, the equation for $\dot{\mu}$ can be derived since the derivation involves the results given in equations (77) and (78). Starting from equation (57) on page 56 of Reference [12], where, for this document $\beta = -\sigma$ in Reference [12], the relationship between the angular body accelerations and the local horizon angular accelerations are expressed as

$$\begin{bmatrix} p - \dot{\beta} S_\alpha \\ q - \dot{\alpha} \\ r + \dot{\beta} C_\alpha \end{bmatrix} = \begin{bmatrix} C_\alpha C_\beta & -C_\alpha S_\beta & -S_\alpha \\ S_\beta & C_\beta & 0 \\ S_\alpha C_\beta & -S_\alpha S_\beta & C_\alpha \end{bmatrix} \begin{bmatrix} 1 & 0 & -S_\gamma \\ 0 & C_\mu & S_\mu C_\gamma \\ 0 & -S_\mu & C_\mu C_\gamma \end{bmatrix} \begin{bmatrix} \dot{\mu} \\ \dot{\gamma} \\ \dot{\chi} \end{bmatrix}. \tag{79}$$

Taking the inverse of equation (79), the equation for $\dot{\mu}$ is determined to be

$$\begin{aligned} \dot{\mu} = &(p - \dot{\beta} S_\alpha)(C_\alpha C_\beta - T_\gamma C_\alpha S_\beta S_\mu - T_\gamma S_\alpha C_\mu) + (q - \dot{\alpha})(S_\beta + T_\gamma C_\beta S_\mu) \\ &+ (r + \dot{\beta} C_\alpha)(S_\alpha C_\beta + T_\gamma C_\alpha C_\mu - T_\gamma S_\alpha S_\beta S_\mu). \end{aligned} \tag{80}$$

Substituting equations (77) and (78) into equation (80) and simplifying gives the final equation for $\dot{\mu}$, which is

$$\begin{aligned} \dot{\mu} = \frac{1}{mV_T}\Big(&L_s(T_\beta + T_\gamma S_\mu) + (Y_s + F_{T_y}) T_\gamma C_\mu C_\beta - mgC_\gamma C_\mu T_\beta + (F_{T_x} S_\alpha - F_{T_z} C_\alpha)(T_\gamma S_\mu + T_\beta) \\ &- (F_{T_x} C_\alpha + F_{T_z} S_\alpha) T_\gamma C_\mu S_\beta\Big) + pC_\alpha \sec(\beta) + rS_\alpha \sec(\beta). \end{aligned} \tag{81}$$

Finally, equations (77), (78), and (81) are combined together in vector-matrix equation form as

$$\begin{bmatrix}\dot{\beta}\\ \dot{\alpha}\\ \dot{\mu}\end{bmatrix} = \begin{bmatrix} \frac{1}{mV_T}\left((Y_s+F_{T_y})C_\beta + mgS_\mu C_\gamma - F_{T_x}C_\alpha S_\beta - F_{T_z}S_\alpha S_\beta\right) \\ \frac{1}{mV_T C_\beta}\left(-L_s + mgC_\mu C_\gamma - F_{T_x}S_\alpha + F_{T_z}C_\alpha\right) \\ \frac{1}{mV_T}\Big(L_s(T_\beta + T_\gamma S_\mu) + (Y_s+F_{T_y})T_\gamma C_\mu C_\beta - mgC_\gamma C_\mu T_\beta \\ \quad + (F_{T_x}S_\alpha - F_{T_z}C_\alpha)(T_\gamma S_\mu + T_\beta) - (F_{T_x}C_\alpha + F_{T_z}S_\alpha)T_\gamma C_\mu S_\beta\Big) \end{bmatrix} + \begin{bmatrix} S_\alpha & 0 & -C_\alpha \\ -T_\beta C_\alpha & 1 & -T_\beta S_\alpha \\ \sec(\beta)C_\alpha & 0 & \sec(\beta)S_\alpha \end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix} \tag{82}$$

where p, q, and r are the desired angular body rates. It should be noted that it is assumed that the forces due to control surface deflections are negligible, and therefore, the force terms in equation (82) are approximated from look-up tables for the force and moment coefficients at points where the control surface deflections are equal to 0. Also, it is assumed that for the desired angular body rates that the inner loop p, q, r controller is perfect, which means that the desired angular rates equal the commanded angular rates.

Given equation (82), which is now in the form of equation (1), the adaptive dynamic inversion controller can be constructed using equations (5) and (13).

VI. Robustness Analysis

Based on the control and adaptive laws derived in the previous sections, a simulation of the entire GHV system with the nonlinear adaptive nonlinear dynamic inversion control architecture was created in Simulink. In order to make the simulation more realistic, second-order actuator dynamics with damping ratio $\zeta = 0.7$ and natural frequency $\omega_n = 25$ Hz are included in the current simulation, and position and rate limits are placed on the control surfaces of 30 deg and 100 deg/s, respectively. Additionally, a time delay of 0.03 s is included in the simulation; however, it should be noted that the simulation can tolerate time delays of up to 0.04 s without the responses becoming significantly oscillatory. Commands to α, β, and μ are given as ramp signals from 0 degrees to a commanded angle in fixed time. For the α, β, μ inversion controller, the basis function $\beta(x;d)$ is chosen to be $\beta(x;d) = \begin{bmatrix} c & \alpha & \beta & \mu & M \end{bmatrix}^T$, where c is a constant bias term. For the p, q, r inversion controller, the basis function $\beta(x;d)$ is chosen to be $\beta(x;d) = \begin{bmatrix} c & p & q & r & \alpha & \beta & M \end{bmatrix}^T$, where c is a constant bias term.

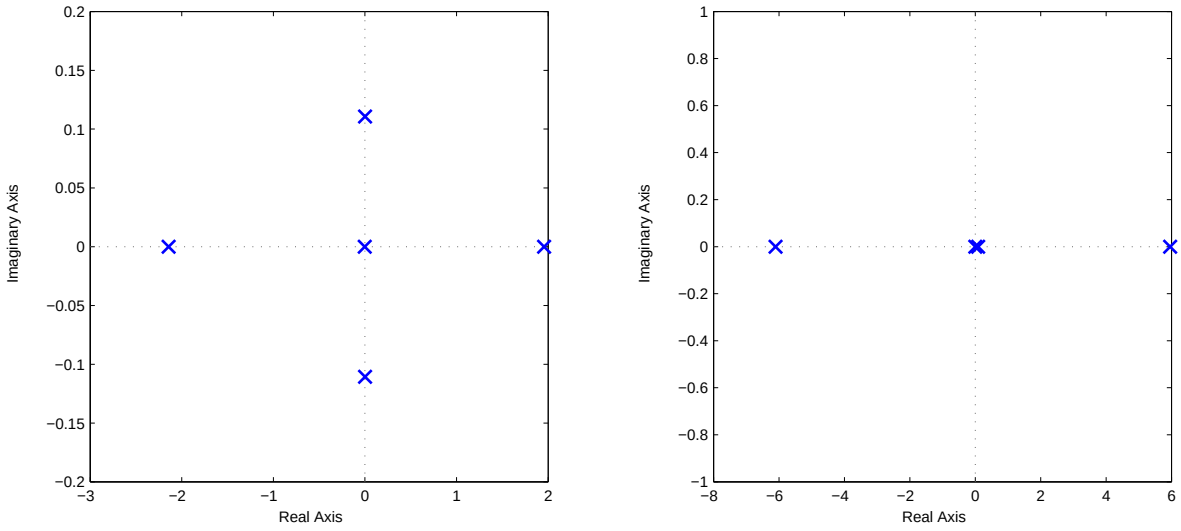
The total velocity of the vehicle is controlled using a PID controller, which is not depicted in Figure 2. The input to the controller is the commanded total velocity of the GHV, and the output of the controller is the equivalence ratio. The equivalence ratio is the fifth control, and along with the four control surfaces, completes the the control complement for the vehicle. Additionally, a saturation limiter has been added after the velocity PID controller to constrain the equivalence ratio to be between 0 and 1.

In the derivation of the adaptive dynamic controllers in Section III, a reference model was described. The difference between this reference model and the actual system dynamics constitutes the tracking error of the system. In order to determine the reference states of the system, the reference signal r must be defined. For the α, β, μ inversion controller, the reference signal consists of the commanded values of α, β, and μ. For the p, q, r inversion controller, the reference signal consists of the commanded angular body rates from the α, β, μ inversion controller. Both of the reference models have the general form

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix} = \begin{bmatrix}\xi_1 & 0 & 0\\ 0 & \xi_2 & 0\\ 0 & 0 & \xi_3\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix} + \begin{bmatrix}\eta_1 & 0 & 0\\ 0 & \eta_2 & 0\\ 0 & 0 & \eta_3\end{bmatrix}\begin{bmatrix}r_1\\ r_2\\ r_3\end{bmatrix} \tag{83}$$

where $\xi_1, \xi_2, \xi_3, \eta_1, \eta_2,$ and η_3 are scalars that define the desired time constants of each control channel.

The open-loop poles of the linearized dynamics at the flight condition of Mach 6 at 80,000 ft for both the longitudinal and lateral-directional states are shown in Figure 3. It should be noted from the eigenvalues



(a) Longitudinal.

(b) Lateral/Directional.

Figure 3. Open-loop poles for the linearized longitudinal and lateral/directional dynamics.

Table 1. Eigenvalues for the linearized longitudinal dynamics.

Eigenvalue	Damping Ratio	Natural Frequency (rad/s)
-2.14	1.00	2.14
-2.79×10^{-3}	1.00	2.79×10^{-3}
$1.25 \times 10^{-3} \pm 0.111j$	-0.0113	0.111
1.96	-1.00	1.96

Table 2. Eigenvalues for the linearized lateral/directional dynamics.

Eigenvalue	Damping Ratio	Natural Frequency (rad/s)
-6.10	1.00	6.10
2.22×10^{-16}	-1.00	2.22×10^{-16}
0.088	-1.00	0.088
5.96	-1.00	5.96

listed in Tables 1 and 2 that both the longitudinal and lateral-directional states have several eigenvalues in the right-half plane, which indicates that the GHV is an unstable vehicle. A nonlinear adaptive dynamic inversion controller will be able to suppress the unstable dynamics and replace them with desired dynamics for the aircraft.

Figures 4 and 5 show representative simulation results with the nonlinear adaptive dynamic inversion control architecture for the commands $\alpha = \pm 2$ deg, $\beta = 0$ deg, and $\mu = 70$ deg. The responses are well-behaved, and the control architecture is able to achieve the desired tracking performance without excessive control effort.

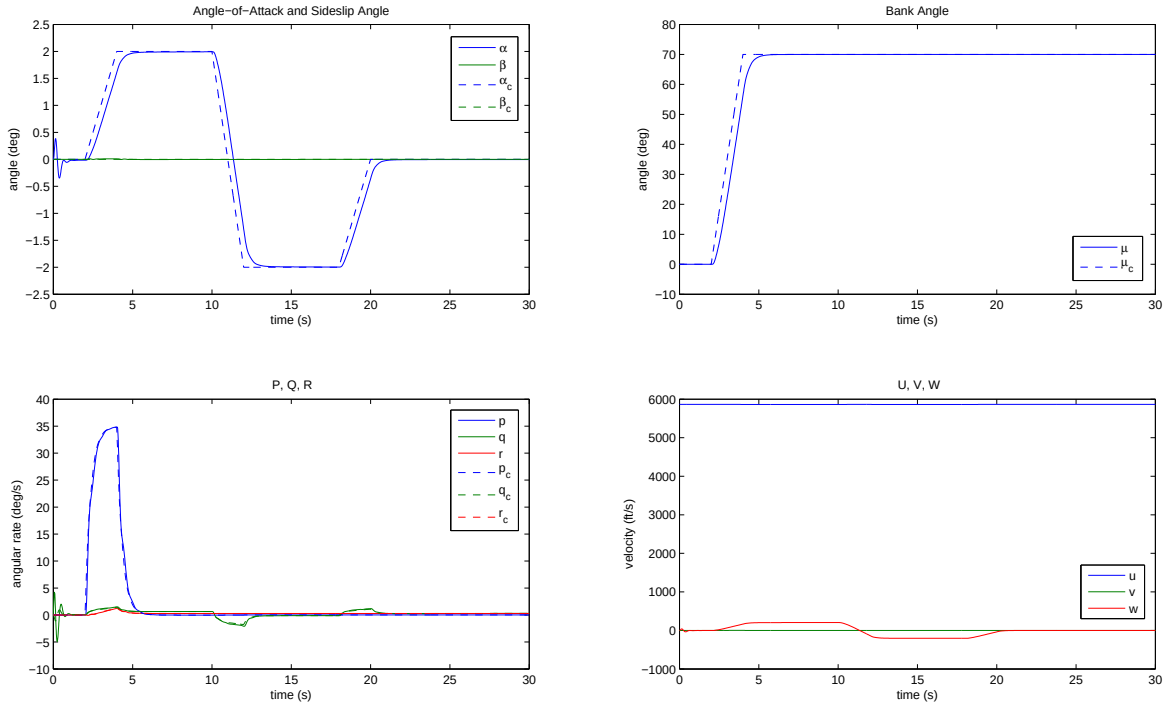


Figure 4. State responses for the commands $\alpha = \pm 2$ deg, $\beta = 0$ deg, and $\mu = 70$ deg.

A robustness analysis was performed via simulation on the designed adaptive nonlinear dynamic inversion control architecture from the previous section in order to determine the maximum tolerable uncertainties in selected parameters for the control architecture. Uncertainties in the plant examined in the analysis include the additive uncertainties ΔC_{m_α}, ΔC_{n_β}, and ΔC_m and multiplicative gains D on the control surface deflections, given in terms of equations as

$$C_m = C_{m_{baseline}} + \Delta C_{m_\alpha} \alpha \tag{84}$$

$$C_n = C_{n_{baseline}} + \Delta C_{n_\beta} \beta \tag{85}$$

$$C_m = C_{m_{baseline}} + \Delta C_m \tag{86}$$

$$C_\delta = D C_{\delta_o}. \tag{87}$$

The criteria for determining the bounds on the uncertainties is that the states must not demonstrate oscillatory behavior.

Tables 3 and 4 provide the maximum and minimum values of the additive uncertainties ΔC_{m_α} and ΔC_{n_β} for various α, β, and μ commands. It should be noted that an examination of the maximum and minimum baseline values of C_{m_α} and C_{n_β} show that these values are on the order of 10^{-4}, which is typical of a high-speed vehicle. The maximum and minimum values for ΔC_{m_α} and ΔC_{n_β} in Tables 3 and 4 are on the order of $10^{-4} - 10^{-3}$, and therefore, the control architecture is able to withstand considerable uncertainties

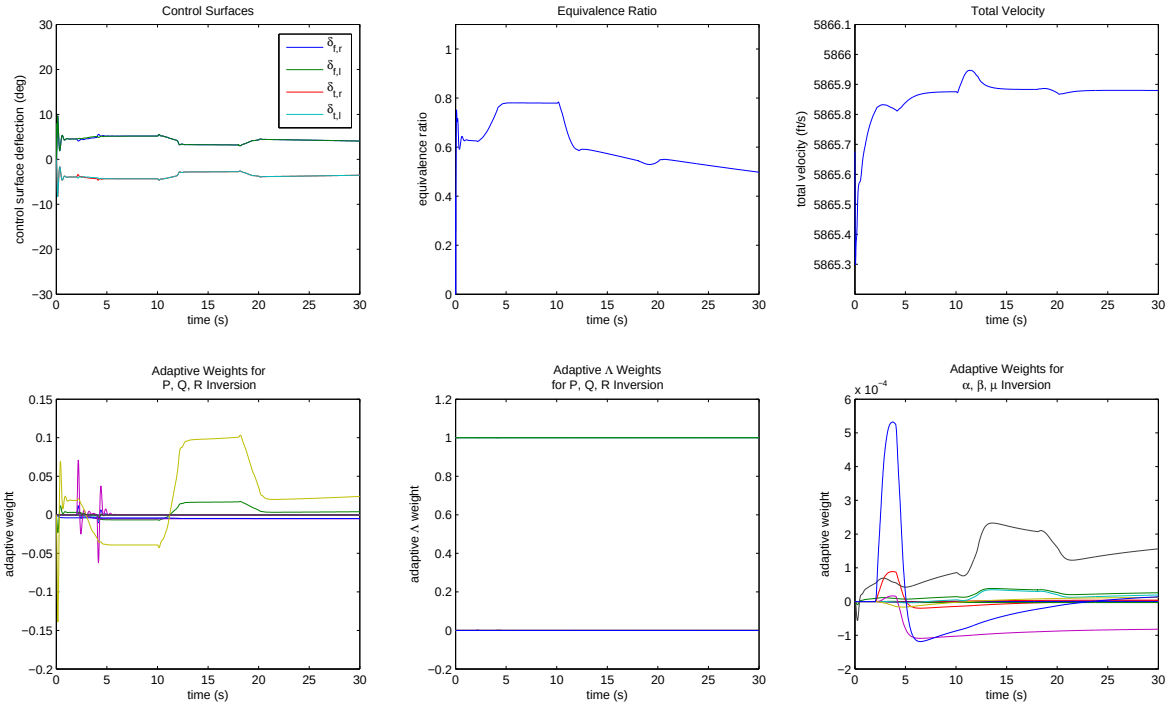


Figure 5. Control and adaptive weight responses for the commands $\alpha = \pm 2$ deg, $\beta = 0$ deg, and $\mu = 70$ deg.

in C_{m_α} and C_{n_β} and maintain stable tracking flight. Similar results for the additive uncertainty ΔC_m can be found in Table 5.

Table 6 contains the minimum allowable multiplicative gains D on the control surface deflections for various α, β, and μ commands. These gains represent a loss of control effectiveness for one or more of the control surfaces on the GHV. For all cases, the vehicle is able to tolerate low values of control effectiveness, which shows that the control architecture is robust to loss of control effectiveness.

Table 3. Additive uncertainty ΔC_{m_α} over a 30 s period with 0.03 s time delay.

α (deg)	β (deg)	μ (deg)	max ΔC_{m_α} (deg^{-1})	min ΔC_{m_α} (deg^{-1})
5	0	0	0.0005	-0.0013
5	1	20	0.0003	-0.0011

It should be noted that following this preliminary analysis of the nonlinear adaptive dynamic inversion control architecture, pseudo-control hedging ([17],[18]) was added to the simulation in order to protect the nonlinear adaptive dynamic inversion control architecture during periods of control surface saturation. However, for most of the cases simulated for the GHV for this paper, control surface saturation was not encountered.

VII. Reference Trajectory Generation

While the tracking of α, β, and μ was achieved as demonstrated in Section VI, it was desired that the GHV have the ability to track a realistic trajectory instead of selected commands. In order to control flight path angle γ as opposed to α, a nonzero setpoint (NZSP) controller ([19], [20]) was designed to generate

Table 4. Additive uncertainty ΔC_{n_β} over a 30 s period with 0.03 s time delay.

α (deg)	β (deg)	μ (deg)	max ΔC_{n_β} (deg^{-1})	min ΔC_{n_β} (deg^{-1})
0	1	0	0.007	-0.003
5	0	20	0.01	-0.004
5	1	20	0.006	-0.003

Table 5. Additive uncertainty ΔC_m over a 30 s period with 0.03 s time delay.

α (deg)	β (deg)	μ (deg)	max ΔC_m	min ΔC_m
5	0	0	0.0005	-0.003
5	1	20	0.0005	-0.002

Table 6. Multiplicative gains D on control surface deflection terms over a 30 s period with 0.03 s time delay.

α (deg)	β (deg)	μ (deg)	$D_{\delta_{f,r}}$	$D_{\delta_{f,l}}$	$D_{\delta_{t,r}}$	$D_{\delta_{t,l}}$
5	0	0	1	0.14	1	1
5	0	0	1	1	1	0.01
5	0	0	0.15	0.15	1	1
5	0	0	1	1	0.15	0.15
5	0	20	1	0.31	1	1
5	0	20	1	1	1	0.01
5	0	20	0.21	0.21	1	1
5	0	20	1	1	0.30	0.30
5	1	20	1	0.42	1	1
5	1	20	1	1	1	0.05
5	1	20	0.38	0.38	1	1
5	1	20	1	1	0.38	0.38

trajectories for the GHV to follow. The NZSP controller requires a linear model, so the nonlinear GHV plant model was linearized about a flight condition specified by the Mach number and altitude. Assuming that the vehicle remains wings-level during its flight of the trajectory, only the longitudinal dynamics model will be required for the trajectory generation. For the NZSP controller, the longitudinal states are $\begin{bmatrix} u & \theta & q & \alpha \end{bmatrix}^T$, and the controls are $\begin{bmatrix} \delta_T & \delta_e \end{bmatrix}^T$ where δ_T represents the equivalence ratio control, and δ_e represents the elevator control, expressed in terms of the GHV controls as $\delta_e = (\delta_{f,r} + \delta_{f,l})/2$. The outputs y_m to be commanded by the NZSP controller are velocity u and flight path angle γ, which can be expressed in matrix-vector form as

$$y_m = \begin{bmatrix} u \\ \gamma \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ \theta \\ q \\ \alpha \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_T \\ \delta_e \end{bmatrix}. \tag{88}$$

By fitting a polynomial to the trajectory generated for γ, and finding the derivative of that polynomial, the reference model for γ is completely defined for the GHV simulation. In order to implement the γ, β, μ inversion controller, the dynamic equation for $\dot{\gamma}$ must be derived and written in the form of equation (1) as

$$\dot{\gamma} = f(s) + g(s) \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{89}$$

where s represents the states in the GHV simulation. The equation for $\dot{\gamma}$ is derived using the same process that was applied to find $\dot{\mu}$ in Section V. Starting from equation (79), and taking its inverse, the equation for $\dot{\gamma}$ is determined to be

$$\dot{\gamma} = D(p - \dot{\beta}S_\alpha) + E(q - \dot{\alpha}) + F(r + \dot{\beta}C_\alpha) \tag{90}$$

where

$$D = C_\alpha S_\beta C_\mu - S_\alpha S_\mu \tag{91}$$

$$E = C_\beta C_\mu \tag{92}$$

$$F = \frac{{C_\alpha}^2 S_\beta {C_\mu}^2 - C_\alpha S_\alpha \left({S_\beta}^2 + 1\right) C_\mu S_\mu + \left({S_\alpha}^2 {S_\mu}^2 - 1\right) S_\beta}{C_\alpha S_\beta S_\mu + S_\alpha C_\mu}. \tag{93}$$

Consider the equations for $\dot{\beta}$ and $\dot{\alpha}$ in equations (77) and (78), respectively to have the following form

$$\dot{\beta} = f_\beta + (pS_\alpha - rC_\alpha) \tag{94}$$

$$\dot{\alpha} = f_\alpha + (-pC_\alpha T_\beta + q - rS_\alpha T_\beta) \tag{95}$$

where f_β and f_α represent the terms in $\dot{\beta}$ and $\dot{\alpha}$, respectively, that do not depend explicitly on the angular rates p, q, and r. Substituting equations (94) and (95) into equation (90) and simplifying the expression gives the resulting equation for $\dot{\gamma}$ that

$$\begin{aligned} \dot{\gamma} = & -Df_\beta S_\alpha - Ef_\alpha + Ff_\beta C_\alpha \\ & + p\left(D - D{S_\alpha}^2 + EC_\alpha T_\beta + FC_\alpha S_\alpha\right) \\ & + q\,(0) \\ & + r\left(DC_\alpha S_\alpha + ES_\alpha T_\beta + F - F{C_\alpha}^2\right). \end{aligned} \tag{96}$$

Note that there is no dependence on q in the equation for $\dot{\gamma}$. Consequently, when the dynamic equations for β, γ, and μ are expressed in the form

$$\begin{bmatrix} \dot{\beta} \\ \dot{\gamma} \\ \dot{\mu} \end{bmatrix} = f(s) + g(s) \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{97}$$

where s represents the states of the GHV, the resulting expression for g(s) is

$$g(s) = \begin{bmatrix} S_\alpha & 0 & -C_\alpha \\ \left(D - D{S_\alpha}^2 + EC_\alpha T_\beta + FC_\alpha S_\alpha\right) & 0 & \left(DC_\alpha S_\alpha + ES_\alpha T_\beta + F - F{C_\alpha}^2\right) \\ \sec(\beta)C_\alpha & 0 & \sec(\beta)S_\alpha \end{bmatrix}. \tag{98}$$

As can be seen in equation (98), $g(s)$ is not invertible, which causes a problem with the computation of p, q, and r in the new γ, β, μ inversion block. If $g(s)$ cannot be inverted, then the commands for p, q, and r cannot be determined for the inversion controller. Therefore, substituting the equation for $\dot{\gamma}$ for the equation for $\dot{\alpha}$ in the α, β, μ inversion controller in order to track a trajectory for γ is not possible for the GHV simulation, and another method of including the γ trajectory in the GHV simulation had to be determined.

In order to allow for the GHV simulation to track a flight path angle trajectory, a method from Reference [21] was applied in which the equation for $\ddot{h}$, where h represents the altitude of the aircraft, is written in the form

$$\ddot{h} = f_h(s) + g_h(s) \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{99}$$

Given the equation for $\dot{h}$

$$\dot{h} = V\left(C_\beta C_\alpha S_\theta - S_\beta S_\phi C_\theta - C_\beta S_\alpha C_\phi C_\theta\right), \tag{100}$$

where V is the total velocity of the vehicle, the equation for $\ddot{h}$ is determined to be

$$\ddot{h} = \left[b_0\dot{V} + b_1\dot{\beta} + b_2\dot{\alpha}\right] + \begin{bmatrix} a_0 & a_1 & a_2 \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{101}$$

where

$$\begin{aligned} a_0 &= b_4 \\ a_1 &= b_3 C_\phi + b_4 S_\phi T_\theta \\ a_2 &= b_4 C_\phi T_\theta - b_3 S_\phi \end{aligned}$$

and

$$\begin{aligned} b_0 &= C_\beta C_\alpha S_\theta - S_\beta S_\phi C_\theta - C_\beta S_\alpha C_\phi C_\theta \\ b_1 &= V\left(-S_\beta C_\alpha S_\theta - C_\beta S_\phi C_\theta + S_\beta S_\alpha C_\phi C_\theta\right) \\ b_2 &= V\left(-C_\beta S_\alpha S_\theta - C_\beta C_\alpha C_\phi C_\theta\right) \\ b_3 &= V\left(C_\beta C_\alpha C_\theta + S_\beta S_\phi S_\theta + C_\beta S_\alpha C_\phi S_\theta\right) \\ b_4 &= V\left(-S_\beta C_\phi C_\theta + C_\beta S_\alpha S_\phi C_\theta\right). \end{aligned}$$

Because $\ddot{h}$ has a nonzero coefficient for q, which means that the term $g(s)$ in equation (97) is invertible, the equation for $\ddot{h}$ can replace the equation for $\dot{\alpha}$ in equation (82) for the α, β, μ inversion controller. The original reference trajectory that is generated for γ using the NZSP controller can be converted to $\dot{h}$ using the relation from aircraft kinematics that $\dot{h} = VS_\gamma$. Once a polynomial is fitted to the new trajectory for $\dot{h}$, and the derivative of that polynomial is determined, the reference model is defined for $\dot{h}$. The $\dot{h}$, β, μ inversion controller replaces the original α, β, μ inversion controller in the GHV simulation, and now desired trajectories for γ can be tracked.

VIII. Simulation Results

The GHV simulation with the $\dot{h}$, β, μ inversion controller was implemented in Simulink. A trajectory for γ was generated at the flight condition of Mach 6 at 80,000 ft. Figures 6, 7, and 8 present representative simulation results for the flight path angle trajectory shown in Figure 6. It should be noted that in this particular simulation, a command of $\beta = -4^o$ also is given to the GHV. The nonlinear adaptive dynamic inversion control architecture achieves the desired tracking performance for the generated flight path angle trajectory and sideslip angle.

A simplified inlet unstart model was added to test the ability of the nonlinear adaptive dynamic inversion control architecture to handle tracking a trajectory during an inlet unstart. For the simulation, an inlet unstart is triggered at a specified time, and the loss of thrust and changes to aerodynamic parameters following the unstart are modeled as instantaneous changes. The coefficient of the axial force (C_A) is increased slightly, and the coefficient of the normal force (C_N) is decreased slightly. Additive variations in C_{m_α} and C_{n_β} are included in the plant through the equations

$$C_m = C_{m_{baseline}} + \Delta C_{m_\alpha}\alpha \tag{102}$$

$$C_n = C_{n_{baseline}} + \Delta C_{n_\beta}\beta. \tag{103}$$

Through a robustness analysis, it was determined that the maximum destabilizing additive variations in C_{m_α} and C_{n_β} that the nonlinear adaptive dynamic inversion control architecture could tolerate were $\Delta C_{m_\alpha} = 0.001$ deg^{-1} and $\Delta C_{n_\beta} = -0.001$ deg^{-1}.

Figures 9, 10, and 11 show the results for the GHV simulation during flight path angle tracking with an inlet unstart that occurs at time t = 10 seconds. It should be noted in Figure 11 that while the equivalence ratio is commanded to its maximum value following the inlet unstart, thrust is not being generated by the vehicle after time t = 10 seconds. While tracking performance is somewhat degraded, the aircraft is still able to nominally track the specified flight path angle trajectory.

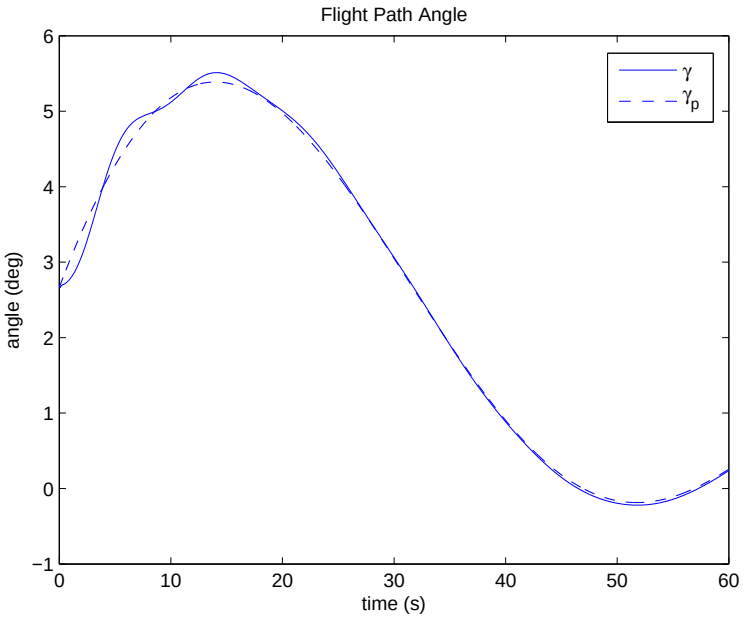


Figure 6. Flight path angle response compared with the generated flight path angle trajectory. The subscript p represents the flight path angle computed from the polynomial fit of $\dot{h}$.

IX. Conclusions

Because the dynamic equations for the GHV are inherently nonlinear and the aerodynamic and control derivatives for the aircraft have significant uncertainty associated with them, a nonlinear adaptive dynamic inversion control architecture was selected as the preferred control architecture to stabilize and control the aircraft. Based on the simulation results and the robustness analysis, it can be seen that the objective of designing a control architecture that is robust in order to achieve desired tracking performance was achieved for the GHV. The control architecture is robust to decreases in control surface effectiveness, changes in system parameters, and time delays of 0.04 s or less. The responses for tracking generated flight-path angle trajectories are well behaved, and the necessary control effort for tracking is not excessive. Additionally, the control architecture is able to tolerate an inlet unstart and maintain nominal tracking of a specified flight

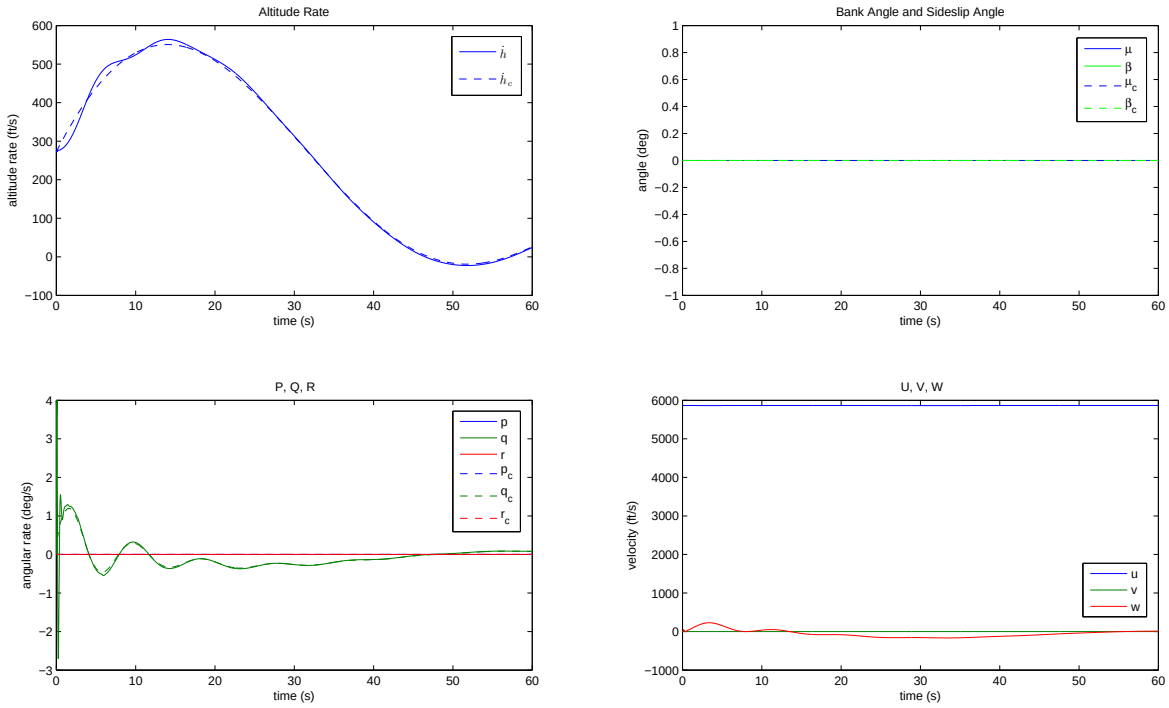


Figure 7. State responses for the generated flight path angle trajectory.

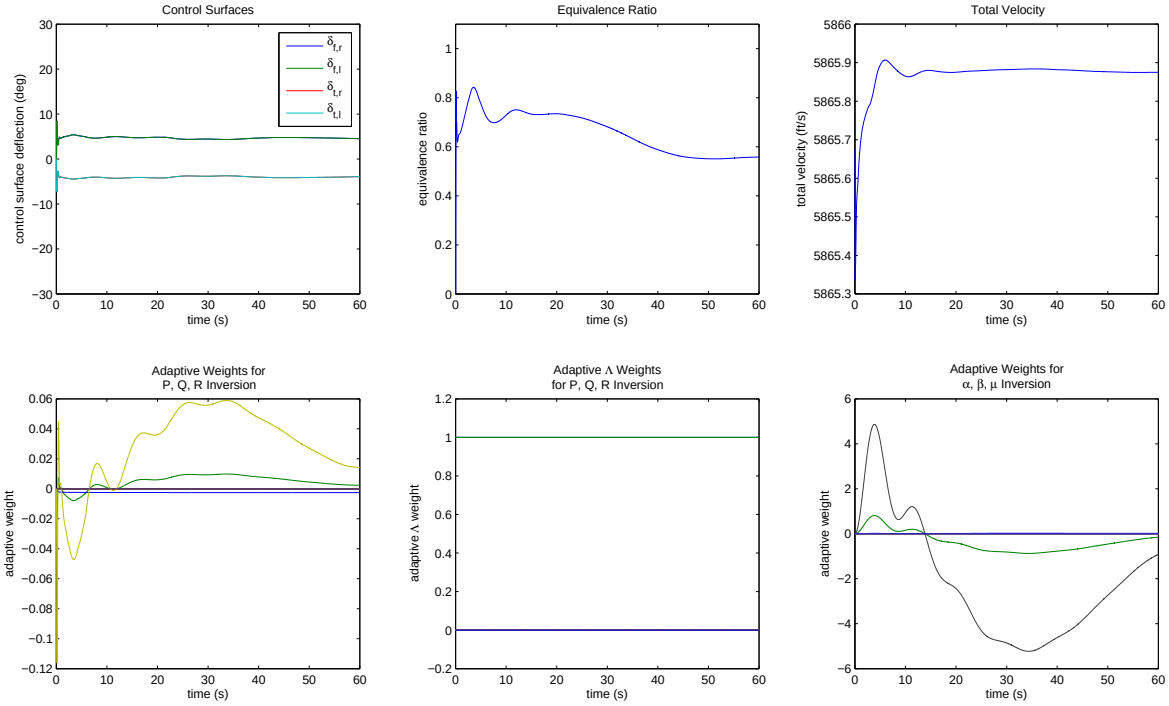


Figure 8. Control and adaptive weight responses for the generated flight path angle trajectory.

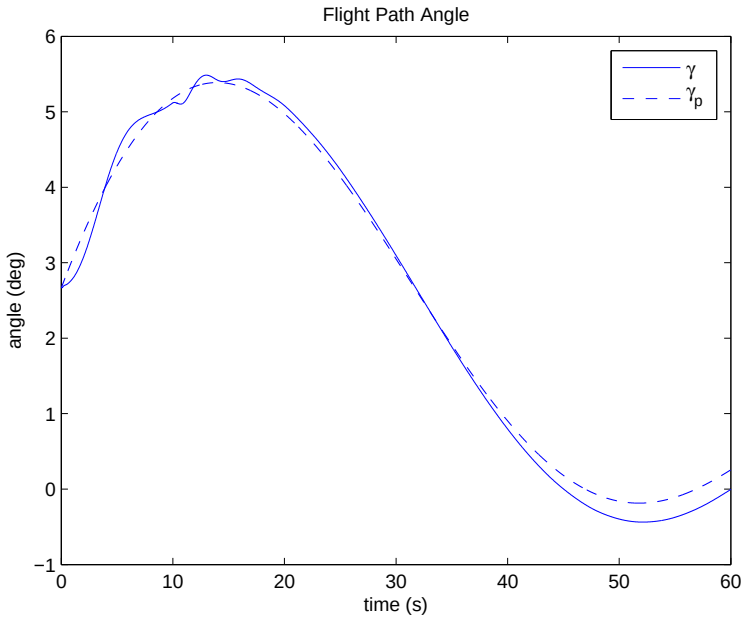


Figure 9. Flight path angle response compared with the generated flight path angle trajectory during an inlet unstart. The subscript p represents the flight path angle computed from the polynomial fit of $\dot{h}$.

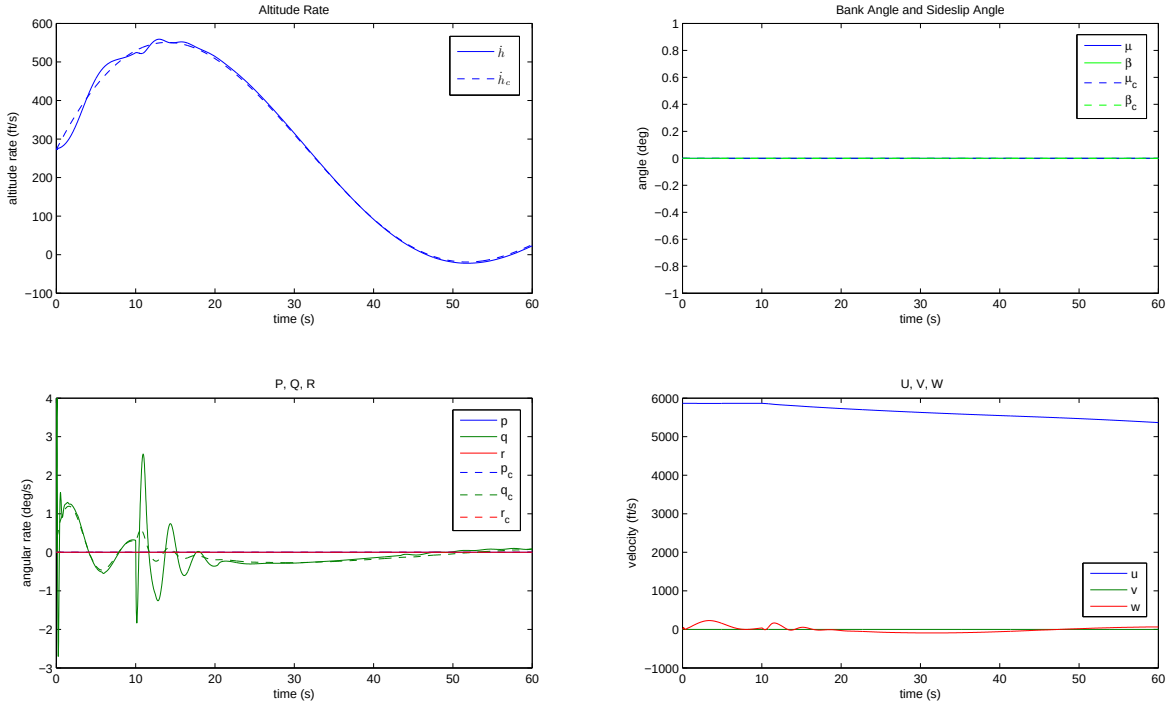


Figure 10. State responses for the generated flight path angle trajectory during an inlet unstart.

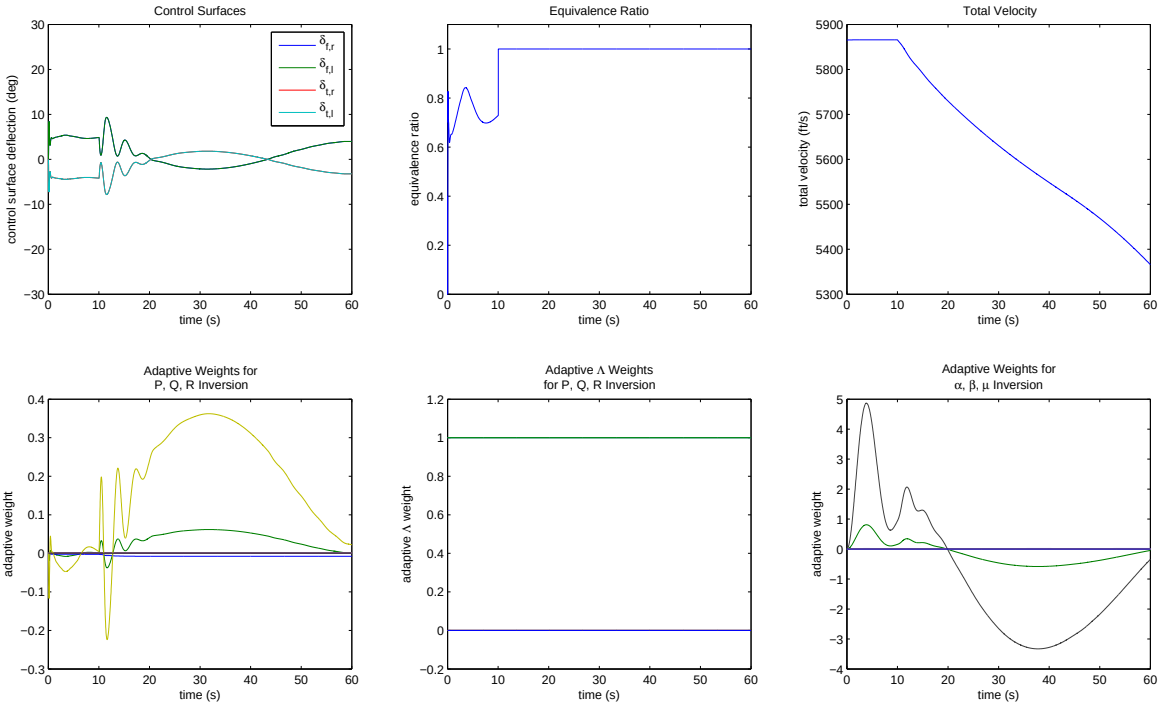


Figure 11. Control and adaptive weight responses for the generated flight path angle trajectory during an inlet unstart.

path angle trajectory. Therefore, it can be concluded that this approach of nonlinear adaptive dynamic inversion control works well as a control architecture for the GHV.

Acknowledgements

This research was funded in part by the Air Force Research Laboratory/Air Vehicles Directorate Summer Researcher Program. The author gratefully acknowledges this support. Approved for Public Release; Distribution Unlimited. Case Number 88ABW-2013-3391.

References

[1]Heiser, W. H. and Pratt, D. T., *Hypersonic Airbreathing Propulsion*, AIAA Education Series, American Institute of Aeronautics and Astronautics, Washington D. C., 1994.

[2]Annaswamy, A. M., Jang, J., and Lavretsky, E., "Adaptive gain-scheduled controller in the presence of actuator anomalies," *AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, August 2008.

[3]Gibson, T. E. and Annaswamy, A. M., "Adaptive Control of Hypersonic Vehicles in the Presence of Thrust and Actuator Uncertainties," *AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, August 2008.

[4]Groves, K. P., Sigthorsson, D. O., Serrani, A., Yurkovich, S., Bolender, M. A., and Doman, D. B., "Reference Command Tracking for a Linearized Model of an Air-breathing Hypersonic Vehicle," *AIAA Guidance, Navigation and Control Conference and Exhibit*, San Francisco, California, August 2005.

[5]Bolender, M. A., Staines, J. T., and Dolvin, D. J., "HIFiRE 6: An Adaptive Flight Control Experiment," *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, Nashville, TN, January 2012.

[6]Johnson, E. N., Calise, A. J., Curry, M. D., Mease, K. D., and Corban, J. E., "Adaptive Guidance and Control for Autonomous Hypersonic Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 3, May-June 2006, pp. 725–737.

[7]Fiorentini, L., Serrani, A., Bolender, M. A., and Doman, D. B., "Nonlinear Robust Adaptive Control of Flexible Air-breathing Hypersonic Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 32, No. 2, Mar.-Apr. 2009, pp. 401–416.

[8]Parker, J. T., Serrani, A., Yurkovich, S., Bolender, M. A., and Doman, D. B., "Approximate Feedback Linearization of an Air-breathing Hypersonic Vehicle," *AIAA Guidance, Navigation and Control Conference and Exhibit*, Keystone, Colorado, August 2006.

[9]Parker, J. T., Serrani, A., Yurkovich, S., Bolender, M. A., and Doman, D. B., "Control-Oriented Modeling of an Air-Breathing Hypersonic Vehicle," *Journal of Guidance, Control, and Dynamics*, Vol. 30, No. 3, May-June 2007, pp. 859–869.

[10]Snell, S. A., Enns, D. F., and Garrard Jr., W. L., "Nonlinear Inversion Flight Control for a Supermaneuverable Aircraft," *AIAA Guidance, Navigation, and Control Conference Technical Papers*, Vol. Part 1, American Institute of Aeronautics and Astronautics, Washington D. C., August 1990, pp. 808–825.

[11]Vick, T. J., "Documentation for Generic Hypersonic Vehicle Model," Tech. rep., U.S. Air Force Research Laboratory, Wright-Patterson AFB.

[12]Miele, A., *Flight Mechanics*, Vol. 1: Theory of Flight Paths, Addison-Wesley Publishing Company, Inc., Reading, 1962.

[13]Pomet, J.-B. and Praly, L., "Adaptive Nonlinear Regulation: Estimation from the Lyapunov Equation," *IEEE Transactions on Automatic Control*, Vol. 37, No. 6, June 1992, pp. 729–740.

[14]Slotine, J.-J. E. and Li, W., *Applied Nonlinear Control*, Prentice Hall, Inc., Upper Saddle River, 1991.

[15]Stevens, B. L. and Lewis, F. L., *Aircraft Control and Simulation*, Wiley, Hoboken, 2nd ed., 2003.

[16]Schmidt, D. K., *Modern Flight Dynamics*, McGraw-Hill, New York, 2012.

[17]Johnson, E. N., *Limited Authority Adaptive Flight Control*, Ph.D. thesis, Georgia Institute of Technology, November 2000.

[18]Johnson, E. N. and Kannan, S. K., "Adaptive Flight Control for an Autonomous Unmanned Helicopter," *AIAA Guidance, Navigation and Control Conference and Exhibit*, Monterey, California, August 2002.

[19]Valasek, J., "Optimal Setpoint Controller for a Generic X-29A Aircraft with Forebody Vortex Nozzles," *Proceedings of the 1996 IEEE International Conference on Control Applications*, Dearborn, Michigan, September 1996.

[20]Stengel, R. F., *Stochastic Optimal Control: Theory and Applications*, Wiley, New York, 1986.

[21]Menon, P., Badgett, M., Walker, R., and Duke, E., "Nonlinear Flight Test Trajectory Controllers for Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 10, No. 1, Jan.-Feb. 1987, pp. 67–72.

AIAA SciTech
13-17 January 2014, National Harbor, Maryland
52nd Aerospace Sciences Meeting

SciTech 2014, 13-17 January 2014, National Harbor, Maryland

Autonomous Soaring Using Reinforcement Learning for Trajectory Generation

Tim Woodbury*, Caroline Dunn†, and John Valasek‡
Texas A&M University, College Station, TX 77843-3141

Autonomous soaring is a concept in which the endurance of unmanned aircraft can be increased by exploiting wind updrafts. Recent research has explored traditional feedback control methods for autonomous navigation of vehicles to thermal updrafts. This paper develops an approach for planar lateral/directional guidance of a linear dynamic gliding aircraft to a known thermal location. Reinforcement learning is utilized to generate reference bank angle commands for directing the aircraft to close proximity of the updraft, and from there the aircraft follows a circling trajectory centered on the thermal to gain energy. A Lyapunov-based feedback control law is used to generate bank angle commands when circling the thermal. By using reinforcement learning the problem of online trajectory generation is reduced to a simple search in a static state-action value table. This approach has the advantage of low computational burden/overhead in practice. Furthermore, the need for a precise aircraft model for learning and simulation is reduced. Monte Carlo results presented in the paper demonstrate that the reinforcement learning guidance agent can consistently navigate the aircraft to the thermal. Reliable navigation is achieved after a relatively small number of learning episodes. An analysis of typical energy gains circling a thermal of constant shape and size is also presented. These results indicate that the approach is a suitable candidate for autonomous soaring.

Nomenclature

β	aircraft sideslip angle
p	aircraft roll rate
r	aircraft yaw rate
ϕ	aircraft bank angle, defined as positive when rolling right
ψ	aircraft heading angle, defined as positive when east of north
δ_a	aileron deflection angle
δ_r	rudder deflection angle
x_k	value of the discrete-time variable x at time $t = T(k-1)$, where T is the sample period
α	step-size parameter for reinforcement learning
γ	discount-rate parameter in reinforcement learning
$Q*$	optimal state-action value function
Q	learned, approximate state-action value function
s	learning state
a	available actions, given the learning state s
ϵ	frequency with which exploratory actions are taken under ϵ–greedy policy

*Graduate Research Assistant, Vehicle Systems & Control Laboratory, Department of Aerospace Engineering, Student Member AIAA, twoodbury@tamu.edu
†Undergraduate Research Assistant, Vehicle Systems & Control Laboratory, Department of Aerospace Engineering, Student Member AIAA, cmdunn@neo.tamu.edu
‡Professor and Director, Vehicle Systems & Control Laboratory, Department of Aerospace Engineering, Associate Fellow AIAA, valasek@tamu.edu

I. Introduction

Wind velocity gradients in the atmosphere enable long-duration missions of gliding aircraft. By flying near wind updrafts, gliders can gain altitude with a minimal expenditure of energy. This process is referred to as "static soaring." Thermals are one source of updrafts and are caused by uneven heating of the earth's surface, leading to localized regions of heated air that rise.[1] Static soaring is the process by which a gliding vehicle gains altitude (and ultimately energy) by flying through or circling an updraft. Thermals are the most common updraft source for manned gliders and have been exploited since the early 20th century to extend the range and endurance of gliding flight.[2] Glider pilots may identify thermals by noticing the formation of cumulus clouds, the composition of the local terrain, or birds circling the updraft. Within the last fifteen years,[3] thermal soaring has attracted attention as means of extending the mission duration of unmanned air systems (UAS). Miniaturized sensors enable small UAS to accomplish tasks that previously would have required a manned-scale vehicle. However, small UAS lack the range and endurance of larger vehicles due to the relatively low energy-to-weight ratio of batteries and motors at this scale.[4] Thermal updrafts represent a way for small UAS to extend their flight time, and by exploiting updrafts a battery-powered electric UAS can theoretically achieve endurance comparable to that of a solar-powered aircraft while supporting a higher wing loading and smaller battery size. Compared to a solar-powered aircraft, the soaring UAS has greater flexibility to carry larger payloads or to satisfy performance constraints.[3]

Most of the flight test results in literature have performed some variation on monitoring the aircraft's total energy to infer the presence of thermals from local air currents.[5,6,4] Some research has considered remote detection of the thermal from the UAS using visual or infrared (IR) imaging. Akhtar[7] conducted field trials with an IR camera, and demonstrated that the camera could detect "hot spots" on the ground as well as clouds, both of which correlate with the presence of updrafts. Another work describes calibration of a thermal infrared camera for remote sensing onboard a UAS.[8] Although thermal soaring was not considered, a similar architecture might be used for remote detection and exploitation of updrafts.

A variety of approaches have been investigated to achieve autonomous soaring. Ref. 4 implemented an asymptotically stable thermal centering control and validated the work in flight test. Ref. 9 simultaneously explored and exploited an unknown wind field using a Gaussian Process model for mapping. A receding horizon framework for optimizing energy gain based on local knowledge of the wind field was implemented in Ref. 10. This study considered both thermal soaring and dynamic soaring in which energy is absorbed from local wind gusts. Allen and Lin[5] used a guidance and control algorithm in which the aircraft estimated the location of a thermal based on energy measurements and an assumed thermal profile. Flight test results were presented and showed an average energy gain of 173 m per thermal over seventeen flights. In one case the effective endurance was extended from two to fourteen hours.[11] Edwards[6] presents flight test results in which an autonomous glider traveled 48 km over 1.5 hrs from an initial altitude of 140 m. In this approach the local updraft speed was estimated and a grid of nodes was evaluated to determine the most likely center of a parameterized thermal.

This paper develops a novel concept for autonomous navigation using reinforcement learning (RL) to generate bank angle commands that navigate an unmanned glider to a thermal updraft at a known location. This approach is distinct from previous works, which generally localize thermals using the energy rate of change with some form of feedback control for guidance and navigation. The RL algorithm used is Q-learning, in which an approximation to the optimal state-action value mapping is learned and stored. This control scheme has three primary advantages. First, by allowing low-level feedback controllers determine the required control deflections, the reinforcement learning agent can be trained in a theoretically model-agnostic fashion, so long as the vehicle can match the commanded bank angle sufficiently quickly. This fact should also make the RL guidance law robust to some plant uncertainties, reducing the need for very accurate modelling in simulation. Second, by utilizing RL to generate bank angle commands, the problem of generating reference trajectories to the updraft is essentially reduced to a table lookup problem. This arrangement allows for a computationally simple control scheme consisting of the RL lookup for navigation with generic state feedback command and hold controllers to determine control deflections. A small UAS platform is assumed to have limited onboard capacity to carry sensors and computing equipment, and reducing the navigational overhead frees up computing power for other tasks. Furthermore, the navigation component can be easily integrated with existing autopilot hardware and software. Third, the use of Q-learning gives a designer flexibility to tailor and optimize performance by shaping rewards appropriately.

The paper is organized as follows. The RL algorithm and its application to the aircraft guidance problem are considered. This is followed by modelling and simulation of the vehicle. Results of a preliminary

implementation using RL for lateral/directional control with conventional longitudinal control are then presented, and conclusions and plans for extending this implementation are given.

II. Control Policy

II.A. Q-learning

Reinforcement learning describes a general class of algorithms that attempt to learn state-action mappings to maximize a reward signal. With Q-learning, a decision agent seeks to learn the optimal action-value function, $Q^*(a_t, s_t)$, which is the value of taking action a_t from the state s_t. The algorithm learns Q, an approximation to Q^*. For learning, an ϵ–greedy policy is used. Under this policy, the decision agent takes randomly selected actions with frequency ϵ, and takes the action with the highest value with frequency $1-\epsilon$. The value of ϵ is varied from 1.0 initially, to encourage learning, to a minimum of 0.05 over the course of learning. The decision agent receives rewards based on the states visited. From any given initial state, the goal of the learning agent is to maximize the total reward received until a terminal condition is met.[12] Q-learning has several features that make it advantageous for the autonomous soaring problem. Q-learning is guaranteed to converge to Q^*, although it requires a theoretically infinite number of learning episodes. The convergence to Q^* is also policy-independent. Lastly, learning can be model-free, in that the planning agent does not require an explicit model of the system.

The Q-learning algorithm is straightforward. In the update equation for $Q(s,a)$, α is a step-size parameter and γ is a discount-rate parameter that influences the extent to which received rewards influence the learned value of subsequent states. Primed quantities represent first future value.

1. $Q(s,a)$ is initialized arbitrarily
2. For each episode in the learning:
   (a) Initialize at state s
   (b) For each step in the episode:
      i. Choose action a from s via policy
      ii. Observe the next state, s'
      iii. $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
      iv. $s \leftarrow s'$
   (c) Break when s is a terminal state

II.B. Q-learning for autonomous soaring

Previous work[13] has demonstrated the use of Q-learning for navigation of a powered vehicle. In the current work, a linear dynamic sailplane model is implemented and the performance of RL for lateral/direction guidance is evaluated. Two primary assumptions are made in developing the RL controller. First, the glider possesses feedback controllers sufficient to match bank angle commands; second, the location of the thermal center is known *a priori* by the vehicle. The first assumption can be readily achieved by even hobby-grade equipment.[14] The second assumption is somewhat more limiting. Remote detection using an IR camera or other sensor is the assumed *modus operandi*. The use of RL is restricted to guidance of the UAS to the proximity of the thermal; from this point a feedback-based circling controller is used to evaluate the altitude gain at the thermal. Longitudinal-axis control is effected by a state feedback regulator about a steady-state. The RL guidance control is discrete and is wrapped around a continuous-time Proportional-Integral-Derivative (PID) controller for bank angle command and hold. The only requirement of the PID controller is that it consistently match the commanded bank angle faster than the update rate of the RL guidance algorithm, without destabilizing the system or driving other states to unacceptably large values. The RL guidance algorithm can be summarized as follows:

For each episode:

1. Initialize the state: r, the range to the thermal, λ, azimuth to the thermal relative to aircraft heading, ϕ, current bank angle
2. Loop over 500 discrete steps, or until a break condition is met:

(a) Choose action from available commanded changes in bank: $\Delta = \begin{bmatrix} 0 & -\Delta\phi & \Delta\phi \end{bmatrix}$.

(b) Simulate the system with commanded bank angle $\phi + \Delta$ for T_Q seconds, using the continuous-time bank angle and elevator controls.

(c) Observe the next state. If $r > R_1$, where R_1 is the maximum allowed range to the thermal, receive a penalty reward r_b and break. If $r < R_2$, where R_2 is the effective radius of the updraft, receive a goal reward r_g and break the loop. Else, receive no reward, and continue.

Once learning is conducted with one vehicle model, the Q-learning update rate T_Q can be tailored to provide guidance for any vehicle, regardless of its forward speed. This could eliminate the need for re-learning the Q-matrix whenever the plant model changes. In practice, the low-level bank angle controller also influences performance so it may be difficult to select the appropriate value of T_Q for any glider. It is anticipated that the Q-matrix from one vehicle can be used to "seed" an initial Q-matrix for a different vehicle, reducing the overall learning time.

III. Aircraft Modelling and Simulation

III.A. Aircraft model

The dynamical aircraft model is based on a Schweizer SGS 1-36 sailplane (Fig. 1). The SGS 1-36 is a manned, single-seat advanced trainer.[15] Ref. 1 gives coefficients for a linear model of an SGS 1-36 modified for flight at high angles of attack, using values derived from flight test data in Ref. 15. The continuous-time linear model and steady-state values used in simulation are given in Appendix A. The SGS 1-36 is not an ideal match for the intended application of this RL controller, as it is substantially heavier and faster than what would be considered a "small" unmanned glider. However, it is assumed that a manned glider is more representative of the intended system dynamics than a powered unmanned vehicle model, relatively few gliding vehicle models are available in the literature, and using a large vehicle allows future evaluation of how well the Q-learning will scale to a smaller glider.

Figure 1. Schweizer SGS 1-36 sailplane in flight. Source: "Schweizer 1-36 Photo Collection," Dryden Flight Research Center, Accessed 2013/06/04, http://www.dfrc.nasa.gov/Gallery/Photo/Schweizer-1-36/HTML/ECN-26847.html

III.B. Q-learning settings

For the purposes of learning, the maximum allowed range to the thermal is $R_1 = 1200$ m. If the aircraft flies too far away, the learning agent receives a penalty and the episode terminates. A goal reward is given if the aircraft flies within the effective updraft radius, which is approximately 460 m. The goal and boundary rewards are $+20$ and -20 respectively. The maximum bank angle is $\pm 30°$, and if the RL agent attempts to command a bank angle outside that range, another action is selected and the simulation continues. A timestep of $T_Q = 5$ sec is used for the navigation problem, and the RL agent commands changes in bank angle of $\Delta\phi = 5°$.

III.C. Low-level control laws

The bank angle control law is a simple continuous-time PID aileron control law given by Eq. (1). In this equation, ϕ_r is the reference bank angle generated from the RL agent, and is treated as piecewise constant between RL updates. The gains used in simulation are $K_p = 5.5, K_i = 0.1, K_d = 7.0$ and the maximum commanded aileron deflection allowed is 25°. The gains are selected so that the vehicle can match a 5°commanded change in bank to with 0.01°in five seconds, without exceeding an aileron deflection of 5°.

$$\delta_a = \begin{cases} \delta_{pid} = -K_p(\phi - \phi_r) - K_i \int_0^t (\phi(t) - \phi_r)\mathrm{d}t - K_d\dot{\phi} & -\delta_{max} < \delta_{pid} < \delta_{max} \\ -\delta_{max} & \delta_{pid} < -\delta_{max} \\ \delta_{max} & \delta_{pid} > \delta_{max} \end{cases} \tag{1}$$

The elevator control law is a simple Linear Quadratic Regulator (LQR) state feedback control law given in Eq. (2), where u is the perturbed body 1-axis speed, α is the perturbed angle-of-attack, q is the body 2-axis (pitch) angular rate, and θ is the perturbed pitch attitude angle. The LQR weighting matrices were selected to closely regulate u and α and tuned to produce well-damped convergence to the steady-state in the presence of the external updraft during circling flight.

$$\delta_e = \begin{bmatrix} 0.1416 & 1.925 & -0.4718 & -1.479 \end{bmatrix} \begin{bmatrix} u \\ \alpha \\ q \\ \theta \end{bmatrix} \tag{2}$$

III.D. Thermal model

The thermal updraft model is taken from Ref. 11. In the following equations, w^* and z_i are convective-layer scale parameters. Updraft velocity is calculated using Eq. (3), in which z is the height-above-ground altitude of the aircraft. Note that in the convention used in Eq. (3) z is defined as positive up.

$$w_T = w^* \left(\frac{z}{z_i}\right)^{\frac{1}{3}} (1 - 1.1\frac{z}{z_i}) \tag{3}$$

Updraft diameter D increases exponentially with increasing altitude:

$$D = 0.203 \left(\frac{z}{z_i}\right)^{\frac{1}{3}} (1 - 0.25\frac{z}{z_i}) z_i \tag{4}$$

Note that in 11, conservation of mass is used outside the updraft to calculate a downdraft velocity. For the purposes of simulation downdraft velocity in this study is assumed negligibly small. This updraft model produces a radially symmetric thermal. The updraft parameters used are: $w^* = 2.56\ \frac{\mathrm{m}}{\mathrm{s}}$ and $z_i = 660$ m, and the updraft diameter is scaled so that the updraft radius at a 300 m altitude is approximately 460 m. The updraft is applied as a disturbing input on the aircraft u, α, and β channels. The question of whether the energy gained from circling a thermal is worth the energy cost of flying to it is not considered here. Consequently, when thermal circling effectiveness is examined, an updraft of constant size and strength is used in all simulations. Since evaluation of the circling controller is not the primary objective, the updraft size is effectively arbitrary. Two factors are considered in sizing:

1. The updraft radius is chosen to be wide enough that the SGS 1-36 model can perform steady level turns at a bank angle of 30°or less at an altitude of 300 m.

2. the updraft height z_i is chosen so that most simulations using the circling feedback control law will experience a net loss in energy if circling continues for 1000 seconds or less.

The purpose of the circling feedback control law is to establish a reasonable performance baseline against which an RL-based circling controller can be evaluated. For numerical convenience, a relatively weak thermal is selected so that the total time the aircraft can circle before it begins losing energy can be compared using relatively short simulation times.

III.E. Thermal circling control law

A feedback control law is used to create baseline circling results against which a later RL-based controller can be evaluated. The objective of this controller is only to provide a baseline of reasonably good performance, with no consideration of optimizing any performance metric. Since the baseline controller need not be implemented in experiment, real-time computational tractability is largely ignored. A Lyapunov-based feedback control law is designed that guarantees convergence to a reference trajectory under the assumption of constant speed level turns. This approximation is found to yield acceptable performance.

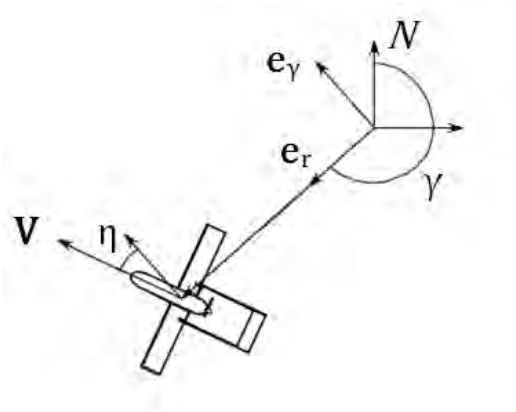


Figure 2. Coordinate system used for circling control law. Values are positive as shown.

The coordinate system referenced for the circling control is shown in Fig. 2. An inertial north-east-down reference frame is used. The variable to be controlled is η, the angle between the aircraft X-Y plane velocity vector $\mathbf{V}$ and the normal to the X-Y plane position vector $\hat{\mathbf{e}}_\gamma$. η can be written in terms of the position vector $\mathbf{r}$ and velocity vector as in Eq. (7):

$$\mathbf{r} = r\hat{\mathbf{e}}_r \tag{5}$$

$$\frac{d\mathbf{r}}{dt} = \mathbf{V} = \dot{r}\hat{\mathbf{e}}_r + r\dot{\gamma}\hat{\mathbf{e}}_\gamma \tag{6}$$

$$\tan\eta = \frac{\dot{r}}{r\dot{\gamma}} \tag{7}$$

Defining the error coordinate $e_\eta = \eta - \eta_{ref}$ in terms of η and a reference angle η_{ref}, a stabilizing control law can be developed beginning with the Lyapunov function $V = \frac{1}{2}e_\eta^2$. Its time rate and a stabilizing controller are given by Eqs. (8) and (9).

$$\dot{V} = e_\eta\left(\frac{r\ddot{r}\dot{\gamma} - \dot{r}^2\dot{\gamma} - r\dot{r}\ddot{\gamma}}{\dot{r}^2 + (r\dot{\gamma})^2} - \dot{\eta}_{ref}\right) \tag{8}$$

$$-K_L e_\eta = \frac{r\ddot{r}\dot{\gamma} - \dot{r}^2\dot{\gamma} - r\dot{r}\ddot{\gamma}}{\dot{r}^2 + (r\dot{\gamma})^2} - \dot{\eta}_{ref} \tag{9}$$

The second derivative of the X-Y position vector is:

$$\frac{d^2\mathbf{r}}{dt^2} = (\ddot{r} - r\dot{\gamma}^2)\hat{\mathbf{e}}_r + (2\dot{r}\dot{\gamma} + r\ddot{\gamma})\hat{\mathbf{e}}_\gamma \tag{10}$$

Under the assumption of steady, level, 1 g turns the aircraft acceleration vector is $-g\tan\phi\hat{\mathbf{e}}_r$. Under this assumption the second derivatives of r and γ are given by Eqs. (11) and (12). Under the further assumption that the magnitude of the velocity $\mathbf{V}$ is approximately the steady-state forward speed U_1, the continuous-time control law used to generate reference bank angles is given by Eq. (13):

$$\ddot{r} = r\dot{\gamma}^2 - g\tan\phi \tag{11}$$

$$\ddot{\gamma} = \frac{-2\dot{r}\dot{\gamma}}{r} \tag{12}$$

$$g\tan\phi = r\dot{\gamma}^2 - \frac{U_1^2(\dot{\eta}_{ref} - K_L e_\eta) - \dot{r}^2\dot{\gamma}}{r\dot{\gamma}} \tag{13}$$

Reference trajectories are generated by computing polynomial fits for $r(\gamma)$ that connect an initial point r_0, γ_0 to a final point on the desired circular trajectory at $\gamma_0 + \pi$. Continuity of r and its first derivative at the initial state are enforced along the reference, and the target state is $r = R_d, \dot{r} = 0$. Starting from the controller used in Ref. 11, the target radius is eventually selected to be $R_d = 0.6R_2$, with R_2 being the local

updraft outer radius. New trajectories are computed every half-orbit of the updraft and the aircraft is found to converge to a circular trajectory within one or two orbits typically. The feedback gain $K_L = 0.25$ gives acceptable results.

III.F. Simulation framework

Fig. 3 shows the coordinate system used for learning and its relation to the dynamical system coordinates used. Inertial position is given by a north-east-down reference frame centered at the thermal at ground level. A standard body-fixed reference frame is attached to the aircraft with the 1-axis out the nose, the 2-axis along the right wing, and the 3-axis down. The body frame is related to the inertial frame via the standard 3/2/1 Euler angle rotation sequence through $\psi/\theta/\phi$.

The variables r and γ are polar coordinates for the X-Y plane position vector of the vehicle. Note that r is the length of the projected position vector and is always positive. λ is the angle between the X-Y plane projection of the aircraft body 1-axis and the vector from the aircraft to the thermal; λ is positive when the thermal is out the right wing. λ is related to heading angle ψ and γ by Eq. (14):

$$\pi + \gamma - \lambda = \psi \tag{14}$$

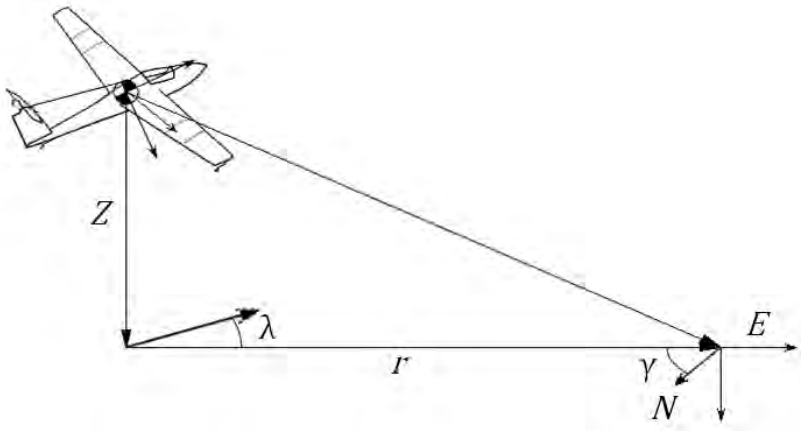


Figure 3. Coordinate system used for RL state.

For learning the Q-matrix in the navigation problem, the glider is initialized with uniformly distributed random states 500 m $\leq r \leq$ 700 m, $-135° \leq \lambda \leq 135°$ and zero bank angle. The inertial state is always initialized at $Z = -300$ m and at arbitrary X, Y, ψ coordinates that satisfy the relative placement specified by λ. All other aircraft perturbed states are assumed zero.

IV. Numerical Results

The RL navigation agent is trained on 100,000 episodes using an ϵ-greedy policy. 100,000 episodes was found to be a sufficient number to achieve acceptable performance. The RL agent was evaluated in 1,000 Monte Carlo simulations, each of which terminates when the aircraft either flies within the updraft radius R_2 or outside the boundary R_1, or after 500 discrete timesteps. Subsequently, the circling control law is evaluated in 1,000 simulations, each starting at the final state of an RL evaluation episode. Results are presented in two sections. The first section displays performance of the RL navigation agent with representative episodes and analyzes the Monte Carlo results. In the second section the baseline circling performance is considered.

IV.A. RL navigation Monte Carlo results

For Monte Carlo simulations, the set of initial conditions is expanded to 250 m $\leq r \leq$ 700 m, $-15° \leq \phi \leq 15°$ with $-135° \leq \lambda \leq 135°$ as before. A graphical summary of the success and failure cases as a function of initial conditions is shown in Fig. 5. In 1,000 simulations the RL agent correctly navigates to the thermal in 962 of the cases. The cases in which the agent failed mostly correspond to nonzero initial bank angles at or

near extrema of range and azimuth. The aircraft presumably visited these states infrequently if at all during learning, so failure is predicated on whether the initial conditions naturally drive the aircraft towards or away from states the RL agent has learned. Since the navigation agent is successful in a high percentage of cases, it is reasonable to expect that with additional learning the agent could consistently reach the thermal from this expanded set of initial conditions. It is important to note that when the agent is evaluated on 1,000 episodes using the same initial conditions used for learning in Sec. III.F, the agent succeeds in all cases.

The altitude lost in navigating to the thermal in Monte Carlo simulations is shown in Fig. 4. This is used as a performance metric. Note that the cases in which the UAS did not reach the thermal are not shown. The distribution exhibits two distinct modes for altitude loss of less than 100 m. There are also 26 flights with distinctly greater altitude loss than the others. The latter group demonstrate inefficient flight paths in which the vehicle typically rotates through a full 2π rotation in heading before reaching the thermal. This behavior could be caused by the use of new initial conditions in Monte Carlo, or could simply have occurred because the RL reward structure does not give any preference to short paths. The bimodality of the remainder of the distribution is unexpected, since the initial conditions were chosen uniformly. The bimodal distribution is presumed to be either an effect of not using a time-optimal reward structure or a natural consequence of the dynamics of the problem.

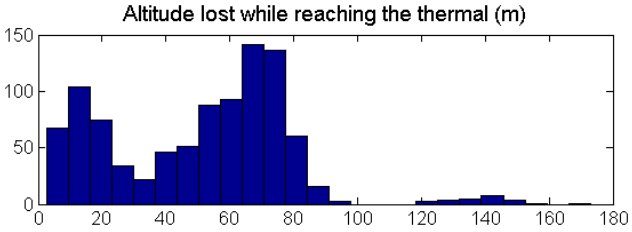
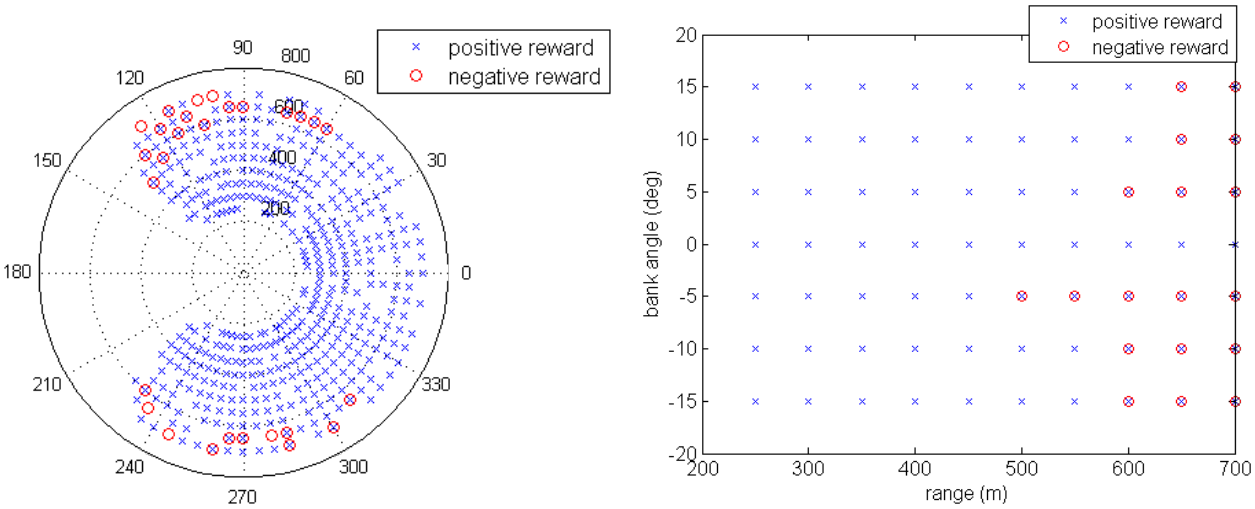


Figure 4. Altitude loss in reaching the thermal with RL navigation.

Plots of ground tracks and bank angle histories for three simulations are shown in Fig. 6. The initial conditions shown were selected arbitrarily from the set of Monte Carlo initial conditions, for the purpose of showing typical histories only.



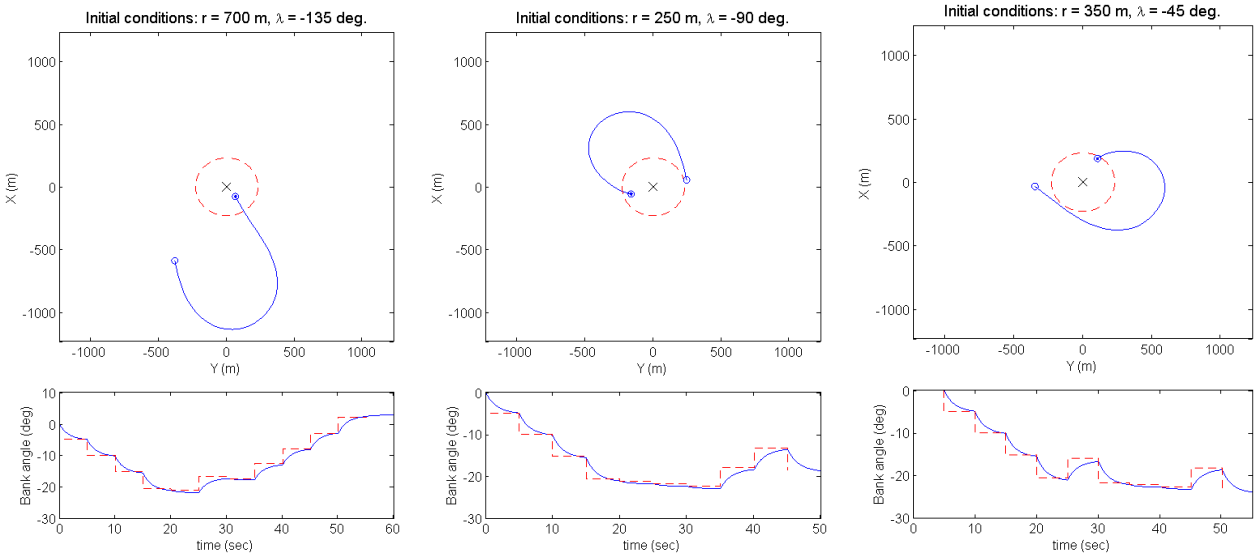
(a) Initial range versus azimuth.

(b) Initial bank angle versus range.

Figure 5. Summary results of Monte Carlo success/failure versus initialization conditions.

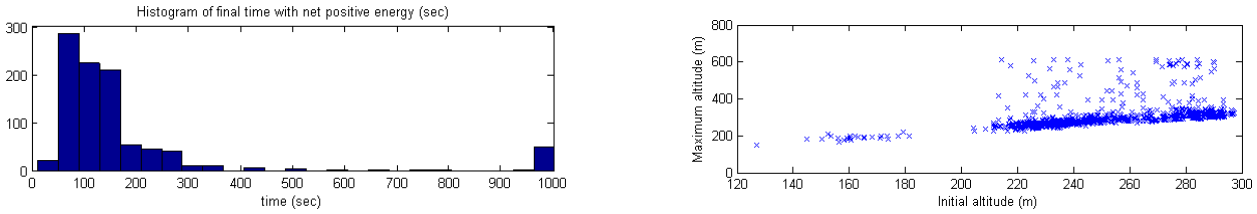
IV.B. Baseline circling results

The thermal circling baseline results are generated by evaluating 1,000 simulations. Each simulation starts at the final state of a Monte Carlo navigation episodes, using the fixed thermal described in Sec. III.D. The thermal is deliberately weak for this scale of vehicle for numerical convenience in future comparison with an RL circling controller. The circling implementation does not stop circling at peak altitude; for simplicity, state histories are simulated for 1,000 seconds and the aircraft continues circling until either the time runs out or the aircraft hits the ground. Results are evaluated in two figures. First, the time-of-flight before the current kinetic and potential energy normalized by mass is less than its initial value is computed. A histogram of time-of-flight is shown in Fig. 7(a). Note that in 51 cases, the aircraft energy was still increasing after 1000 seconds. The remainder of the cases are significantly right skewed, having a median of 118.2 and mean of 151.3 sec.



(a) Case 1 (b) Case 2 (c) Case 3

Figure 6. Representative X-Y position and bank angle histories showing RL commands. Initial position is indicated by an open circle and final position is indicated by a circled dot.



(a) Histogram of circling time-of-flight for which altitude remains greater than altitude when circling started.

(b) Scatterplot of peak altitude versus initial altitude for circling.

Figure 7. Thermal circling results.

Second, peak altitude is plotted as a function of initial altitude in Fig. 7(b). For a majority of cases, the relationship appears to be roughly linear. However, there are approximately 140 cases that exhibit very different behavior. These cases are scattered roughly uniformly between initial altitudes of 210 and 290 meters and have peak altitudes bounded above by about 650 meters and bounded below by the approximately linear behavior of the other data. While this second set does appear to contain the 51 cases in which energy had not yet peaked, there are still 90 cases for which the cause of differing behavior has not been explained. Additional analysis did not determine any trend between initial conditions and peak altitude gain which might be explain the very different results for these 90 cases. Despite this somewhat unexpected behavior, the circling controller provides a baseline against which future RL-based circling can be evaluated. The mean difference in initial and maximum altitude is 50.1 m with a standard deviation of 69.9 m.

V. Conclusions

This paper presents the development and initial simulation results for reinforcement learning-based navigation of a gliding vehicle to reach a remotely detected thermal updraft. This approach has the advantage of low computational overhead in practice. It is model-agnostic if certain conditions are met so learning should be transferable between different vehicles. By using Q-learning, a significant flexibility to tailor performance via reward shaping is introduced. Numerical simulation is performed using a simulated Schweizer SGS 1-36 glider dynamic model with a constant updraft. Monte Carlo results show that the RL guidance agent navigates to the updraft in 100% of cases when initialized at the same states used to initialize learning. When the set of initial conditions is expanded to include initial bank angles and ranges not used in learning, the navigation agent succeeds in 96.2% of Monte Carlo cases. This performance is achieved after a relatively

small number of learning episodes (100,000). A baseline analysis of altitude and flight time gains from circling the thermal is performed using a Lyapunov-based circling law. This control law establishes a baseline mean altitude gain of 50.1 m.

For future work, four primary areas for further study have been identified. First, it is of interest to compare the RL guidance agent against a time-optimal agent as a metric of navigation efficiency. It may be appropriate to weight the learning rewards so as to give a greater reward for reaching the updraft quickly, as there is currently no mechanism to ensure that the agent selects a direct route. Second, the issue of disturbance tolerance in the trajectory generation should be considered. The commanded bank angle is essentially open-loop for the duration of the reinforcement learning timestep; it may be necessary to either reduce the timestep or implement a system to detect if tracking the reference angle is causing other aircraft states to diverge. Third, the ability of an RL guidance agent to fly close to an updraft and gain energy should be examined. A future work is planned that will encompass RL guidance for thermal circling as well as longitudinal-axis control for 3D flying with non-constant updrafts. Fourth and finally, the extensibility of this approach should be evaluated by extending the Q-matrix learned on the SGS 1-36 to a new plant.

Acknowledgments

This material is based upon work supported in part by the U.S. Air Force Office of Scientific Research under contract FA9550-08-1-0038, with technical monitor Dr. Fariba Fahroo. It is also supported by the National Science Foundation Graduate Research Fellowship, and by Texas A&M University from the Undergraduate Summer Research Grant program. This support is gratefully acknowledged by the authors. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Air Force.

A. Dynamic Model

The lateral-directional continuous-time dynamic model derived from Refs. 1 and 15 is:

$$\begin{bmatrix}\dot{\beta}\\ \dot{p}\\ \dot{r}\\ \dot{\phi}\end{bmatrix} = \begin{bmatrix}-0.1854 & 0.04069 & 0.9719 & -0.2984\\ 9.732 & -19.49 & 2.585 & -0.2655\\ -2.024 & 0.6734 & -0.6171 & 0.1212\\ 0 & 1 & 0 & 0\end{bmatrix}\begin{bmatrix}\beta\\ p\\ r\\ \phi\end{bmatrix} + \begin{bmatrix}0.02569 & 0.09295\\ -14.77 & -1.278\\ 0.7632 & 1.799\\ 0 & 0\end{bmatrix}\begin{bmatrix}\delta_a\\ \delta_r\end{bmatrix} \tag{15}$$

The longitudinal axis continuous-time dynamic model from the same sources is:

$$\begin{bmatrix}\dot{u}\\ \dot{\alpha}\\ \dot{q}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}-0.02451 & 5.938 & 0.4913 & -11.17\\ -0.01475 & -2.187 & 0.7916 & 0.01964\\ -0.05919 & -14.6 & 1.305 & -0.1116\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}u\\ \alpha\\ q\\ \theta\end{bmatrix} + \begin{bmatrix}-0.1833\\ -0.1432\\ -0.9847\\ 0\end{bmatrix}\delta_e \tag{16}$$

The steady-state speeds and attitude, estimated from the flight test results in Ref. 15, are listed below. Unlisted values are zero in the steady-state.

- Body-axis forward speed: $U_1 = 34.4063\ \frac{\text{m}}{\text{s}}$
- Pitch-axis Euler angle (positive nose up): $\theta_1 = -1.8873°$
- Angle of attack (positive nose up): $0.1766°$
- Body-axis vertical speed (positive down): $W_1 = 0.1061\ \frac{\text{m}}{\text{s}}$

References

[1] Boslough, M. B., "Autonomous dynamic soaring platform for distributed mobile sensor arrays," *Sandia National Laboratories, Sandia National Laboratories, Tech. Rep. SAND2002-1896*, 2002.

[2] *Glider Flying Handbook*, chap. Chapter 10: Soaring techniques, Flight Standards Service, Federal Aviation Administration, 2013, FAA-H-8083-13A.

[3]Wharington, J. and Herszberg, I., "Control of a high endurance unmanned air vehicle," Proceedings of the 21st ICAS Congress, 1998.

[4]Andersson, K., Kaminer, I., Dobrokhodov, V., and Cichella, V., "Thermal Centering Control for Autonomous Soaring; Stability Analysis and Flight Test Results," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 3, 2012, pp. 963–975.

[5]Allen, M. and Lin, V., "Guidance and control of an autonomous soaring UAV," *NASATM-2007-214611*, 2007.

[6]Edwards, D. J., "Implementation details and flight test results of an autonomous soaring controller," *North Carolina State University*, 2008.

[7]Akhtar, N., "Control system development for autonomous soaring," 2010.

[8]Sheng, H., Chao, H., Coopmans, C., Han, J., McKee, M., and Chen, Y., "Low-cost UAV-based thermal infrared remote sensing: Platform, calibration and applications," *Mechatronics and Embedded Systems and Applications (MESA), 2010 IEEE/ASME International Conference on*, IEEE, 2010, pp. 38–43.

[9]Lawrance, N. R. and Sukkarieh, S., "Autonomous exploration of a wind field with a gliding aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 34, No. 3, 2011, pp. 719–733.

[10]Depenbusch, N. T. and Langelaan, J. W., "Receding horizon control for atmospheric energy harvesting by small UAVs," *AIAA Guidance, Navigation and Controls Conference, Toronto, Canada*, 2010.

[11]Allen, M. J., "Autonomous soaring for improved endurance of a small uninhabited air vehicle," Proceedings of the 43rd Aerospace Sciences Meeting, AIAA, 2005.

[12]Sutton, R. S. and Barto, A. G., *Reinforcement Learning: An Introduction*, MIT Press, 1998.

[13]Dunn, C., Valasek, J., and Kirkpatrick, K., "Unmanned Air System Search and Localization Guidance Using Reinforcement Learning," Infotech@Aerospace 2012, 2012.

[14]"Official Arduplane Repository," https://code.google.com/p/ardupilot-mega/wiki/home?tm=6, Accessed May 24, 2013.

[15]Sim, A. G., "Flight characteristics of a modified Schweizer SGS 1-36 sailplane at low and very high angles of attacks," *NASA TP-3022*, 1990.

AIAA SciTech
13-17 January 2014, National Harbor, Maryland
AIAA Guidance, Navigation, and Control Conference

Autopilot for a Nonlinear Non-Minimum Phase Tail-Controlled Missile

Anshu Narang-Siddarth[a*], Florian Peter[b†], Florian Holzapfel[b‡] and John Valasek[c§]

[a] *William E. Boeing Department of Aeronautics and Astronautics, University of Washington, Seattle, WA 98195*

[b] *Institute of Flight System Dynamics, Munich, Germany, 85748*

[c] *Department of Aerospace Engineering, Texas A&M University, College Station, TX 77843-3141*

Acceleration control of highly agile, aerodynamically-controlled missiles is a well-known non-minimum phase control problem. This problem is revisited here for a planar tail-controlled generic missile, and a globally stable nonlinear autopilot command structure is synthesized to maximize performance. For the first time the non-minimum phase characteristics of the vehicle are addressed by making no modification to the output definition by inducing an inherent time scale separation in the closed-loop dynamics. Unlike, previous time scale control techniques, results presented here are based on theoretical advancements made in control of nonlinear singularly perturbed systems. Conditions under which the induced time scale separation can be employed for a stable autopilot design are also discussed. The state feedback controller proposed is real-time implementable, independent of operating condition and desired output trajectory. Simulation results presented in the paper show that the approach is able to accomplish perfect tracking while keeping all closed-loop signals bounded.

Nomenclature

$(a_A^G)_B$	forward and normal linear aerodynamic accelerations $[(a_{A,X}^G)_B, (a_{A,Z}^G)_B]$
$(F_X^G)_B$	horizontal force acting about the c.g of the missile represented in the body frame
$(F_Z^G)_B$	vertical force acting about the c.g of the missile represented in the body frame
$(M_Y^G)_B$	pitching-moment about the c.g of the missile represented in the body frame
$\bar{q}$	dynamic pressure
C_m	pitch-moment coefficient
C_x	horizontal force coefficient
C_z	vertical force coefficient
G	center of gravity of the generic missile
g	acceleration due to gravity
h	altitude of the missile, $= -(z^G)_I$
I_{YB}	moment of inertia of the missile about the y_B axis
K_i	feedback gains, positive quantities
l_{ref}	reference length
M	Mach number
m	mass of the generic missile model
q_K^{0B}	body pitch rate of the missile relative to the North-East-Down frame
S_{ref}	reference surface area
u_K^G	forward velocity in the body frame
v_K^G	lateral velocity in the body frame

*Assistant Professor, Member AIAA, *anshu@aa.washington.edu (Corresponding Author)*

†Ph.D Candidate, Student Member AIAA, *florian.peter@tum.de*

‡Professor, Senior Member AIAA, *florian.holzapfel@tum.de*

§Professor and Director, Vehicle Systems & Controls Laboratory, Associate Fellow AIAA, *valasek@tamu.edu, http://vscl.tamu.edu/valasek*

V_K^G total velocity of the missile
w_K^G vertical velocity in the body frame
x_0, y_0, z_0 triad representing the North-East-Down frame
x_B, y_B, z_B triad representing the body-frame, B

Subscripts

$,\alpha$ derivative with respect to angle-of-attack
$,\delta$ derivative with respect to pitch-control deflection
$,A$ aerodynamic component
$,G$ gravitational component
$,q$ derivative with respect to pitch rate
0 reference quantity
r reference trajectory

Symbols

α_K^G angle-of-attack of the missile
β_K^G side-slip of the missile
δ_M pitch control deflection
θ pitch-attitude angle of the missile
$O(.)$ Order symbol

I. Introduction

The main challenge in designing a controller for an aerodynamically controlled missile is to achieve maximum performance it is capable of, without exciting the unstable internal dynamics. To qualitatively understand this non-minimum phase behaviour consider the control problem of accelerating the missile (Figure 1) upward. Typically a tail-controlled missile (i.e control surface aft of the center of gravity, G) is statically stable with $C_{m_\alpha} < 0$, $C_{z_\delta} < 0$ and $C_{m_\delta} < 0$. This means that a negative unit-step pitch deflection command initially induces a downward force on the missile causing the missile to accelerate downward. This downward force also induces a counter-clockwise pitching-moment about the center of gravity that tries to push the nose-up. But due to the inherent tendency of the missile to oppose any such change in angle-of-attack the missile continues to accelerate downward until an overall positive pitching moment about the center of gravity develops. Eventually the trim angle-of-attack and consequently the lift acting on the vehicle increase which together create an upward force about the fuselage; and thus the missile accelerates upward as desired.

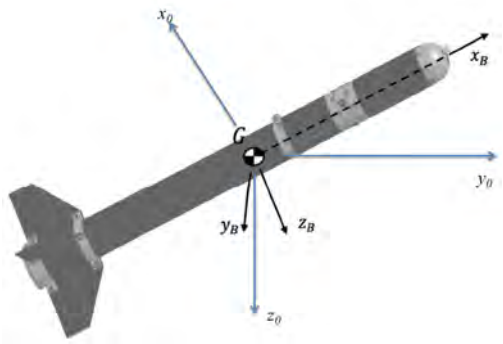


Figure 1. Generic tail-controlled missile with body fixed-frame B and North-East-Down Frame 0

The above described non-minimum phase behaviour is a characteristic of several important flight control problems such as control of Vertical Take-off and Landing (VTOL) aircraft, and Conventional Take-off and Landing (CTOL) aircraft. This behaviour is modeled in the open-loop input-output transfer function as a zero in the right half s-plane. An autopilot design based on cancellation (more commonly known as feedback linearization) for such class of dynamical systems gives an unstable closed-loop as one of the closed-loop poles migrates into the right half-plane.[1] Researchers in the past have mitigated this undesirable behaviour

by slightly modifying the output definition,[2,3] These approaches guarantee 'local' tracking specific to the desired operating condition and reference command as a result of the modification. Other methods[4] proposed modify the sign of some of the control derivatives in the differential equation description of the input-output relationship to render the modified output dynamics minimum phase.

The common control technique used for missiles is the gain-scheduled linear controller,[5,6,7,8] However, dynamics of a missile significantly change during flight and the nonlinearities of the system must be explicitly taken into account during design of the controller for improved maneuverability and stealth. The common nonlinear technique is to employ approximate input-output linearization[9,10] wherein the coupling between the pitching moment and the aerodynamic pitch-deflection surface that causes the non-minimum phase behaviour is ignored during autopilot design. This approximation render the input-output relationship minimum phase but is limited in performance and domain of operation.

This paper revisits the acceleration control problem for a planar aerodynamically-controlled generic missile and the main result is a globally stable nonlinear autopilot design that makes no modification to the output definition and induces a time scale separation in the closed-loop. Unlike previous time scale approaches[11,12,13] and stable-inversion[14] results presented here are based on theoretical advancements in control of nonlinear time scale systems;[15] successful applications of which have been demonstrated on non-minimum phase nap-of-the-earth maneuver for CTOL aircraft[16] and hover control of an autonomous helicopter.[17] The state feedback controller designed here is real-time implementable and independent of any particular operating conditions and desired output trajectory. It is causal and does not require any knowledge or preview of the output trajectory beforehand. This is an important contribution as the proposed nonlinear autopilot can be integrated with an on-board guidance module for real-time operation.

The paper is organized as follows. Section II describes the medium-fidelity generic missile model used for simulation that includes realistic actuators and sensors, and Section III formulates the control problem. The main ideas of the paper and derivation of the autopilot are presented in Section IV. The evaluation of the developed approach in simulation is presented in Section V. Finally, conclusions are presented in Section VI.

II. Simulation Model of the Missile

This section describes the governing equations for a generic missile depicted in Figure 1 along with details of the actuator and sensor subsystems.

A. Rigid-Body Equations of Motion

The dynamics of the three degree-of-freedom generic missile under study are modeled by the following four first-order differential equations[18]

$$\dot{u}_K^G = \frac{1}{m}\left(F_X^G\right)_B - w_K^G q_K^{0B}, \tag{1a}$$

$$\dot{w}_K^G = \frac{1}{m}\left(F_Z^G\right)_B + u_K^G q_K^{0B}, \tag{1b}$$

$$\dot{\theta} = q_K^{0B}, \tag{1c}$$

$$\dot{q}_K^{0B} = \frac{1}{I_{YB}}\left(M_Y^G\right)_B, \tag{1d}$$

written in the body-frame about the center of gravity, G using Newton's laws of motion[19] under the assumption of flat, non-rotating earth and balanced lateral/directional motion with Euler angles ϕ and ψ, roll-rate p, and yaw rate r stabilized about the origin. The reader is referred to the nomenclature for definition of the various symbols in (1).

An equivalent representation for the equations given in (1) can be obtained by noting that the translational velocities (u_K^G, w_K^G) along with the aerodynamic angle α_K^G and the absolute velocity V_K^G satisfy the following

relationships:

$$V_K^G = \sqrt{\left(u_K^G\right)^2 + \left(w_K^G\right)^2}, \tag{2a}$$

$$\alpha_K^G = \arctan\left(\frac{w_K^G}{u_K^G}\right). \tag{2b}$$

$$\tag{2c}$$

Relations given in (2) can be further re-arranged as

$$u_K^G = V_K^G \cos\alpha_K^G, \tag{3a}$$

$$w_K^G = V_K^G \sin\alpha_K^G, \tag{3b}$$

to give expressions for the translational velocities in terms of the angle-of-attack and the total velocity. Using these relations in (1) gives the following equivalent dynamical model representation of the generic missile:

$$\dot{V}_K^G = \frac{1}{m}\left[\cos\alpha_K^G \left(F_X^G\right)_B + \sin\alpha_K^G \left(F_Z^G\right)_B\right], \tag{4a}$$

$$\dot{\alpha}_K^G = q_K^{0B} + \frac{1}{V_K^G}\left[\frac{1}{m}\left\{\cos\alpha_K^G \left(F_Z^G\right)_B - \sin\alpha_K^G \left(F_X^G\right)_B\right\}\right], \tag{4b}$$

$$\dot{\theta} = q_K^{0B}, \tag{4c}$$

$$\dot{q}_K^{0B} = \frac{1}{I_{YB}}\left(M_Y^G\right)_B. \tag{4d}$$

This representation is used in simulation to validate the performance of the autopilot.

The forces $\left(F_X^G\right)_B$, $\left(F_Z^G\right)_B$ and the moment $\left(M_Y^G\right)_B$ in (4) consist of contributions from gravitational, aerodynamic, and propulsive components. Specifically, the aerodynamic forces $(F_A^G)_B$ and the pitching moment $(M_A^G)_B$ with respect to the center of gravity are calculated via the following equations:[18]

$$\begin{bmatrix} (F_{A,X}^G)_B \\ (F_{A,Z}^G)_B \end{bmatrix} = \bar{q} S_{ref} \begin{bmatrix} C_x \\ C_z \end{bmatrix}, \tag{5}$$

$$(M_A^G)_B = \bar{q} S_{ref} l_{ref} C_m, \tag{6}$$

where l_{ref} and S_{ref} denote the reference length and the reference area, respectively and

$$\begin{bmatrix} C_x \\ C_z \end{bmatrix} = \begin{bmatrix} C_{x,0}(\alpha_K^G, \beta_K^G, M) + C_{x,Alt}(\alpha_K^G, (z^G)_I, M) + C_{x,q}(\alpha_K^G, \beta_K^G, M) q_K^{0B} + C_{x,\delta_M}(\alpha_K^G, \beta_K^G, M)\delta_M \\ C_{z,0}(\alpha_K^G, \beta_K^G, M) + C_{z,q}(\alpha_K^G, \beta_K^G, M) q_K^{0B} + C_{z,\delta_M}(\alpha_K^G, \beta_K^G, M)\delta_M \end{bmatrix}, \tag{7}$$

$$C_m = \left[C_{m,0}(\alpha_K^G, \beta_K^G, M) + C_{m,q}(\alpha_K^G, \beta_K^G, M) q_K^{0B} + C_{m,\delta_M}(\alpha_K^G, \beta_K^G, M)\delta_M\right]. \tag{8}$$

The aerodynamic coefficients defined above are nonlinear but smooth functions depending on the angle-of-attack α_K^G, side slip angle β_K^G and Mach number, M. The flight envelope covers the Mach $M = 0.9, \ldots, 4.4$ and altitude region $h = 0, \ldots, 11$km.

B. Modeling of the Actuator and Sensor Subsystem

The missile is controlled via its fin attached at the tail of the missile's body denoted here in the paper as δ_M. The fin is modeled as a second-order linear system (described by damping and a natural frequency) with deflection $\delta_{(limit)}$, deflection rate $\dot{\delta}_{(limit)}$ and deflection acceleration $\ddot{\delta}_{(limit)}$ limits. The fin deflection is assumed to be measurable.

The Inertial Measurement Unit (IMU), located at the missile's c.g., is modeled as a first order element outputting the angular rate q_K^{0B} and the linear accelerations resulting from the aerodynamic forces (5):

$$(a_A^G)_B = \frac{1}{m}(F_A^G)_B. \tag{9}$$

C. Parametric and Estimation Uncertainties

In addition to the outputs obtained from the sensors, the non-measurable states (e.g. aerodynamic angles, Mach number, etc.), the parameters and the aerodynamic data (and their derivative with respect to angle-of-attack) necessary for control design purposes are estimated. The uncertainties associated with these estimates are assumed to time-independent throughout the flight envelope and Table 1 provides an overview of the considered uncertainties.

Table 1. Overview of considered uncertainties

Variable	Uncertainty Range	Units
I_{YB}	$\pm 5\%$	$[-]$
m	$\pm 1\%$	$[-]$
x_{cg}	± 50	$[mm]$
$C_{x,0}$, $C_{z,0}$	$\pm 10\%$	$[-]$
$C_{m,0}$	$\pm 20\%$	$[-]$
$C_{x,\delta_M}, C_{z,\delta_M}, C_{m,\delta_M}$	$\pm 20\%$	$[-]$
$C_{m,q}$	$\pm 20\%$	$[-]$
α_K^G	± 2.5	$[deg]$
M	$\pm 10\%$	$[-]$
$\bar{q}$	$\pm 5\%$	$[-]$

III. Problem Description

The objective of this study is to force the missile to track a desired aerodynamic normal acceleration command denoted as y_r asymptotically. The normal acceleration output is defined as

$$y = \frac{\bar{q}S_{ref}}{m}\left[C_{z,0}(\alpha_K^G, \beta_K^G, M) + C_{z,q}(\alpha_K^G, \beta_K^G, M)q_K^{0B} + C_{z,\delta_M}(\alpha_K^G, \beta_K^G, M)\delta_M\right]. \tag{10}$$

The control problem consists of generating a pitch control deflection δ_M that produces the desired normal acceleration while ensuring complete closed-loop stability. Throughout this study the total velocity V_K^G is assumed to be maintained constant through application of appropriate reaction jets. The autopilot is turned on when the missile is flying at the steady-state horizontal flight condition at $h = 5km$, $\alpha_K^G = 0$deg and $\delta_M = 0$deg. Additionally, the dynamics of the pitch-attitude angle is ignored in the autopilot design as it is related to the pitch rate q_K^{0B} through an exact kinematic relation (see (4)). Consequently the autopilot design for acceleration command tracking function is accomplished through study of the following reduced dynamical model:

$$\dot{\alpha}_K^G = q_K^{0B} + \frac{1}{V_K^G}\left[\frac{1}{m}\left\{\cos\alpha_K^G\left(F_Z^G\right)_B - \sin\alpha_K^G\left(F_X^G\right)_B\right\}\right], \tag{11a}$$

$$\dot{q}_K^{0B} = \frac{1}{I_{YB}}\left(M_Y^G\right)_B, \tag{11b}$$

with output defined in (10).

For convenience in design and analysis the following compact equations for (11) and the output (10) will be used throughout the paper:

$$\dot{\alpha}_K^G = c_0 + c_1 q_K^{0B} + c_2\delta_M, \tag{12}$$

where

$$c_0 = \frac{g}{V_K^G}\cos(\theta - \alpha_K^G) + \frac{\bar{q}S_{ref}}{mV_K^G}\left[b_0\cos\alpha_K^G - a_0\sin\alpha_K^G\right]$$
$$c_1 = 1 + \frac{\bar{q}S_{ref}}{mV_K^G}\left[b_1\cos\alpha_K^G - a_1\sin\alpha_K^G\right]$$
$$c_2 = \frac{\bar{q}S_{ref}}{mV_K^G}\left[b_2\cos\alpha_K^G - a_2\sin\alpha_K^G\right]$$

$$\dot{q}_K^{0B} = c_3 + c_4 q_K^{0B} + c_5\delta_M, \tag{13}$$

where

$$c_3 = \frac{\bar{q}S_{ref}l_{ref}}{I_{YB}}d_0$$
$$c_4 = \frac{\bar{q}S_{ref}l_{ref}}{I_{YB}}d_1$$
$$c_5 = \frac{\bar{q}S_{ref}l_{ref}}{I_{YB}}d_2$$

and

$$y = c_6 + c_7 q_K^{0B} + c_8\delta_M, \tag{14}$$

where

$$c_6 = \frac{\bar{q}S_{ref}}{m}b_0$$
$$c_7 = \frac{\bar{q}S_{ref}}{m}b_1$$
$$c_8 = \frac{\bar{q}S_{ref}}{m}b_2$$

and a_i, b_i, d_i are short-hand definitions for the aerodynamic coefficients defined in the Table 2. This table also gives the order these coefficients take over the range of angle-of-attack and Mach number.

Table 2. Short-hand definitions for aerodynamic coefficients used in control design

Definition	Aerodynamic Coefficient	Order
a_0	$C_{x,0}(\alpha_K^G, \beta_K^G, M) + C_{x,Alt}(\alpha_K^G, (z^G)_I, M)$	$O(0.1)$
a_1	$C_{x,q}(\alpha_K^G, \beta_K^G, M)$	identically 0
a_2	$C_{x,\delta_M}(\alpha_K^G, \beta_K^G, M)$	$O(0.1)$
b_0	$C_{z,0}(\alpha_K^G, \beta_K^G, M)$	$O(10)$
b_1	$C_{z,q}(\alpha_K^G, \beta_K^G, M)$	$O(10)$
b_2	$C_{z,\delta_M}(\alpha_K^G, \beta_K^G, M)$	$O(0.01)$
d_0	$C_{m,0}(\alpha_K^G, \beta_K^G, M)$	$O(10)$
d_1	$C_{m,q}(\alpha_K^G, \beta_K^G, M)$	$O(100)$
d_2	$C_{m,\delta_M}(\alpha_K^G, \beta_K^G, M)$	$O(0.1)$

IV. Autopilot Design

This section details the design procedure of a nonlinear autopilot for the output (14) using the reduced model (12), (13). From the discussion (see Section I) of the non-minimum phase behaviour of a generic missile it is apparent that the pitching motion plays an integral part in ensuring the desired normal acceleration

command is followed. This important role of the angular moment is not accounted in autopilot designs based on exact input-output feedback linearization; hence resulting in undesirable closed-loop missile behaviour. A stable missile autopilot design must ensure that the control deflection being commanded generates commensurable pitch rate through change in aerodynamic pitching moment that will essentially produce the desired normal acceleration. This requirement is posed in this paper in the form of a sequential two-part control problem.

Problem 1: First, given the desired normal acceleration command, determine the sufficient pitch rate required to change the angle-of-attack of the missile appropriately.

Problem 2: Second, using the knowledge of the computed pitch rate, determine the control deflection required to generate the appropriate pitching moment.

Note here that an autopilot designed by solving the above problems can also handle time-varying commands. There is no need for gain scheduling or separate design for different command set-points. These computations can be made in real-time and hence for the same vehicle there is no need for any mission-dependent tuning of feedback gains. Some readers may recognize that the two-part problem posed above is standard in flight control literature for aerial vehicles. But it is important to point out that success of the past studies relied upon the presence of the inherent time scale separation between the angular moment and the translational states. **The main contribution of this paper is formulation of design criteria that guarantee such an autopilot design can be implemented for vehicles that either do not possess a time scale separation or those whose time scale separation changes widely over the flight operation envelope**. In this paper these conditions are developed by identifying criteria that ensure the autopilot design formulated by solving Problem 1 and Problem 2 stabilize the overall missile dynamics. Essential ground work on developing these conditions comes from study of non-standard singularly perturbed systems done by the first author. In this paper the aim is to develop and identify the practical implications of these conditions on the performance and the robustness of a missile autopilot. For more details on the general conditions the reader is referred to 15,17. In the following, subsection A details the design procedure of the autopilot and subsection B develops and analyzes the important design criteria.

A. Control Formulation

In this section control law for the pitch control deflection $\delta_M(\alpha_K^G, q_K^{0B})$ is developed by sequentially addressing Problem 1 and Problem 2. Toward this end, the discussion for solving each of these problems is detailed:

1. *Solution to Problem 1*

The objective here is to determine the sufficient pitch rate required to follow a desired acceleration command. Let us denote this sufficient pitch rate command as q^0 since it is different from the dynamic state q_K^{0B}. Thus, mathematically the objective is to find $q^0(t)$ such that the output defined in (14) satisfies $y(t) \to y_r(t)$ as time $t \to \infty$. In this step we assume that the pitch deflection command is such that the dynamic pitch rate $q_K^{0B}(t)$ is identically equal to $q^0(t)$ (this assumption will be satisfied upon solving Problem 2, and the effect of initial error will be analyzed in subsection B). Under this assumption, the output (14) can be rearranged as

$$y = c_6 + c_7 q^0 + c_8 \delta_M(\alpha_K^G, q^0). \tag{15}$$

From a mathematical standpoint we find that the sufficient pitch rate q^0 can be determined by straightforward algebraic manipulation of (15). But this computation is not useful for control as it does not accommodate for desired time-domain specifications. Hence, we take an additional derivative of (15) to get

$$\dot{y} = \dot{c}_6 + \dot{c}_7 q^0 + \dot{c}_8 \delta_M(\alpha_K^G, q^0) + c_8 \dot{\delta}_M(\alpha_K^G, q^0), \tag{16}$$

by noting that the q^0 acts as a equilibrium for the dynamic pitch rate in this step. From Table 2 we know that the nonlinearities c_i are time-varying due to changes in angle-of-attack (Mach number is assumed to be constant in our study, see first paragraph under section III, and side slip is maintained at zero). This means that $\dot{c}_6 = \frac{\partial c_6}{\partial \alpha_K^G}\dot{\alpha}_K^G$, $\dot{c}_7 = \frac{\partial c_7}{\partial \alpha_K^G}\dot{\alpha}_K^G$, and $\dot{c}_8 = \frac{\partial c_8}{\partial \alpha_K^G}\dot{\alpha}_K^G$. Furthermore we find that $c_8 = \frac{\bar{q}S_{ref}}{m} b_2$ is

comparatively smaller than c_6 and c_7 due to small contribution from the aerodynamic coefficient b_2. Using these approximations (16) becomes

$$\dot{y} = \frac{\partial c_6}{\partial \alpha_K^G}\dot{\alpha}_K^G + \frac{\partial c_7}{\partial \alpha_K^G}\dot{\alpha}_K^G q^0. \tag{17}$$

Analyzing the aerodynamic data from reference 19 we find that $\frac{\partial c_6}{\partial \alpha_K^G}$ is atleast two times greater in magnitude than $\frac{\partial c_7}{\partial \alpha_K^G}$. We make use of this fact to further reduce (17) to

$$\dot{y} = \frac{\partial c_6}{\partial \alpha_K^G}\dot{\alpha}_K^G. \tag{18}$$

Upon expanding out the angle-of-attack dynamics using (12) and the definitions $c_9 \triangleq \frac{\bar{q}S_{ref}}{m}$ and $c_{10} \triangleq \frac{c_9}{V_K^G}a_2 \sin\alpha_K^G$ above we get

$$\dot{y} = \frac{\partial c_6}{\partial \alpha_K^G}\left[c_0 + c_1 q^0 + \left(\frac{b_2 c_9}{V_K^G}\cos\alpha_K^G - c_{10}\right)\delta_M(\alpha_K^G, q^0)\right]. \tag{19}$$

Equation (19) gives the explicit input-output relationship between the output y and the input q^0. Thus, ignoring the small contribution from b_2 and computing q^0 such that

$$\frac{\partial c_6}{\partial \alpha_K^G}\left[c_0 + c_1 q^0 - c_{10}\delta_M(\alpha_K^G, q^0)\right] = \bar{\nu} \tag{20}$$

we find that the closed-loop output dynamics satisfies

$$\dot{y} = \bar{\nu}. \tag{21}$$

The term $\bar{\nu}$ acts as a virtual control input in (21). This input can be designed using any linear control technique to assign desired time-domain specifications to the output. In this paper, we formulate $\bar{\nu}$ as a dynamic controller of the form

$$\bar{\nu} = K_v x \tag{22}$$

where

$$\dot{x} = -K_p(K_v x - \dot{y}_r) - K_I(y - y_r),$$

K_i are positive gains, and K_p and K_I act as proportional and integral gains respectively that can be varied to achieve desired output performance. However, the control designer must be wary of the fact that these gains also influence the nonlinear closed-loop stability of the missile and must be selected by ensuring the criterion derived later in subsection B is also met.

2. Solution to Problem 2

With the desired pitch rate computed using (20), the objective here is to develop a control law for the pitch deflection $\delta_M(\alpha_K^G, q_K^{0B})$ to ensure the dynamic state q_K^{0B} stabilizes about q^0. This can be achieved by noting that the pitch rate dynamics is related to the control deflection δ_M via equation (13). Using feedback linearization then, the control law

$$\delta_M = -\frac{K_q(q_K^{0B} - q^0) + c_3 + c_4 q_K^{0B}}{c_5} \tag{23}$$

will ensure the closed-loop dynamics of the pitch rate becomes

$$\dot{q}_K^{0B} = -K_q(q_K^{0B} - q^0), \tag{24}$$

such that it is uniformly asymptotically stable about the q^0. The constant K_q introduced in (23) above is another positive feedback gain selected by the designer to specify desired time-response properties.

The final step in the control design process is to ensure the control law is specified in terms of quantities that can be either estimated or computed in real-time. For this, rearrange the quantity (20) using the expression $\delta_M(\alpha_K^G, q^0) = -\frac{c_3+c_4q^0}{c_5}$ to get the following algebraic expression

$$\frac{\partial c_6}{\partial \alpha_K^G}\left[c_0 + c_1 q^0 + c_{10}\left(\frac{c_3 + c_4 q^0}{c_5}\right)\right] = \bar{\nu} \tag{25}$$

to compute q^0 as a function of the angle-of-attack in real-time. Hence, (25) together with (22) and (23) describe the autopilot control law.

B. Closed-Loop Analysis

The purpose of this section is to develop criterion under which the control law designed in (23) will asymptotically stabilize the dynamics of the missile, and ensure the output follows the desired command. As mentioned earlier in the paper, this condition must be met when selecting the feedback gains. This analysis is carried out using the Lyapunov's direct method and is detailed in the following five steps.

The first step involves developing the closed-loop dynamics. For this let us rearrange (12) and (14) by expanding c_2 and c_8 to get

$$\dot{\alpha}_K^G = c_0 + c_1 q_K^{0B} + \left[\epsilon \frac{c_9}{V_K^G}\cos\alpha_K^G - c_{10}\right]\delta_M, \tag{26}$$

$$y = c_6 + c_7 q_K^{0B} + \epsilon c_9 \delta_M, \tag{27}$$

using the definitions of c_9 and c_{10} used in (20) and $\epsilon \triangleq b_2$. As we will see later this rearrangement will assist us in studying the effect of ignoring b_2 dependent terms during control design on the closed-loop stability. Next substitute the control law (23) into (26) and (13), and replace the output y in (22) with (27) to get the following closed-loop:

$$\dot{\alpha}_K^G = \frac{K_v x}{\frac{\partial c_6}{\partial \alpha_K^G}} + c_1\left(q_K^{0B} - q^0\right) + K_q \frac{c_{10}}{c_5}(q_K^{0B} - q^0) + \epsilon \frac{c_9}{V_K^G}\cos\alpha_K^G \delta_M, \tag{28a}$$

$$\dot{q}_K^{0B} = -K_q(q_K^{0B} - q^0), \tag{28b}$$

$$\dot{x} = -K_p(K_v x - \dot{y}_r) - K_I\left(c_6 + c_7 q_K^{0B} + \epsilon c_9 \delta_M - y_r\right). \tag{28c}$$

The closed-loop system obtained in (28) is time-varying and has a trim corresponding to different output commands y_r. Since our aim here is to assess stability corresponding to all possible trims, let us translate the trim of the closed-loop system to the origin. For this we develop the general equilibrium equations that are always satisfied in the following second step.

Let us denote (α^*, q^*, x^*) as the trim point. Setting the time derivative of the states α_K^G, q_K^{0B} and x to zero in (28) we get the following three equilibrium equations:

$$q^* = q^0(\alpha^*), \tag{29}$$

$$\frac{K_v x^*}{\frac{\partial c_6}{\partial \alpha_K^G}|_{\alpha^*}} + \epsilon \frac{c_9}{V_K^G}\cos\alpha^* \delta_M(\alpha^*, q^*) = 0, \tag{30}$$

$$-K_p(K_v x^* - \dot{y}_r) - K_I\left(c_6 + c_7 q^* + \epsilon c_9 \delta_M(\alpha^*, q^*) - y_r\right) = 0 \tag{31}$$

where equations (29) through (31) are written about α^*. Notice that trim point for the dynamic pitch rate q^* is exactly equal to q^0 computed about α^*, as desired. Further since (31) is satisfied only if the individual components are zero we get

$$K_v x^* = \dot{y}_r, \tag{32}$$

$$\frac{\partial c_6}{\partial \alpha_K^G}|_{\alpha^*}\epsilon(\alpha^*)\frac{c_9}{V_K^G}\cos\alpha^*\delta_M(\alpha^*, q^*) = -\dot{y}_r, \tag{33}$$

$$c_6(\alpha^*) + c_7(\alpha^*)q^* + \epsilon(\alpha^*)c_9\delta_M(\alpha^*, q^*) = y_r, \tag{34}$$

that define three equations in three unknowns.

The third step in the analysis is to repose the closed-loop system (28) such that the origin becomes the unique equilibrium. This is done by using the equilibrium conditions (32) through (34) in (28)

$$\dot{\alpha}_K^G = \left[\frac{K_v x}{\frac{\partial c_6}{\partial \alpha_K^G}} - \frac{K_v x^*}{\frac{\partial c_6}{\partial \alpha_K^G}|_{\alpha^*}}\right] + \left[c_1 + K_q \frac{c_{10}}{c_5}\right](q_K^{0B} - q^*) + \left[\epsilon \frac{c_9}{V_K^G}\cos\alpha_K^G \delta_M - \epsilon(\alpha^*)\frac{c_9}{V_K^G}\cos\alpha^* \delta_M(\alpha^*, q^*)\right]$$
$$+ \left[c_1 + K_q \frac{c_{10}}{c_5}\right](q^* - q^0(\alpha_K^G)), \tag{35a}$$
$$\dot{q}_K^{0B} = -K_q(q_K^{0B} - q^*) - K_q(q^* - q^0(\alpha_K^G)), \tag{35b}$$
$$\dot{x} = -K_p K_v (x - x^*) - K_I\left[\{c_6 - c_6(\alpha^*)\} + \{c_7 q_K^{0B} - c_7(\alpha^*)q^*\} + \{\epsilon c_9 \delta_M - \epsilon(\alpha^*)c_9\delta_M(\alpha^*, q^*)\}\right]. \tag{35c}$$

In the fourth step select a Lyapunov function candidate $V = \frac{1}{2}(x - x^*)^2 + \frac{1}{2}(\alpha_K^G - \alpha^*)^2 + \frac{1}{2}(q_K^{0B} - q^*)^2$ to study the closed-loop system (35). Taking derivative of this Lyapunov function candidate we get

$$\dot{V} = (1/c_9)(x - x^*)\dot{x} + (\alpha_K^G - \alpha^*)\dot{\alpha}_K^G + (q_K^{0B} - q^*)\dot{q}_K^{0B}. \tag{36}$$

Evaluating (36) along (35)

$$\dot{V} = -K_v K_p (1/c_9)(x - x^*)^2 - K_q(q_K^{0B} - q^*)^2 - K_q(q^* - q^0(\alpha_K^G))(q_K^{0B} - q^*)$$
$$- (1/c_9)K_I\left[\{c_6 - c_6(\alpha^*)\} + \{c_7 q_K^{0B} - c_7(\alpha^*)q^*\} + \{\epsilon c_9\delta_M - \epsilon(\alpha^*)c_9\delta_M(\alpha^*, q^*)\}\right](x - x^*)$$
$$+ \left[\frac{K_v x}{\frac{\partial c_6}{\partial \alpha_K^G}} - \frac{K_v x^*}{\frac{\partial c_6}{\partial \alpha_K^G}|_{\alpha^*}}\right](\alpha_K^G - \alpha^*) + \left[c_1 + K_q\frac{c_{10}}{c_5}\right](\alpha_K^G - \alpha^*)(q_K^{0B} - q^*)$$
$$+ \left[\epsilon\frac{c_9}{V_K^G}\cos\alpha_K^G\delta_M - \epsilon(\alpha^*)\frac{c_9}{V_K^G}\cos\alpha^*\delta_M(\alpha^*, q^*) + \left\{c_1 + K_q\frac{c_{10}}{c_5}\right\}(q^* - q^0(\alpha_K^G))\right](\alpha_K^G - \alpha^*). \tag{37}$$

Note that the difference terms in (37) are a function of the design constants and the aerodynamic coefficients. These terms capture the deviations occurring in the dynamics of the missile due to initial condition errors, and approximations made during control design. This means that the feedback gains must be selected in a way that ensures the derivative of the Lyapunov function candidate is negative definite for a range of parametric uncertainties and reference commands. One way of selecting these gains is by performing Monte Carlo simulations over the range of uncertainties. Instead, in this paper we take the approach of deriving sufficient condition for selecting feedback gains that will ensure stability over a block of uncertainties. This is done in this fifth step by verifying whether the individual difference terms are upper-bounded by some inequality:

1. By the quadratic nature of the term $K_v^2 K_p(x - x^*)^2$ we know that

   $$(1/c_9)K_v K_p(x - x^*)^2 = (1/c_9)K_v K_p|(x - x^*)|^2.$$

2. Similarly

   $$K_q(q_K^{0B} - q^*)^2 = K_q|(q_K^{0B} - q^*)|^2.$$

3. Next consider the difference terms appearing due to $\dot{x}$ dynamics. Using the definitions of c_6, and c_7 from (14) see that

   $$K_I\left[\{c_6 - c_6(\alpha^*)\} + \{c_7 q_K^{0B} - c_7(\alpha^*)q^*\} + \{\epsilon c_9\delta_M - \epsilon(\alpha^*)c_9\delta_M(\alpha^*, q^*)\}\right](x - x^*)$$
   $$= K_I c_9\left[\{b_0 - b_0(\alpha^*)\} + \{b_1 q_K^{0B} - b_1(\alpha^*)q^*\} + \{\epsilon\delta_M - \epsilon(\alpha^*)\delta_M(\alpha^*, q^*)\}\right](x - x^*).$$

   From aerodynamic data we find that the coefficient $b_0(\alpha_K^G) = s_{b0}(\alpha_K^G - \alpha^*)$ is approximately linear in the angle-of-attack with slope $s_{b0} = -|s_{b0}|$. Additionally since the coefficient b_2 is comparatively small we assume that $\epsilon(\alpha_K^G) = \epsilon(\alpha^*) = -|\epsilon|$. Then the above difference term becomes

   $$K_I c_9\left[\{b_0 - b_0(\alpha^*)\} + \{b_1 q_K^{0B} - b_1(\alpha^*)q^*\} + \{\epsilon\delta_M - \epsilon(\alpha^*)\delta_M(\alpha^*, q^*)\}\right](x - x^*)$$
   $$\leq K_I c_9\left[\{s_{b0}(\alpha_K^G - \alpha^*)\} + \{b_1 q_K^{0B} - b_1(\alpha^*)q^*\} + \epsilon\{\delta_M - \delta_M(\alpha^*, q^*)\}\right](x - x^*).$$

Using the control law from (23) the above inequality can be further rearranged to

$$K_I c_9 \left[\left\{s_{b0}(\alpha_K^G - \alpha^*)\right\} + \left\{b_1 q_K^{0B} - b_1(\alpha^*)q^*\right\} + \epsilon\left\{\delta_M - \delta_M(\alpha^*, q^*)\right\}\right](x - x^*)$$
$$\leq K_I c_9 \left[\left\{s_{b0}(\alpha_K^G - \alpha^*)\right\} + \left\{b_1 q_K^{0B} - b_1(\alpha^*)q^*\right\} + \frac{\epsilon}{c_5}\left\{-K_q(q_K^{0B} - q^*)\right\}\right](x - x^*)$$
$$\leq -K_I c_9 |s_{b0}||(\alpha_K^G - \alpha^*)||(x - x^*)| - K_I K_q c_9 \left|\frac{\epsilon}{c_5}\right| |(q_K^{0B} - q^*)||(x - x^*)|$$
$$+ K_I c_9 |b_1 q_K^{0B} - b_1(\alpha^*)q^*||(x - x^*)|$$

since $c_5 = -|c_5|$ due to the nature of aerodynamic coefficient d_2. Using the triangle inequality $|b_1 q_K^{0B} - b_1(\alpha^*)q^*| \leq |b_1(\alpha_K^G)\left\{q_K^{0B} - q^*\right\}| + |b_1(\alpha_K^G) - b_1(\alpha^*)||q^*|$ and the fact from the aerodynamic data $|b_1(\alpha_K^G) - b_1(\alpha^*)| \leq -\beta|\alpha_K^G - \alpha^*|$ where $\beta > 0$ for all angle-of-attack and Mach number values, then

$$K_I \left[\{c_6 - c_6(\alpha^*)\} + \left\{c_7 q_K^{0B} - c_7(\alpha^*)q^*\right\} + \{\epsilon c_9 \delta_M - \epsilon(\alpha^*)c_9\delta_M(\alpha^*, q^*)\}\right](x - x^*)$$
$$\leq -K_I c_9 |s_{b0}||(\alpha_K^G - \alpha^*)||(x - x^*)| - K_I K_q c_9 \left|\frac{\epsilon}{c_5}\right| |(q_K^{0B} - q^*)||(x - x^*)|$$
$$+ K_I c_9 |b_1(\alpha_K^G)\left\{q_K^{0B} - q^*\right\}||(x - x^*)| - K_I c_9 \beta |q^*||\alpha_K^G - \alpha^*||(x - x^*)|.$$

4. The error due to arbitrary initial condition of the integrator state x is expressed as

$$\left[\frac{K_v x}{\frac{\partial c_6}{\partial \alpha_K^G}} - \frac{K_v x^*}{\frac{\partial c_6}{\partial \alpha_K^G}|_{\alpha^*}}\right](\alpha_K^G - \alpha^*) \leq -\left|\frac{K_v}{c_9 s_{b0}}\right| |(x - x^*)||(\alpha_K^G - \alpha^*)|$$

by using the definition of the aerodynamic coefficient c_6 and linear dependence of b_0.

5. This next term appears due to coupling between the translational and rotational motion components:

$$\left[c_1 + K_q \frac{c_{10}}{c_5}\right](\alpha_K^G - \alpha^*)\left(q_K^{0B} - q^*\right) \leq -|c_1||(\alpha_K^G - \alpha^*)|| \left(q_K^{0B} - q^*\right)| - K_q \left|\frac{c_{10}}{c_5}\right| |(\alpha_K^G - \alpha^*)|| \left(q_K^{0B} - q^*\right)|$$

since c_1 and c_5 are always negative.

6. The effect of ignoring the small quantity ϵ in the autopilot design is captured by the following difference term:

$$\left[\epsilon \frac{c_9}{V_K^G}\cos\alpha_K^G \delta_M - \epsilon(\alpha^*)\frac{c_9}{V_K^G}\cos\alpha^* \delta_M(\alpha^*, q^*)\right](\alpha_K^G - \alpha^*)$$
$$\leq \epsilon \frac{c_9}{V_K^G}\left[-\frac{K_q}{c_5}\cos\alpha_K^G(q_K^{0B} - q^*) + \delta_M(\alpha^*, q^*)\left\{\cos\alpha_K^G - \cos\alpha^*\right\}\right](\alpha_K^G - \alpha^*)$$
$$\leq \left|\epsilon \frac{c_9}{V_K^G}\frac{K_q}{c_5}\cos\alpha_K^G\right| |q_K^{0B} - q^*||(\alpha_K^G - \alpha^*)| - \left|\epsilon \frac{c_9}{V_K^G}\delta_M(\alpha^*, q^*)\right| |(\alpha_K^G - \alpha^*)|^2$$

where we have used the control law (23) and the assumption that the ϵ variation with angle-of-attack is negligible.

7. Finally the difference in the pitch rate trim point q^* and the input q^0 is a function of the difference angle-of-attack and its trim point from (25). That is

$$q^* - q^0(\alpha_K^G) = \frac{1}{c_1(\alpha^*) + c_{10}\frac{c_4(\alpha^*)}{c_5(\alpha^*)}}\left[\frac{K_v x^*}{\frac{\partial c_6}{\partial \alpha_K^G}|_{\alpha^*}}\right] - \frac{1}{c_1(\alpha_K^G) + c_{10}\frac{c_4(\alpha_K^G)}{c_5(\alpha_K^G)}}\left[\frac{K_v x}{\frac{\partial c_6}{\partial \alpha_K^G}}\right]$$
$$- \frac{1}{c_1(\alpha^*) + c_{10}\frac{c_4(\alpha^*)}{c_5(\alpha^*)}}\left\{c_0(\alpha^*) + c_{10}(\alpha^*)\frac{c_3(\alpha^*)}{c_5(\alpha^*)}\right\} + \frac{1}{c_1(\alpha_K^G) + c_{10}\frac{c_4(\alpha_K^G)}{c_5(\alpha_K^G)}}\left\{c_0(\alpha_K^G) + c_{10}(\alpha_K^G)\frac{c_3(\alpha_K^G)}{c_5(\alpha_K^G)}\right\}.$$

Using properties 4 and 5 given above,

$$\frac{1}{c_1(\alpha^*)+c_{10}\frac{c_4(\alpha^*)}{c_5(\alpha^*)}}\left[\frac{K_v x^*}{\frac{\partial c_6}{\partial \alpha_K^G}|_{\alpha^*}}\right]-\frac{1}{c_1(\alpha_K^G)+c_{10}\frac{c_4(\alpha_K^G)}{c_5(\alpha_K^G)}}\left[\frac{K_v x}{\frac{\partial c_6}{\partial \alpha_K^G}}\right]\leq\frac{1}{\left|c_1+c_{10}\frac{c_4}{c_5}\right|}\left|\frac{K_v}{c_9 s_{b0}}\right||(x-x^*)|,$$

$$\frac{1}{c_1(\alpha^*)+c_{10}\frac{c_4(\alpha^*)}{c_5(\alpha^*)}}\left\{c_0(\alpha_K^G)-\ c_0(\alpha^*)\right\}\leq\frac{1}{\left|c_1+c_{10}\frac{c_4}{c_5}\right|}\left|\frac{c_9}{V}\right|\left\{-|a_0||\alpha_K^G-\alpha^*|-|s_{b0}||\alpha_K^G-\alpha^*|+|b_0||\alpha_K^G-\alpha^*|\right\},$$

and

$$\frac{1}{c_1(\alpha^*)+c_{10}\frac{c_4(\alpha^*)}{c_5(\alpha^*)}}\left\{c_{10}(\alpha_K^G)\frac{c_3(\alpha_K^G)}{c_5(\alpha_K^G)}-c_{10}(\alpha^*)\frac{c_3(\alpha^*)}{c_5(\alpha^*)}\right\}\ \leq\frac{1}{\left|c_1+c_{10}\frac{c_4}{c_5}\right|}\left|\frac{c_{10}c_9 m l_{ref}}{c_5 I_{YB}}\right|\left\{|s_{d0}||\alpha_K^G-\alpha^*|\right\}$$

hold using the definition of c_3 and the fact that $d_0=-|s_{d0}|(\alpha_K^G-\alpha^*)$. Then we have that

$$q^*-q^0(\alpha_K^G)\leq\frac{1}{\left|c_1+c_{10}\frac{c_4}{c_5}\right|}\left[\left|\frac{K_v}{c_9 s_{b0}}\right||(x-x^*)|+\left(\left|\frac{c_9}{V}\right|\{-|a_0|-|s_{b0}|+|b_0|\}+\left|\frac{c_{10}c_9 m l_{ref}s_{d0}}{c_5 I_{YB}}\right|\right)|\alpha_K^G-\alpha^*|\right]$$

Substituting the above computed inequalities into (37) we get

$$\begin{aligned}\dot{V}=&-(1/c_9)K_v K_p|(x-x^*)|^2-K_q(q_K^{0B}-q^*)^2\\&+\left[K_I|s_{b0}|+K_I\beta|q^*|-\frac{K_v}{c_9|s_{b0}|}-\left\{|c_1|+\left|\frac{K_q c_{10}}{c_5}\right|\right\}\frac{1}{\left|c_1+c_{10}\frac{c_4}{c_5}\right|}\left|\frac{K_v}{c_9 s_{b0}}\right|\right]|\alpha_K^G-\alpha^*||x-x^*|\\&\left[-|c_1|-K_q|\frac{c_{10}}{c_5}|+\left|\epsilon\frac{K_q}{V_K^G}\frac{c_9}{c_5}\cos\alpha_K^G\right|-\frac{K_q}{\left|c_1+c_{10}\frac{c_4}{c_5}\right|}\left(\left|\frac{c_9}{V}\right|\{-|a_0|-|s_{b0}|+|b_0|\}+\left|\frac{c_{10}c_9 m l_{ref}s_{d0}}{c_5 I_{YB}}\right|\right)\right]|\alpha_K^G-\alpha^*||q_K^{0B}-q^*|\\&\left[K_I K_q|\frac{\epsilon}{c_5}|-K_I|b_1(\alpha_K^G)|-\frac{K_q}{\left|c_1+c_{10}\frac{c_4}{c_5}\right|}\left|\frac{K_v}{c_9 s_{b0}}\right|\right]|x-x^*||q_K^{0B}-q^*|\\&-\left[\left|\epsilon\frac{c_9}{V_K^G}\delta_M(\alpha^*,q^*)\right|+\left\{|c_1|+\left|\frac{K_q c_{10}}{c_5}\right|\right\}\frac{1}{\left|c_1+c_{10}\frac{c_4}{c_5}\right|}\left(\left|\frac{c_9}{V}\right|\{-|a_0|-|s_{b0}|+|b_0|\}+\left|\frac{c_{10}c_9 m l_{ref}s_{d0}}{c_5 I_{YB}}\right|\right)\right]|(\alpha_K^G-\alpha^*)|^2,\end{aligned}\tag{38}$$

which can be rearranged to

$$\dot{V}\leq[(x-x^*)\quad(\alpha_K^G-\alpha^*)\quad(q_K^{0B}-q^*)]\mathbb{K}[(x-x^*)\quad(\alpha_K^G-\alpha^*)\quad(q_K^{0B}-q^*)]^T\tag{39}$$

with matrix $\mathbb{K}$ defined as

$$\mathbb{K}=\begin{bmatrix}-K_v K_p(1/c_9)&\mu_1&\mu_2\\\mu_1&-\mu_\alpha&\mu_3\\\mu_2&\mu_3&-K_q\end{bmatrix}\tag{40}$$

with μ_i defined in Table 3. Then from Lyapunov's method it is clear that the missile will have stable closed-loop dynamics if the matrix $\mathbb{K}$ is negative-definite. Furthermore, if the matrix $\mathbb{K}$ is negative definite, the states (x,α_K^G,q_K^{0B}) will asymptotically approach their trim points defined in (32) through (34), which will in turn mean that the output will asymptotically approach the desired reference command y_r.

The above analysis indicates that the feedback gains must be chosen to ensure the matrix $\mathbb{K}$ is negative-definite for a range of parametric uncertainties. This is a sufficiency condition and can be met by following the two steps. First, to ensure negative definiteness for a range of uncertainties the designer must use the least upper bound values for the aerodynamic coefficients and the flight conditions in

Table 3. Definition of elements of $\mathbb{K}$

Parameter	Definition
μ_α	$\left[\left\|\epsilon\frac{c_9}{V_K^G}\delta_M(\alpha^*,q^*)\right\| + \left\{\|c_1\| + \left\|\frac{K_q c_{10}}{c_5}\right\|\right\}\frac{1}{\left\|c_1+c_{10}\frac{c_4}{c_5}\right\|}\left(\left\|\frac{c_9}{V}\right\|\{-\|a_0\| - \|s_{b0}\| + \|b_0\|\} + \left\|\frac{c_{10}c_9 m l_{ref} s_{d0}}{c_5 I_{YB}}\right\|\right)\right]$
μ_1	$\frac{1}{2}\left[K_I\|s_{b0}\| + K_I\beta\|q^*\| - \frac{K_v}{c_9\|s_{b0}\|} - \left\{\|c_1\| + \left\|\frac{K_q c_{10}}{c_5}\right\|\right\}\frac{1}{\left\|c_1+c_{10}\frac{c_4}{c_5}\right\|}\left\|\frac{K_v}{c_9 s_{b0}}\right\|\right]$
μ_2	$\frac{1}{2}\left[K_I K_q\|\frac{\epsilon}{c_5}\| - K_I\|b_1(\alpha_K^G)\| - \frac{K_q}{\left\|c_1+c_{10}\frac{c_4}{c_5}\right\|}\left\|\frac{K_v}{c_9 s_{b0}}\right\|\right]$
μ_3	$\frac{1}{2}\left[-\|c_1\| - K_q\|\frac{c_{10}}{c_5}\| + \left\|\epsilon\frac{K_q}{V_K^G}\frac{c_9}{c_5}\cos\alpha_K^G\right\| - \frac{K_q}{\left\|c_1+c_{10}\frac{c_4}{c_5}\right\|}\left(\left\|\frac{c_9}{V}\right\|\{-\|a_0\| - \|s_{b0}\| + \|b_0\|\} + \left\|\frac{c_{10}c_9 m l_{ref} s_{d0}}{c_5 I_{YB}}\right\|\right)\right]$

μ_i. This will ensure that gains computed in the next step will guarantee stability for a block of parametric values. At the end of the first step the matrix $\mathbb{K}$ will become purely a function of the feedback gains, selection of which requires some iteration. In the second step the designer must make an initial guess for K_v, K_p and K_I. This can be done by noticing that the output response transfer function is

$$\frac{y(s)}{y_r(s)} = \frac{K_v[K_p s + K_I]}{s^2 + K_v K_p s + K_v K_I} \tag{41}$$

at the end of Problem 1. This means that appropriate selection of K_v, K_p and K_I will help in assigning desired time-domain characteristics to the output. On the other hand feedback gain K_q assigns speed of response to the pitch rate dynamics and must be chosen to be of higher order than K_v, K_p and K_I. Using these values as the initial guess into matrix $\mathbb{K}$ the designer must iterate a few more times to get desired negative definiteness. These iterations may be required for two reasons: one for ensuring the stability of the nonlinear closed-loop system and the other to ensure the closed-loop has desired properties. This is an important feature about the construction of matrix $\mathbb{K}$. Notice that due to choice of the Lyapunov function, the eigenvalues of matrix $\mathbb{K}$ directly correspond to the response of the states of the closed-loop system and can be modified as desired.

Finally, before leaving this section let us expand further on the choice of the feedback gain K_q. Looking back at the closed-loop (35) notice that if K_q is chosen to assign a small time constant to the angular rate dynamics, then after some finite time t^* which is greater than initial time the dynamics (35) will reduce to:

$$q_K^{0B}(t > t^*) = q^0(\alpha_K^G), \tag{42a}$$

$$\dot{\alpha}_K^G = \left[\frac{K_v x}{\frac{\partial c_6}{\partial \alpha_K^G}} - \frac{K_v x^*}{\frac{\partial c_6}{\partial \alpha_K^G}|_{\alpha^*}}\right] + \left[\epsilon\frac{c_9}{V_K^G}\cos\alpha_K^G\delta_M(\alpha_K^G, q^0) - \epsilon(\alpha^*)\frac{c_9}{V_K^G}\cos\alpha^*\delta_M(\alpha^*, q^*)\right], \tag{42b}$$

$$\dot{x} = -K_p K_v(x - x^*) - K_I\left[\{c_6 - c_6(\alpha^*)\} + \{c_7 q^0 - c_7(\alpha^*)q^*\} + \{\epsilon c_9\delta_M(\alpha_K^G, q^0) - \epsilon(\alpha^*)c_9\delta_M(\alpha^*, q^*)\}\right]. \tag{42c}$$

Use the definition of the output y from (27), and equilibrium relation (34), to further reduce (42)

$$q_K^{0B}(t > t^*) = q^0(\alpha_K^G), \tag{43a}$$

$$\dot{\alpha}_K^G = \left[\frac{K_v x}{\frac{\partial c_6}{\partial \alpha_K^G}} - \frac{K_v x^*}{\frac{\partial c_6}{\partial \alpha_K^G}|_{\alpha^*}}\right] + \left[\epsilon\frac{c_9}{V_K^G}\cos\alpha_K^G\delta_M(\alpha_K^G, q^0) - \epsilon(\alpha^*)\frac{c_9}{V_K^G}\cos\alpha^*\delta_M(\alpha^*, q^*)\right], \tag{43b}$$

$$\dot{x} = -K_p(K_v x - \dot{y}_r) - K_I\left[y(\alpha_K^G, q^0) - y_r\right], \tag{43c}$$

or

$$q_K^{0B}(t > t^*) = q^0(\alpha_K^G), \tag{44a}$$

$$\ddot{x} = -K_p(K_v\dot{x} - \ddot{y}_r) - K_I\left[\frac{\partial y}{\partial \alpha_K^G}\left[\frac{K_v x}{\frac{\partial c_6}{\partial \alpha_K^G}} - \frac{K_v x^*}{\frac{\partial c_6}{\partial \alpha_K^G}|_{\alpha^*}}\right] - \dot{y}_r\right] \tag{44b}$$

$$- K_I\frac{\partial y}{\partial \alpha_K^G}\left[\epsilon\frac{c_9}{V_K^G}\cos\alpha_K^G\delta_M(\alpha_K^G, q^0) - \epsilon(\alpha^*)\frac{c_9}{V_K^G}\cos\alpha^*\delta_M(\alpha^*, q^*)\right].$$

Notice (44b) is composed of three terms. The first two terms appear due to arbitrary initial conditions, and from design of virtual input $\bar{\nu}$ (22) these terms will die out in time. The third term captures the effect of ignoring b_2 in the control design, and acts as an external disturbance. Due to the integral action of the virtual input $\bar{\nu}$ qualitatively we may say that the effect of this disturbance on the steady state response of the output will be zero. However, notice this conclusion was based on the assumption that the pitch rate dynamics settles down faster than the other states. **This means that essentially the choice of feedback gains must be made such that a time-scale separation is induced in the closed-loop system**. Furthermore, (44) also suggests that this induced time scale separation does not depend on whether or not the missile has inherently time scale separated dynamics.

V. Case Study

The purpose of this section is to demonstrate the controller performance in simulation for a generic missile model presented in (4), and illustrate how matrix $\mathbb{K}$ can be employed in selection of feedback gains. The operating point $V_K^G = 600$m/sec and $h = 5$km is chosen as initial condition for the simulation. The control command consists of step inputs applied to the acceleration channel with an amplitude of $10g$. This step input command is passed through a reference model to create a ramp-like reference trajectory for the autopilot to follow. With the chosen maneuver, the overall acceleration commanded to the missile is close to the maximum trimmable acceleration at the considered envelope point.

A. Feedback gain selection

The feedback gains are selected by carrying out the two steps listed in analysis section above:

1. In order to ensure stability for the range of values given in Table 1 the least-upper bounds (or greatest lower bounds for denominators) for the aerodynamic data terms are determined. These are tabulated in Table 4 for the missile model under study.

2. At the end of the first step, matrix $\mathbb{K}$ depends purely on the feedback gains. In this second step these gains need to be varied to ensure the matrix attains negative definiteness properties. This will require iterations and one may begin by noting that K_v acts as a multiplying factor in (41) and can be fixed to one initially. Furthermore, if there are no uncertainties in the system then integral action can be temporarily turned off and thus, $K_I = 0$. Then only K_p and K_q need to be varied to ensure the matrix $\mathbb{K}$ is stable. But due to integral action of the state x we find that this matrix is only negative semi-definite for the missile. That is $\dot{V} \leq 0$. Then since the Lyapunov function candidate is lower bounded we have that the error states $(x - x^*)$, $(\alpha_K^G - \alpha^*)$ and $(q_K^{0B} - q^*)$ are bounded. Using this fact and the Barbalat's lemma[20] one can show that $\dot{V} \to 0$ as $t \to \infty$, and hence the errors $(x - x^*) \to 0$, $(\alpha_K^G - \alpha^*) \to 0$ and $(q_K^{0B} - q^*) \to 0$. This proves that the errors converge to zero and the output tracks the reference command asymptotically. For the simulation results presented next $K_y = 8$, $K_I = 2$ and $K_q = 10$ were selected. With these values the eigenvalues of the matrix $\mathbb{K}$ were computed as $\lambda_1 = -0.22$, $\lambda_2 = -0.16$ and $\lambda_3 = 0$.

B. Simulation results

Figure 2 presents the closed-loop results for the generic missile being commanded to acceleration of 10g. Figure 2 indicates that the autopilot demonstrated asymptotic tracking irrespective of the commanded acceleration. The commanded control deflection is consistent with the dynamics of the vehicle described in Section I. Between $5-10$seconds negative pitch-deflection commands are generated to induce a positive change in angle-of-attack and accelerate the missile upward as desired. This negative control deflection command is corrected at 10seconds when the commanded acceleration is brought back to $0g$. Similar behaviour is seen between $15 - 20$ seconds with positive control deflection command inducing a negative angle-of-attack decelerating the missile to $-10g$. At 20seconds the pitch-deflection command starts to go back to $0deg$ and around 22.5 seconds the missile stabilizes about origin.

The inherent tendency of the missile to resist change is evident each time the acceleration command is altered. Initially between $0 - 2.5$ seconds the transient changes in the normal acceleration are corrected through small control deflections. Furthermore at 5seconds, even though a negative deflection is generated,

Table 4. Values used in computation of μ_i(given in appropriate units)

Parameter	Value
$\lvert\epsilon\rvert$	0.0027
$\lvert c_9\rvert$	49.9043
β	9.5077
$\lvert s_{b0}\rvert$	33.07
$\lvert s_{d0}\rvert$	74.28
$\lvert b_0\rvert$	43.75
$\lvert a_0\rvert$	0.2512
c_5	0.176
$\lvert c_1\rvert$	0.2428
$\lvert c_{10}\rvert$	0.0696
$\lvert q*\rvert$	π
$\lvert\delta_M(\alpha^*, q^*)\rvert$	$45(180/\pi)$

the missile starts to accelerate in the wrong direction until sufficient change in angle-of-attack is created. Similar behaviour is seen at 10, 15,and 20 seconds respectively. Despite this non-minimum phase behaviour the autopilot maintained asymptotic tracking throughout the simulation.

VI. Conclusions

In this paper synthesis of a nonlinear autopilot was presented for a generic aerodynamically controlled missile. The non-minimum phase characteristics of the missile were addressed by inducing a time scale separation in the closed-loop to ensure the rotational dynamics remain stable at all times. The designed autopilot exhibits almost linear time-invariant closed-loop dynamics as a result of which the designed missile control system successfully fulfilled the demanding maneuver. Based on the results presented in the paper, the following three conclusions are drawn. First, the final output tracking rise-time of the step response is 1.5seconds and the settling time is around 4seconds. This close output tracking was a result of computing the desired pitch rate in real-time. Second, the autopilot is independent of the commanded reference and globally applicable. Third, unlike other exact output tracking approaches for non-minimum phase systems, the autopilot is causal and does not require any prior information or preview of the desired reference.

References

[1]Isidori, A., *Nonlinear Control Systems*, Springer-Verlag, London, 3rd ed., 1995, ISBN 3-540-19916-0.

[2]Benvenuti, L., Benedetto, M. D. D., and Grizzle, J. W., "Approximate output tracking for nonlinear nonminimum phase systems with an application to flight control," *Journal of Nonlinear Robust Control*, Vol. 4, 1994, pp. 397–414.

[3]Hedrick, J. and Gopalswamy, S., "Nonlinear Flight Control Design via Sliding Manifolds," *Journal of Guidance*, Vol. 13, No. 5, 1990, pp. 850–858.

[4]Doyle, F. J., Allgöwer, F., and Morari, M., "A normal form approach to approximate input-output linearization for maximum phase nonlinear SISO systems," *IEEE Transactions on Automatic Control*, Vol. 41, No. 2, 1996, pp. 305–309.

[5]Wise, K. and Broy, D. J., "Agile missle dynamics and control," *Journal of Guidance, Control, and Dynamics*, Vol. 21, 1998, pp. 441–449.

[6]Shamma, J. and Cloutier, J. R., "Gain-scheduled missile autopilot design using linear parameter varying transformation," *Journal of Guidance, Control and Dynamics*, Vol. 16, No. 2, 1993, pp. 256–263.

[7]Wu, F., Packard, A., and Balase, G., "LPV Control design for pitch axis missile autopilots," *Proceedings of the 34th IEEE control decision conference*, New Orleans, LA, 1995, pp. 188–191.

[8]Ryu, S.-M., Won, D.-Y., Lee, C.-H., and Tahk, M.-J., "Missile Autopilot Design for Agile Turn Control During Boost-Phase," *International Journal of Aeronautical & Space Sciences*, Vol. 12, No. 4, 2011, pp. 365–370.

[9]Hauser, J., Sastry, S. S., and Meyer, G., "Nonlinear control design for slightly nonminimum phase systems: Application to V/STOL aircraft," *Automatica*, Vol. 28, No. 4, 1992, pp. 665–679.

[10]Devaud, E., Siguerdidjane, H., and Font, S., "Some control strategies for a high-angle-of-attack missile autopilot," *Control Engineering Practice*, Vol. 8, 2000, pp. 885–892.

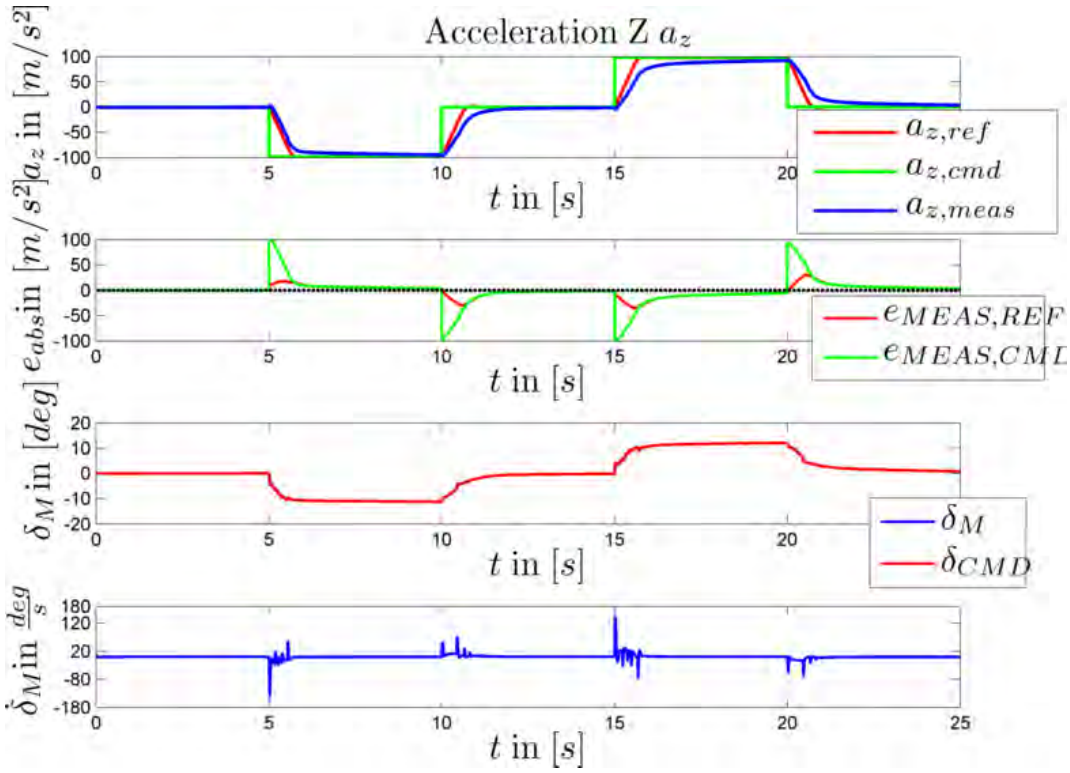


Figure 2. Tracking performance of a generic missile with proposed autopilot design at $V_K^G = 600m/sec$ and $h = 5km$

[11]Lee, J. and Ha, I., "Autopilot design for highly maneuvering STT missiles via singular perturbation-like technique," *IEEE Transactions on Control System Technology*, Vol. 7, No. 5, 1999, pp. 527–541.

[12]Lee, J.-I. and Ha, I.-J., "A Novel Approach to Control of Nonminimum-Phase Nonlinear Systems," *IEEE Transactions on Automatic Control*, Vol. 47, No. 9, 2002, pp. 1480–1486.

[13]Menon, P. K. and M.Yousefpor, "Design of nonlinear autopilots for high angle of attack missiles," *AIAA Guidance, Navigation and, Control conference*, 1996, No. 96-3913.

[14]Devasia, S., Chen, D., and Paden, B., "Nonlinear inversion-based output tracking," *IEEE Transactions on Automatic Control*, Vol. 41, No. 7, 1996, pp. 930–942.

[15]Siddarth, A. and Valasek, J., "Tracking Control Design for Non-Standard Nonlinear Singularly Perturbed Systems," *Proceedings of American Control Conference*, Montreal, Canada, June 2012.

[16]Siddarth, A. and Valasek, J., "Output Tracking of Non-Minimum Phase Dynamics," *AIAA Guidance, Navigation and Control Conference*, Portland, Oregon, August 2011, AIAA-2011-6487.

[17]Siddarth, A. and Valasek, J., "Tracking Control for a Non-Minimum Phase Autonomous Helicopter," *AIAA Guidance, Navigation and Control Conference*, Minneapolis, Minnesota, August 2012, AIAA 2012-4453.

[18]Florian Peter, Miguel Leito, F. H., "Adaptive Augmentation of a New Baseline Control Architecture for Tail-Controlled Missiles Using a Nonlinear Reference Model," *AIAA Guidance, Navigation, and Control Conference*, Minneapolis, Minnesota, 13 - 16 August 2012, AIAA 2012-5037.

[19]Lange, R., *Nonlinear adaptive control of an endo-atmospheric dual-actuator interceptor*, Master's thesis, Technical University of Munich, November 2011.

[20]Khalil, H. K., *Nonlinear Systems*, Prentice Hall, Upper Saddle River, NJ, 3rd ed., December 2001.

TECHNICAL SUMMARIES – SUMAN CHAKRAVORTY

In this project, we have developed sampling based feedback planning techniques for the solution of Markov Decision Problems (MDP) and Partially Observed MDPs (POMDP). The primary application of interest has been the feedback motion planning for unmanned robotic vehicles, both ground and air based. The techniques have been implemented on real hardware and shown to be robust and real time implementable. We have also extended these techniques to the case of multi-vehicle systems. The following papers have resulted form this work. Attached to this document are also four papers that best summarize the contributions in this work.

1. Agha-mohammadi, Suman Chakravorty, Nancy Amato, "FIRM: Sampling-based Feedback Motion Planning Under Motion Uncertainty and Imperfect Measurements", *International Journal of Robotics Research*, 33(2):268-304, February 2014

2. Agha-mohammadi, Saurav Agarwal, Aditya Mahadevan, Suman Chakravorty, Daniel Tomkins, Jory Denny, Nancy Amato, "Robust Online Belief Space Planning in Changing Environments: Application to Physical Mobile Robots," In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Hong Kong, China, May 2014

3. Agha-mohammadi, Saurav Agarwal, Suman Chakravorty, " Periodic-node Sampling-Based Framework for Stochastic Motion Control of Small Aerial Vehicles," *The ASME Journal of Dynamic Systems, Measurement and Control*, Special Issue on Stochastic Models, Control and Algorithms in Robotics [to appear]

4. Agha-mohammadi, Suman Chakravorty, Nancy Amato, "Sampling-based Stochastic Control with Constraints: A Unified Approach in State and Information-state Spaces", in *American Control Conference (ACC)*, invited paper, Stochastic Models, Control and Algorithms in Robotics session, Washington, DC, June 2013

5. Aghamohammadi, S. Chakravorty and N. M. Amato, "Sampling-based Nonholonomic Motion Planning in Belief Space via Dynamic Feedback Linearization-based FIRM", *Proc. 2012 IEEE/ RSJ Int. Conf. on Robotics and Intelligent Systems (IROS)*, Vilamoura, Algarve, Portugal

6. S. Kumar and S. Chakravorty, "Multi-Agent Generalized Probabilistic Roadmaps", *Proc. 2012 IEEE/ RSJ Int. Conf. on on Rob. Intell. Syst. (IROS)*, Vilamoura, Algarve, Portugal

7. Aghamohammadi, S. Chakravorty and N. M. Amato, "Probabilistic Completeness of Feedback Aware Information Roadmaps", *Proc. 2012 IEEE Int. Conf. on Robotics and Automation (ICRA)*, Minneapolis, MN

8. Aghamohammadi, S. Chakravorty and N. M. Amato "Feedback Aware Information Roadmaps", *Proc. 2011 IEEE/ RSJ Int. Conf. on Robotic and Intelligent Systems (IROS)*, San Francisco, CA

The papers attached are 1, 2, 3 and 6 which encapsulate the key contributions of this work. A brief description of these papers is given below.

Paper 1. In this paper we present feedback-based information roadmap (FIRM), a multiquery approach for planning under uncertainty which is a belief-space variant of probabilistic roadmap methods. The crucial feature of FIRM is that the costs associated with the edges are independent of each other, and in this sense it is the first method that generates a graph in belief space that preserves the optimal substructure property. From a practical point of view, FIRM is a robust and reliable planning framework. It is robust since the solution is a feedback and there is no need for expensive replanning. It is reliable because accurate collision probabilities can be computed along the edges. In addition, FIRM is a scalable framework, where the complexity of planning with FIRM is a constant multiplier of the complexity of planning with PRM. In this paper, FIRM is introduced as an abstract framework. As a concrete instantiation of FIRM, we adopt stationary linear quadratic Gaussian (SLQG) controllers as belief stabilizers and introduce the so-called SLQG-FIRM. In SLQG-FIRM we focus on kinematic systems and then extend to dynamical systems by sampling in the equilibrium space. We investigate the performance of SLQG-FIRM in different scenarios.

Paper 2. Motion planning in belief space (under motion and sensing uncertainty) is a challenging problem due to the computational intractability of its exact solution. The Feedback-based Information RoadMap (FIRM) framework made an important theoretical step toward enabling roadmap-based planning in belief space and provided a computationally tractable version of belief space planning. However, there are still challenges in applying belief space planners to physical systems, such as the discrepancy between computational models and real physical models. In this paper, we propose a dynamic replanning scheme in belief space to address such challenges. Moreover, we present techniques to cope with changes in the environment (e.g., changes in the obstacle map), as well as unforeseen large deviations in the robot's location (e.g., the kidnapped robot problem). We then utilize these techniques to implement the first online replanning scheme in belief space on a physical mobile robot that is robust to changes in the environment and large disturbances. This method demonstrates that belief space planning is a practical tool for robot motion planning.

Paper 3. This paper presents a strategy for stochastic control of small aerial vehicles under uncertainty using graph-based methods. In planning with graph-based methods, such as the Probabilistic Roadmap Method (PRM) in state space or the Information RoadMaps (IRM) in information-state (belief) space, the local planners (along the edges) are responsible to drive the state/belief to the final node of the edge. However, for aerial vehicles with minimum velocity constraints, driving the system belief to a sampled belief is a challenge. In this paper, we propose a novel method based on periodic controllers, in which instead of stabilizing the belief to a predefined probability distribution, the belief is stabilized to an orbit (periodic path) of probability distributions. Choosing nodes along these orbits, the node reachability in belief space is achieved and we can form a graph in belief space that can handle higher-order-dynamics or non-stoppable systems (whose velocity cannot be zero), such as fixed wing aircraft. The proposed method takes obstacles into account and provides a query-independent graph, since its edge costs are

independent of each other. Thus, it satisfies the principle of optimality. Therefore, dynamic programming can be utilized to compute the best feedback on the graph. We demonstrate the method's performance on a unicycle robot and a six degrees of freedom small aerial vehicle.

Paper 6. In this paper, the generalized motion planning algorithm (Generalized PRM : GRPM [1, 2, 3]) is extended to a class of multi-agent motion planning problem in presence of process uncertainty and stochastic maps. The proposed algorithm is a hierarchical approach towards constructing a passive coordination strategy which utilizes an existing multiple traveling salesman problem (MTSP) solution methodology in conjunction with the GPRM framework to solve the multi-agent motion planning problem. The proposed algorithm is generalized to tackle multi-agent problems involving heterogeneous agents. The algorithm is used to solve multiagent motion planning problems involving 2-dimensional and 3-dimensional agents in stochastic maps with uncertainty in the motion model. Results indicate that the algorithm successfully solves the problem under uncertainty, and generates a solution having high probability of success. It also demonstrates that the algorithm is scalable in terms of number of start and goal locations, the number of agents and their dynamics.

FIRM: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements

Ali-akbar Agha-mohammadi, Suman Chakravorty and Nancy M Amato

The International Journal of Robotics Research published online 15 November 2013
DOI: 10.1177/0278364913501564

Article

The International Journal of Robotics Research
0(0) 1–37

DOI: 10.1177/0278364913501564
ijr.sagepub.com

FIRM: Sampling-based feedback motion-planning under motion uncertainty and imperfect measurements

Ali-akbar Agha-mohammadi[1], Suman Chakravorty[2] and Nancy M Amato[1]

Abstract

In this paper we present feedback-based information roadmap (FIRM), a multi-query approach for planning under uncertainty which is a belief-space variant of probabilistic roadmap methods. The crucial feature of FIRM is that the costs associated with the edges are independent of each other, and in this sense it is the first method that generates a graph in belief space that preserves the optimal substructure property. From a practical point of view, FIRM is a robust and reliable planning framework. It is robust since the solution is a feedback and there is no need for expensive replanning. It is reliable because accurate collision probabilities can be computed along the edges. In addition, FIRM is a scalable framework, where the complexity of planning with FIRM is a constant multiplier of the complexity of planning with PRM. In this paper, FIRM is introduced as an abstract framework. As a concrete instantiation of FIRM, we adopt stationary linear quadratic Gaussian (SLQG) controllers as belief stabilizers and introduce the so-called SLQG-FIRM. In SLQG-FIRM we focus on kinematic systems and then extend to dynamical systems by sampling in the equilibrium space. We investigate the performance of SLQG-FIRM in different scenarios.

Keywords

Planning, control, uncertainty, information, belief space

1. Introduction

Decision-making under uncertainty is a crucial ability for most robotic systems. In the presence of uncertainty in a robot's motion and uncertainty in its sensory readings, the true robot state is not available for decision-making purposes. In such cases, a state estimation module can provide a probability distribution over all possible states, referred to as *information state* or *belief*. Therefore, decision-making under motion and sensing uncertainties needs to be performed in the information space (belief space). In its most general form, this decision-making can be formulated as a partially observable Markov decision process (POMDP) problem (Astrom, 1965; Smallwood and Sondik, 1973; Kaelbling et al., 1998). However, only a very small class of problems formulated using POMDP can be solved exactly due to its computational complexity (Papadimitriou and Tsitsiklis, 1987; Madani et al., 1999). In particular, planning (i.e. solving POMDPs) over continuous state, control, and observation spaces is a big challenge.

On the other hand, in the absence of uncertainty, sampling-based path-planning algorithms including graph-based methods such as probabilistic roadmap methods (PRMs) (Kavraki et al., 1996) and their variants (see e.g. Amato et al., 1998), and tree-based methods such as rapidly-exploring randomized trees (RRTs) (Lavalle and Kuffner, 2001), expansive space trees (Hsu, 2000) and their variants (e.g. Karaman and Frazzoli, 2011) have shown great success in solving robot motion planning problems. Nevertheless, direct transformation of the roadmap-based methods to planning under uncertainty (in belief space) is a challenge for two main reasons. The first issue is ensuring that the roadmap nodes are reachable. The second challenge is that the incurred costs on different edges of the roadmap depend on each other, which violates a basic assumption in roadmap-based methods that each roadmap edge represent an independent planning problem.

In this paper, we generalize the PRM framework to obtain the feedback-based information roadmap (FIRM) framework that takes into account both motion and sensing uncertainties. FIRM is constructed as a roadmap (graph)

[1]Department of Computer Science and Engineering, Texas A&M University, USA
[2]Department of Aerospace Engineering, Texas A&M University, USA

Corresponding author:
Ali-akbar Agha-mohammadi, Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77840, USA.
Email: aliagha@tamu.edu

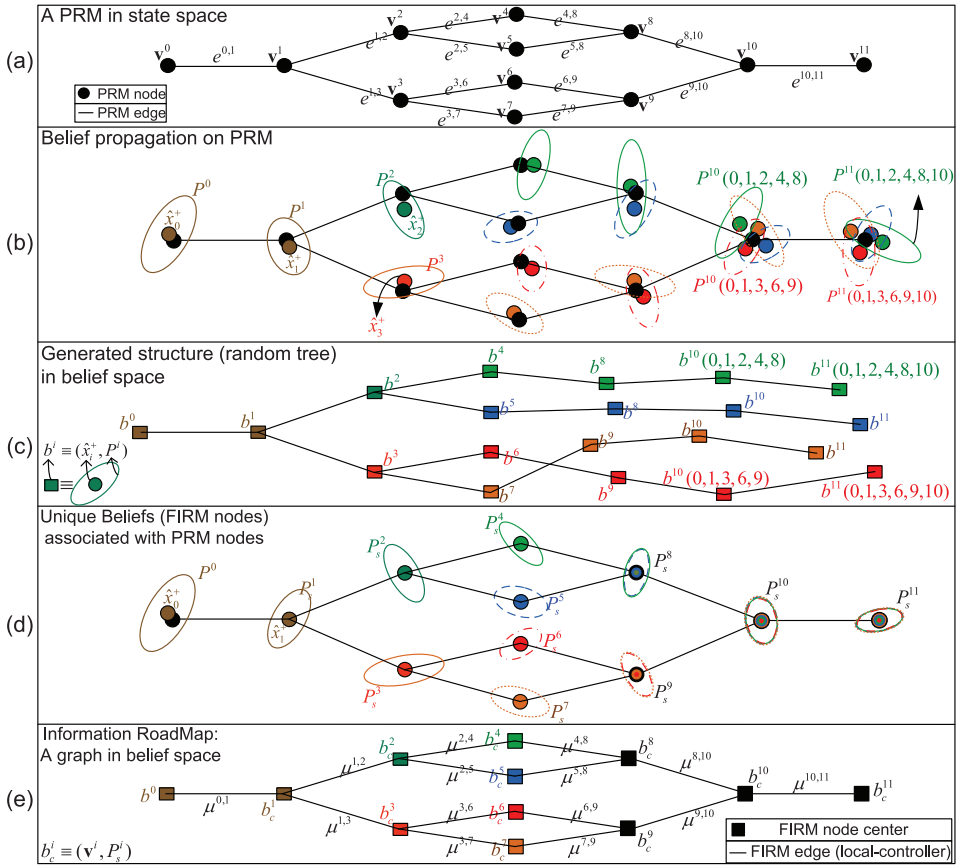


Fig. 1. (a) A simple PRM in state space. (b) Assuming Gaussian belief space, belief snapshots along different paths starting from $\mathbf{v}^0$ and ending at $\mathbf{v}^{11}$ are shown. As can be seen, the obtained belief depends on the path traveled by the robot. For example, $P^{11}(0,1,3,6,9,10)$ denotes the estimation covariance at node $\mathbf{v}^{11}$, when the robot has traversed a path through nodes $(0,1,3,6,9,10)$ prior to node 11. (c) Corresponding belief paths in the belief space. Belief at each node depends on the initial belief, actions taken (edges), and obtained observations (random). Therefore, the generated structure in the belief space is not a graph but a random tree. (d) Unique beliefs assigned to each PRM node. Using stabilizers, regardless of the action and observation history, the belief at each node stops at these predefined beliefs. (e) The FIRM corresponding to the given PRM; b_c^i denotes graph nodes in the belief space and μ^{ij} denotes local planners (graph edges).

in the belief space, where graph nodes are beliefs (rigorously speaking, small subsets of the belief space) and edges are local controllers in belief space. FIRM is an abstract generic framework that relies on the existence of an appropriate belief node sampler and connector (local controller). We also construct a stationary linear quadratic Gaussian controller-based (SLQG-based) instantiation of this generic framework, called SLQG-FIRM, where we provide a specific node sampler and connector. In SLQG-FIRM we first focus on the kinematic systems and then extend it to dynamical systems by restricting sampling space to the equilibrium space. SLQG-FIRM is the first method that generalizes the PRM to the belief space such that the incurred costs on different edges of the roadmap are independent of each other, while providing a straightforward approach to sample reachable belief nodes. This property is a direct consequence of utilizing feedback controllers in the construction of FIRM. Based on this property, the FIRM framework breaks the curse of history in POMDPs (Pineau et al., 2003), and provides the optimal *feedback policy* over the roadmap instead of returning a single nominal path.

Figure 1 illustrates the problem of edge dependence in the direct transformation of PRM to stochastic domains. It also shows the approach of FIRM in generating a graph in belief space with independent edges. Figure 1(a) depicts a simple PRM in the state space with twelve nodes $\mathcal{V} = \{\mathbf{v}^0, \ldots, \mathbf{v}^{11}\}$. Figure 1(b) shows the belief evolution on the underlying PRM. Assuming the belief is Gaussian in this example, we represent a point in belief space using a mean $\widehat{x}^+$ and a covariance P, in other words, a belief b is characterized by the pair $b \equiv (\widehat{x}^+, P)$. In Figure 1(b), mean values are shown by small filled circles, and covariance matrices are shown by their corresponding 3σ ellipses centered at the mean. We drive the system from $\mathbf{v}^0$ toward the node $\mathbf{v}^{11}$. The initial belief at node $\mathbf{v}^0$ is $b^0 \equiv (\widehat{x}_0^+, P^0)$. The belief propagation from left to right starting from b_0 is shown in Figure 1(b).

Although there exists a single edge $e^{(10,11)}$ between nodes $\mathbf{v}^{10}$ and $\mathbf{v}^{11}$ in PRM (see Figure 1(a)), the belief evolution along $e^{(10,11)}$ is not unique (see Figures 1(b)–(c)) since it depends on (i) the initial belief, (ii) obtained observations (observation history), and (iii) the path taken (action history) that has led to $\mathbf{v}^{10}$. Figure 1(c) shows the corresponding belief propagation in the belief space, where each rectangle encodes a mean and covariance. As seen in Figure 1(c), the belief paths do not form a graph; rather, they form a random tree in belief space. Hence, in practice, where observations are random, not only does the number of possible beliefs grow exponentially, but the belief also evolves randomly.

Therefore, to predict edge costs, full knowledge of the belief at the start of the edge is required. This in turn requires full knowledge of the history of observations and actions leading up to the start of the edge.

Even if future observations were assumed to be deterministic for the purpose of planning, the generated structure would still be a tree that grows exponentially in the size of the underlying PRM graph.

In FIRM, we use local feedback planners to drive the belief process toward the predefined unique beliefs associated with PRM nodes (see Figure 1(d)). As a result, the evolution of belief after a FIRM node is reached is independent of the evolution of belief before that node is reached. This breaks the curse of history, allowing us to construct a PRM-like roadmap in the belief space with independent edge costs. Therefore, in contrast to the main body of the literature in motion planning under uncertainty, FIRM can be re-used for future queries and does not need to reconstruct the roadmap every time a new query is submitted.

From an algorithmic perspective, this edge independence is an example of the optimal substructure property. A problem has an optimal substructure only if the optimal solution can be obtained from a combination of optimal solutions to its subproblems (Cormen et al., 2001). To solve a problem using dynamic programming (DP) or its successive approximation schemes, such as Dijkstra's algorithm, the optimal substructure assumption has to hold (Sniedovich, 2006), that is, the cost of any subpath has to be independent of what precedes it and what succeeds it. As mentioned, the direct transformation of sampling-based methods to belief space breaks this assumption, while FIRM preserves it. Furthermore, edge independence allows the challenging task of computing collision probabilities to be done offline, for each edge separately, without performing costly computations repeatedly and without any simplifying assumption.

The current paper draws on earlier work published in conference papers (Agha-mohammadi et al., 2011, 2012b, 2013b). Compared with Agha-mohammadi et al. (2011), in this paper, we construct the FIRM framework more rigorously by detailing the procedure of transforming the POMDP problem to the belief semi-Markov decision process (SMDP) problem, and then to the FIRM Markov decision process (FIRM MDP) problem, where the policy on the graph and overall hybrid policy generated by FIRM are distinguished clearly. Also, in this paper we provide a clearer distinction between the abstract FIRM framework and its instantiations, and we provide more rigorous explanation and proofs on SLQG-FIRM. Further, we append the proofs of the probabilistic completeness of FIRM to this paper, which completes the work in Agha-mohammadi et al. (2012b). We also present new unpublished results on the performance of SLQG-FIRM in more difficult environments, and demonstrate its real-time planning capabilities. Further, we provide a complexity analysis of the method and compare it to state-of-the-art methods.

The outline is as follows. In the next section, we review the most relevant related work. Section 3 provides an overview of the method and its contributions. In Section 4 we describe the general problem of feedback motion-planning under uncertainty, present notation, and formulate the POMDP problem. In Section 5, we present the SLQG-based instantiation of the abstract FIRM framework by providing concrete belief samplers and connectors (local planners). In Section 6, assuming the existence of belief samplers and connectors, we introduce the abstract FIRM framework and detail the process of transforming POMDP to a FIRM MDP. In Section 7, aiming at evaluating the quality of the FIRM solution, we extend the concepts of success and probabilistic completeness to the stochastic setting and prove the probabilistic completeness of the FIRM framework. Experimental results are presented in Section 8. In Section 9, we discuss limitations of the framework, future work, and open issues. Section 10 concludes the paper.

2. Related work

In this section we review the related work and place our work into context. First, we review the related work on planning algorithms under uncertainty, and then we consider the work concerning probabilistic completeness.

2.1. Planning algorithms

Uncertainty in robotic systems usually stems from three sources: (i) motion uncertainty, which results from the noise that affects system dynamics; (ii) sensing uncertainty, caused by noisy sensory measurements, which is also referred to as imperfect state information; and (iii) uncertainty in the environment map, such as uncertain obstacle locations or uncertain locations of features (information sources) in the environment.

Methods such as those in Missiuro and Roy (2006), Guibas et al. (2008), and Nakhaei and Lamiraux (2008) deal with map uncertainty. However, we do not scrutinize these methods, since we assume there is no uncertainty in the environment map. Methods such as those in Alterovitz et al. (2007), Melchior and Simmons (2007),

and Chakravorty and Kumar (2009, 2011) exploit sampling-based motion-planning ideas to deal with motion uncertainty. However, methods that are most related to FIRM consider both motion and sensing uncertainties in planning, where the ultimate goal is to solve a POMDP problem, in other words, to find the best policy that generates optimal actions as a function of belief. However, due to the intractability of the POMDP solution, the practical results using these methods are usually limited to problems with small sets of discrete states (Kaelbling et al., 1998). Point-based POMDP solvers such as those in Porta et al. (2006), Kurniawati et al. (2008), Bai et al. (2010), and Ong et al. (2010) have increased the size of problems that can be solved by POMDPs. However, they do not handle continuous state, control, and observation spaces. For the Gaussian belief case, Van den Berg et al. (2011, 2012) handle continuous spaces locally around a given trajectory in belief space. Platt et al. (2011) generalize the local approaches to non-Gaussian beliefs.

In continuous state, control, and observation space, the main body of methods does not follow the POMDP framework due to its extreme complexity. Instead, these methods return a nominal path as the solution of the planning problem, which is fixed regardless of the process and sensor noise in the execution phase. Censi et al. (2008) propose a planning algorithm based on graph search and constraint propagation on a grid-based representation of the space. Platt et al. (2010) plan in continuous space by finding the best nominal path using nonlinear optimization methods. In the linear quadratic Gaussian motion planning (LQG-MP) method (Van den Berg et al., 2010), among the finite number of RRT paths, the best path is found by simulating the performance of LQG on all RRT paths. Bry and Roy (2011) propose a tree-based approach, in which the underlying nominal trajectory is optimized using RRT*. Vitus and Tomlin (2011) also propose an approach to optimize the underlying trajectory by formulating the problem as a chance-constrained optimal control problem. In Van den Berg et al. (2011), the authors also extend the LQG-MP to roadmaps. Prentice and Roy (2009) and Huynh and Roy (2009) also utilize roadmap-based methods based on the PRM approach, where the best path is found through a breadth-first search on the belief roadmap (BRM). However, in all these roadmap-based methods, the optimal substructure assumption is violated, in other words, the costs of different edges on the graph depend on each other. The point-based POMDP planner in Kurniawati et al. (2012) takes into account motion, observation, and map uncertainties, and advances the previous point-based methods by introducing guided cluster sampling. It starts with a roadmap in the configuration space, and grows a single-query tree in the belief space, rooted in the initial belief. Since these methods return a nominal path instead of a feedback policy, the path needs to be recomputed (in other words, replanning has to be performed) in the case of large deviations or when starting from a new point. However, unless the planning domain is small (e.g. in Platt et al., 2010), replanning using these methods is computationally very expensive. The reason for this is that the constructed planning tree depends on the starting belief, and therefore all computations needed to construct the tree (including predicting future costs) have to be reproduced from the new starting belief. BRM ameliorates this expensive computation using covariance factorization techniques, but it still does not satisfy the optimal substructure assumption. Thus, for a new query from a new initial point, BRM needs to perform the search algorithm again. In the presence of obstacles, recomputing the collision probabilities is also needed, which makes replanning even more expensive. In other words, these methods are single-query, in the sense that the edge costs are computed for a given query.

Since these methods are single-query, online replanning can be done only if the planning domain is small (e.g. in Platt et al., 2010) or if the planning horizon is short, such as receding-horizon-control-based (RHC-based) approaches (e.g. Chakravorty and Erwin, 2011). The method proposed in Toit and Burdick (2010) is an RHC-based method, where the nominal path is updated dynamically over an N-step horizon. The PUMA framework proposed in He et al. (2011) is also an RHC-based framework, where instead of a single action, a sequence of actions (macro-action) is selected at every decision stage. However, these methods entail repeatedly solving open loop optimal control problems at every time step, which is computationally very expensive as the previous computations cannot be reused for the queries from the new initial point. In FIRM, however, a feedback policy (that is, a mapping from belief space to actions) is computed offline. Thus in replanning from a new initial point, the computations need not be reproduced. Thus for a fixed goal, the algorithm is robust to changes in the start point of the query. It is also robust to changes in the goal point, because graph feedback can be computed (see Equation (32)) online, which results in a multi-query roadmap in the belief space.

In the methods that account for sensing uncertainty, the state has to be estimated based on measurements. To handle unknown future measurements in the planning stage, the methods in Censi et al. (2008), Huynh and Roy (2009), Prentice and Roy (2009), Platt et al. (2010), and Toit and Burdick (2010) consider the maximum likelihood (ML) observation sequence to predict the estimation performance. In contrast, FIRM takes all possible future observations into account in the planning stage. The methods in Van den Berg et al. (2010, 2011) also consider all possible future observations.

In the presence of obstacles, due to the dependency of collision events on each other in different time steps, it is a burdensome task to include the collision probabilities in planning. Thus, the methods in Censi et al. (2008), Van den Berg et al. (2010, 2011), and Toit and Burdick (2010)

design some safety measures to account for obstacles in planning. A problem with some of these collision probability measures is that they are built on the assumption that the collision probabilities at different stages along the path are independent of each other, which is not true in general and may lead to very conservative plans (see Figure 2). As a result, different methods (e.g. Patil et al., 2012) aim at providing more accurate and faster ways of computing collision probabilities. In FIRM, however, collision probabilities can be computed and seamlessly incorporated into the planning stage without making any simplifying assumptions.

2.2. Probabilistic completeness

Due to the success of sampling-based methods in many practical planning problems, researchers have investigated the theoretical basis for this success. However, almost all of these investigations have been performed for algorithms that are designed for planning in the absence of uncertainty. The literature in this direction falls into two categories: path-isolation-based methods and space-covering-based methods.

Path-isolation-based analysis: In this approach, one path is chosen, and it is tiled with some sets such as ϵ-balls (Kavraki et al., 1998) or sets with arbitrary shapes but strictly positive measures (Ladd and Kavraki, 2004). Then the success probability is analyzed by investigating the probability of sampling in each of the sets that tile the given path in the obstacle-free space. The methods in Švestka and Overmars (1997), Kavraki et al. (1998), Bohlin (2002), and Ladd and Kavraki (2004) are among those that perform path-isolation-based analysis of the planning algorithm.

Space-covering-based analysis: In space-covering-based analysis, an adequate number of sampled points needed to find a successful path is expressed in terms of a parameter ϵ, which is a property of the environment. A space is ϵ-good if every point in the state space can be connected to at least an ϵ fraction of the space using local planners. The methods in Kavraki et al. (1995) and Hsu (2000) are among these.

These methods were developed for the situation where the desired result from the planning algorithm is a path. However, in the presence of uncertainty, the concept of 'successful path' is no longer meaningful, because on a given path, different policies may result in different success probabilities, where some are interpreted as successful and some are not. Thus, since the planning algorithm returns a policy instead of a path, success has to be defined for a policy. This paper extends these concepts to probabilistic spaces, that is, to sampling-based methods concerning planning under uncertainty. In Section 7, we define and formulate the concepts of successful policy and probabilistic completeness under uncertainty (PCUU).

3. Method overview and contributions

The highlights and contributions of this paper can be divided into theoretical and practical parts. The theoretical highlights can be summarized as follows:

- *Abstract frameworks:* We introduce the abstract information roadmap (IRM) framework as a graph in the belief space, where the graph nodes are beliefs (rigorously speaking, small subsets of the belief space) and edges are local controllers. The abstract FIRM framework is defined as an IRM where local controllers are feedback controllers. These abstract frameworks rely on the existence of an appropriate belief node sampler and connector (local controller) and are general enough to capture any form of belief. Discussing the concept of belief reachability under feedback controllers, we detail the reduction of a POMDP to a tractable MDP on the FIRM graph, referred to as the FIRM MDP.
- *SLQG-FIRM*: To instantiate a FIRM, we need concrete belief samplers and connectors. A concrete example of these components based on SLQG controllers is given in Section 5. Basically, it is shown that under an SLQG controller the belief can be driven into the ϵ-neighborhood of the sampled Gaussian beliefs in finite time, and thus node reachability is achieved. In this fashion, SLQG-FIRM addresses the hard task of sampling in reachable belief space that is required in belief-space planning (Spaan and Vlassis, 2005; Pineau et al., 2006; Kurniawati et al., 2010). The focus of SLQG-FIRM is on kinematic systems. However, we also extend it to dynamical systems by restricting the nodes to the equilibrium space.
- *Graph (multi-query roadmap) in belief space*: FIRM is the first framework that generates a *graph* in the belief space with independent edges. In other words, it is a multi-query roadmap, which distinguishes it from other methods in the belief space.
- *Breaking the curse of history*: A fundamental contribution of FIRM is that the optimal action at a given node does not depend on the traversed nodes, actions, and observations prior to this node, in other words, it is independent of the history of the information process (see Figure 1). This is a direct consequence of inducing reachable belief nodes using feedback controllers, which breaks the curse of history in POMDPs. In addition, the sampling-based nature of the method, borrowed from PRM, allows us to ameliorate the curse of dimensionality.
- *Probabilistic completeness*: Finally, we generalize the conventional concept of 'probabilistic completeness' (which is defined for motion-planning methods in deterministic environments) to the concept of PCUU (which is defined for the planners in the presence of uncertainty). According to this definition, we prove that FIRM is a probabilistically complete algorithm. Moreover, we perform an analysis on the absorption

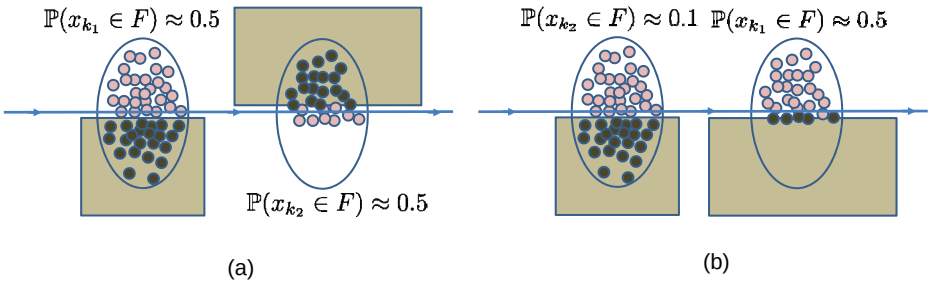


Fig. 2. The dependence of collision events on each other at different time steps. x_k is the robot state and F is the obstacle set (rectangles). P($x_k \in F$) is the collision probability at the k-th time step. Drawn ellipses are 3σ ellipses of Gaussian distributions obtained by Kalman filtering. Also, the samples in Monte Carlo simulation are shown by small circles. The dark ones have collided with obstacles and do not get propagated, and the light ones are the safe samples. Although the overall collision probability in (a) is much more than the collision probability in (b), simplified safety measures based on the ellipse–obstacle intersection area lead to the same safety measure in (a) and (b), and are unable to capture this dependency.

probability of the local planners in the belief space, which provides useful general tools that can be used in analyzing planning methods under uncertainty.

More importantly, FIRM offers a set of practical contributions, which we believe provides an important step toward utilizing POMDPs as a practical tool for robot motion planning under uncertainty. The main practical highlights can be summarized as follows:

- *Efficient planning*: The construction of FIRM is offline and thus online planning (and replanning) is feasible and almost instantaneous.
- *Robustness*: The optimal feedback policy, instead of a nominal path, is computed offline. It is obtained by solving the DP problem associated with the FIRM MDP on the belief graph. As a result, no replanning is needed even in the case of large deviations (or just local real-time replanning is sufficient), and the feedback over the belief space can take care of deviations. Therefore, the method is robust to large deviations. It is also less sensitive to linearization errors, since if the system goes out of the linearization region of a controller, it falls into the valid linearization region of some other controller (assuming a sufficient number of FIRM nodes) that can take the belief and drive it to the goal.
- *Reliability (incorporating obstacles in planning)*: In the FIRM framework, collision probabilities can be computed, which leads to more accurate plans, as opposed to simplified collision measures that may lead to conservative plans (see Figure 2). The obstacles add a *failure node* to the FIRM graph into which the robot can be absorbed. Further, due to the offline construction of FIRM, the heavy computational burden of estimating collision probabilities can be done offline.
- *Scalability*: Belief-space planners usually have an exponential planning complexity either in the number of nodes (if they are sampling-based methods) or in the size of grid (if they rely on discretizing the environment). However, the complexity of the FIRM construction is a constant multiplier of the complexity of the PRM construction. Moreover, the complexity of planning (or replanning) with FIRM is a constant, which is independent of the size of the underlying graph.

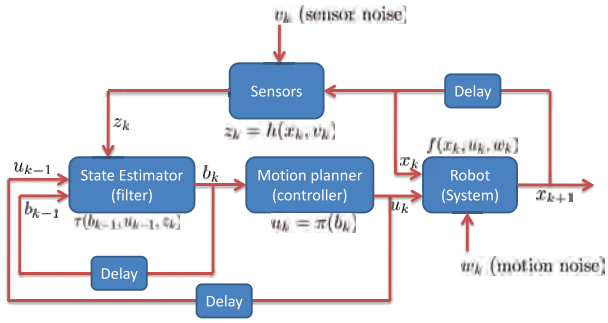


Fig. 3. Block diagram corresponding to the problem of planning under motion and sensing uncertainty.

4. Problem formulation

The main sources of uncertainty in motion planning are the lack of exact knowledge of the robot's motion model, the robot's sensing model, and the environment model, which are referred to as motion uncertainty, sensing uncertainty, and map uncertainty, respectively. In this paper, we focus on motion and sensing uncertainty, but some of the concepts are extensible to problems with map uncertainty. The MDP problem and the POMDP problem are the most general formulations, respectively, for planning problems under motion uncertainty and for planning problems under both motion and sensing uncertainty.

While in the deterministic setting, we seek an optimal path as the solution of the motion-planning problem, in the stochastic setting, we seek an optimal feedback (mapping) π as the solution of the motion-planning problem. In the case of an MDP, π is a mapping from the state space to the

control space, while in the case of POMDP, π is a mapping from the belief space to the control space (see Figure 3). In the rest of paper, we focus on POMDPs.

4.1. Preliminaries and notation

As mentioned, the POMDP formulation is the most general formulation for the planning problem under process (motion) uncertainty and imperfect state information (sensing uncertainty). POMDPs are introduced in Astrom (1965), Smallwood and Sondik (1973), and Kaelbling et al. (1998). In the following, we first explain different elements of the POMDP problem, and then present a form of the POMDP formulation which is known as the belief MDP problem (Thrun et al., 2005; Bertsekas, 2007).

State, control, and observation: Let $x_k \in \mathbb{X}$, $u_k \in \mathbb{U}$, and $z_k \in \mathbb{Z}$ denote the system state, control, and observation at time step k, respectively, where $\mathbb{X} \subseteq \mathbb{R}^{d_x}$, $\mathbb{U} \subseteq \mathbb{R}^{d_u}$, and $\mathbb{Z} \subseteq \mathbb{R}^{d_z}$ are the state, control, and observation spaces. Scalars d_x, d_u, and d_z are the state, control, and observation dimensions and $\mathbb{R}^d$ denotes the d-dimensional Euclidean space.

Basically, system state x encodes all information needed for decision-making at a specific time instant. It is worth noting that the state space in our problem is continuous. Control space $\mathbb{U}$, which contains all possible control inputs (or actions), can also be continuous, and $u_{0:k} := \{u_0, u_1, \dots, u_k\}$ denotes the control history up to step k. Similarly, the observation space $\mathbb{Z}$ that contains all possible observations (sensor measurements) can also be continuous, and $z_{0:k} := \{z_0, z_1, \dots, z_k\}$ is the observation history up to step k.

State evolution model: The process model (or the motion model) $x_{k+1} = f(x_k, u_k, w_k)$ describes how the system state evolves as a function of the applied control u_k and the process (motion) noise w_k, which is distributed according to the (known) probability density function (pdf) $p(w_k|x_k, u_k)$. An equivalent representation of this evolution model is through the transition pdf $p(x'|x, u) : \mathbb{X} \times \mathbb{U} \times \mathbb{X} \to \mathbb{R}_{\geq 0}$, which encodes the probability density of the transition from state x to state x' under the control u.

Observation (sensor) model: Although x_k is sufficient information to make the decision (generate control u_k), in partially observable systems, the system state is *unknown* and the only available data for decision-making is the imperfect measurements of the state made by the sensors. The observation model $z_k = h(x_k, v_k)$ encodes the relation between system state x_k and its measurements z_k, where v_k is the observation noise at time step k, which is distributed according to the (known) pdf $p(v_k|x_k)$. An equivalent representation of this observation model is through the likelihood pdf $p(z|x) : \mathbb{X} \times \mathbb{Z} \to \mathbb{R}_{\geq 0}$.

Information state (belief): In partially observable environments, the available data for decision-making in time step k is the history of observations we have made, $z_{0:k}$, and the history of actions we have taken, $u_{0:k-1}$. Let us denote this data history by $\mathcal{H}_k = \{z_{0:k}, u_{0:k-1}\}$. This data history can be compressed to a conditional probability distribution over all possible states, that is, $b_k = p(x_k|z_{0:k}; u_{0:k-1})$. The pdf $b_k : \mathbb{X} \times \mathbb{Z}^k \times \mathbb{U}^{k-1} \to \mathbb{R}_{\geq 0}$ is called the information state or belief at the kth step. $\mathbb{B}$ denotes the belief space of the problem, containing all possible beliefs $b \in \mathbb{B}$.

Belief evolution model (filter model): In recursive state estimation techniques, belief can be computed recursively. The belief evolution model (or belief dynamics) introduced by this recursion is shown by function $\tau : \mathbb{B} \times \mathbb{U} \times \mathbb{Z} \to \mathbb{B}$, which computes the next belief based on the last action and current observation $b_{k+1} = \tau(b_k, u_k, z_{k+1})$. This belief evolution model can be derived using Bayes' rule and the law of total probability (Thrun et al., 2005; Bertsekas, 2007) as follows:

$$b_{k+1} = p(z_{k+1}|\mathcal{H}_k, u_k)^{-1} p(z_{k+1}|x_{k+1}) \int_{\mathbb{X}} p(x_{k+1}|x_k, u_k)\, b_k dx_k =: \tau(b_k, u_k, z_{k+1}) \quad (1)$$

An equivalent representation of the belief evolution model is through the transition pdf $p(b'|b, u) : \mathbb{B} \times \mathbb{U} \times \mathbb{B} \to \mathbb{R}_{\geq 0}$ that encodes the probability density of the transition from belief b to belief b' under the control u.

Policy: In a partially observable system, the planner π (also called the policy or feedback controller) has to be a function that returns an action u_k given the available data $\mathcal{H}_k$. However, it can be shown that the compression of data $\mathcal{H}_k$ to belief b_k preserves all the information needed for decision-making (Kumar and Varaiya, 1986). Therefore, a policy $\pi(\cdot)$ has to be a function that returns an action u_k given the belief b_k, in other words, $\pi(\cdot) : \mathbb{B} \to \mathbb{U}$:

$$u_k = \pi(b_k), \quad \forall b_k \in \mathbb{B} \quad (2)$$

The space of all possible $\pi(\cdot)$ is denoted by Π.

Cost-to-go: To choose an optimal policy, we need to have a cost function, which is a task-dependent quantity. But let us in general denote the one-step cost of taking action u at belief b by $c(b, u) : \mathbb{B} \times \mathbb{U} \to \mathbb{R}_{\geq 0}$. Then, we can define the cost-to-go function $J^\pi(\cdot) : \mathbb{B} \to \mathbb{R}_{\geq 0}$ from a belief b_0 under the policy π as

$$J^\pi(b_0) := \sum_{k=0}^{\infty} \mathbb{E}[c(b_k, \pi(b_k))] \quad s.t. \quad b_{k+1} = \tau(b_k, \pi(b_k), z_{k+1}), \quad z_k \sim p(z_k|x_k) \quad (3)$$

where $\mathbb{E}[\cdot]$ is the expectation operator. Consider a goal region $B^{goal} \subset \mathbb{B}$ such that, for all u, we have $c(b \in B^{goal}, u) = 0$; in other words, the goal region is cost-absorbing. Then, the above cost-to-go would be finite for a policy that can drive the state to the goal region in finite time.

4.2. POMDP problem

Given the motion model f, observation model h, and cost-to-go J^π, the POMDP problem seeks the best policy that

minimizes the cost-to-go function from every belief in the belief space. Formally, if we denote the *optimal cost-to-go* function by $J(\cdot)$, we can define *optimal policy* $\pi(\cdot)\colon \mathbb{B} \to \mathbb{U}$, which is the solution of POMDP as follows:

$$J(\cdot) := \min_{\Pi} J^{\pi}(\cdot) \tag{4}$$

$$\pi = \arg\min_{\Pi} J^{\pi}(\cdot) \tag{5}$$

This formulation of the POMDP problem is also known as the *belief-MDP* problem (Thrun et al., 2005; Bertsekas, 2007), because it is an MDP over the belief space.

Dynamic programming: It is well known that the optimal cost-to-go is obtained by solving the following stationary DP equation on the belief space $\mathbb{B}$ (Thrun et al., 2005; Bertsekas, 2007). Subsequently, the solution of the POMDP (i.e. π) can be computed as a function that returns the argument of this minimization, that is, returns the optimal action at every belief:

$$J(b) = \min_{u}\{c(b,u) + \int_{\mathbb{B}} p(b'|b,u)\, J(b')\, db'\}, \forall b \in \mathbb{B} \tag{6a}$$

$$\pi(b) = \arg\min_{u}\{c(b,u) + \int_{\mathbb{B}} p(b'|b,u)\, J(b')\, db'\}, \ \forall b \in \mathbb{B} \tag{6b}$$

However, it is well known that this DP equation is exceedingly difficult to solve since it is defined over the entire belief space and suffers from the curse of history (Pineau et al., 2003) and the curse of dimensionality.

Constrained POMDP problems: The presence of constraints makes this problem even more difficult. We denote the constraint set (or the failure set) in the state and control space by $F \subset \mathbb{X} \times \mathbb{U}$, which needs to be avoided by the system, in other words, $(x_k, u_k) \notin F$, for all k.

4.3. Problem description

We aim at constructing a sampling-based solution to the belief MDP problem. The main goals of this paper are as follows.

SLQG-based FIRM: First, we construct a roadmap in belief space utilizing SLQG controllers as belief stabilizers. We perform this construction for a certain class of systems, and show that the belief reachability condition is guaranteed. In designing SLQG-FIRM, we first focus on kinematic systems (satisfying $x = f(x,0,0)$). Then, using the notion of equilibrium space and restricting the sampling to this space, we apply the method to dynamical systems as well.

General FIRM framework: After studying the concrete SLQG-FIRM example, we consider the more general case, where, for a general system, assuming that there exists a controller under which belief reachability is guaranteed, we (i) construct a graph in the belief space encoding the failure probabilities on its edges, (ii) reduce the intractable belief MDP in Equation (4) into a tractable MDP problem on this graph, and (iii) compute a feedback solution on this graph.

5. SLQG-FIRM

In this section, we discuss a particular instance of the FIRM framework in which belief reachability is accomplished by SLQG controllers. In Section 6, we propose the general FIRM framework.

We start this section by restricting our attention to the class of systems that SLQG-FIRM can handle. Then, we present a brief review of LQG controllers and address how we can define nodes in the belief space to satisfy reachability using SLQG controllers. Next we explain the procedure of constructing local controllers (i.e. FIRM edges) and the SLQG-based FIRM graph. Finally, we compute transition probabilities and costs associated with each graph edge and compute the graph feedback.

5.1. Preliminaries on SLQG

In this section, we assume the noise is Gaussian, and we start by defining the notation needed in dealing with Gaussian beliefs.

Gaussian belief space: We denote the random estimation vector by x^+, whose distribution is $b_k = p(x_k^+) = p(x_k|z_{0:k}, u_{0:k-1})$, and denote the mean and covariance of x^+ by $\widehat{x}^+ = \mathbb{E}[x^+]$ and $P = \mathbb{E}[(x^+ - \widehat{x}^+)(x^+ - \widehat{x}^+)^{\mathrm{T}}]$, respectively. Denoting the Gaussian belief space by $\mathbb{GB}$, every function $b(\cdot) \in \mathbb{GB}$ can be characterized by a mean-covariance pair $(\widehat{x}^+, P)$. Abusing the notation, we also show this pair by $b \equiv (\widehat{x}^+, P) \in \mathbb{R}^n \times \mathbb{S}_+^n$, where the mean vector belongs to the n-dimensional Euclidean space $\mathbb{R}^n$ and the covariance matrix belongs to the space of all positive semi-definite $n \times n$ matrices $\mathbb{S}_+^n$.

LQG controllers: An LQG controller is composed of a Kalman filter (KF) as the state estimator and a linear quadratic regulator (LQR) controller (see Figure 3). Thus, the belief dynamic $b_{k+1} = \tau(b_k, u_k, z_{k+1})$ is known and comes from the Kalman filtering equations, and the controller $u_k = \mu(b_k)$ that acts on the belief comes from the LQR equations. Considering a quadratic cost for state error and control error, LQG is an optimal controller for linear systems with Gaussian noise (Bertsekas, 2007). However, it is also often used for stabilization of nonlinear systems around a given trajectory or around a given point.

Stationary and time-varying LQG: Time-varying LQG is designed to track a given trajectory, in which at every time step a different feedback policy is utilized. SLQG is a time-invariant policy, in which LQG is designed around a given point, say $\mathbf{v}$, to steer the state of the system to $\mathbf{v}$ (Bertsekas, 2007). In Appendices B and C we review these controllers in detail.

Equilibrium space: Let us denote a configuration of a robotic system (Lozano-Perez, 1983) by q. Kinematic models are specified in terms of the configuration variable q, while dynamical models are specified by the state $x = (q, \dot{q})$, where $\dot{q}$ denotes the corresponding velocities. In SLQG-FIRM, we sample the underlying PRM nodes (stabilizer parameters) from the configuration space. Thus, for

dynamical systems, we impose the condition $\dot{q} = 0$ on the samples, in other words, we sample from the equilibrium space of the system, which is denoted by X in this paper.

Remark 1. *FIRM can be generalized to cases that do not need to sample in equilibrium space. For example, in systems such as fixed-wing aircraft, the system cannot reach the zero velocity $\dot{q} = 0$. In such cases, SLQG is not a suitable choice and one needs to design more appropriate controllers, such as periodic controllers as detailed in Agha-mohammadi et al. (2012c, 2013a). In such a case, we sample periodic maneuvers as FIRM nodes. In other words, we go from periodic trajectory to periodic trajectory instead of going from point to point (Agha-mohammadi et al., 2012c, 2013a).*

5.2. Belief stabilizers

In SLQG-FIRM nodes, we use SLQG controllers as belief stabilizers, that is, as a tool to reach (stabilize to) a predefined belief (FIRM node). To explain how SLQG works as a belief stabilizer, consider a fixed point $\mathbf{v} \in \mathsf{X}$ in the state space and consider the following linear (linearized) system about $\mathbf{v}$:

$$x_{k+1} = \mathbf{A}x_k + \mathbf{B}u_k + \mathbf{G}w_k, \quad w_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}) \tag{7a}$$

$$z_k = \mathbf{H}x_k + v_k, \quad v_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}) \tag{7b}$$

SLQG controller: The goal of the SLQG controller designed about $\mathbf{v}$ is to keep the state as close as possible to the desired point $\mathbf{v}$ and also keep the energy consumed at a reasonable level. More rigorously, SLQG minimizes the following quadratic cost:

$$J = \mathbb{E}\left\{\sum_{k\geq 0} (x_k - \mathbf{v})^{\mathrm{T}} \mathbf{W}_x (x_k - \mathbf{v}) + u_k^{\mathrm{T}} \mathbf{W}_u u_k\right\} \tag{8}$$

where $\mathbf{W}_x$ and $\mathbf{W}_u$ are positive-definite weight matrices that are defined by the user. In Appendix C, the SLQG controller minimizing the above cost is discussed in detail. However, in brief, the belief propagation and control generation is carried out as follows:

$$b_{k+1} \equiv \begin{bmatrix} \widehat{x}_{k+1}^{+} \\ P_{k+1}^{+} \end{bmatrix} = \begin{bmatrix} \mathbf{A}\widehat{x}_k^{+} + \mathbf{B}u_k + \mathbf{K}_{k+1}(z_{k+1} - \mathbf{H}(\mathbf{A}\widehat{x}_k^{+} + \mathbf{B}u_k)) \\ (I - \mathbf{K}_{k+1}\mathbf{H})(\mathbf{A}P_k^{+}\mathbf{A}^{\mathrm{T}} + \mathbf{G}\mathbf{Q}\mathbf{G}^{\mathrm{T}}) \end{bmatrix} \equiv \tau(b_k, u_k, z_{k+1}) \tag{9}$$

where $\mathbf{K}_k$ is called the Kalman gain at the kth time step and is computed as follows:

$$\mathbf{K}_{k+1} = (\mathbf{A}P_k^{+}\mathbf{A}^{\mathrm{T}} + \mathbf{G}\mathbf{Q}\mathbf{G}^{\mathrm{T}})\mathbf{H}^{\mathrm{T}}(\mathbf{H}(\mathbf{A}P_k^{+}\mathbf{A}^{\mathrm{T}} + \mathbf{G}\mathbf{Q}\mathbf{G}^{\mathrm{T}})\mathbf{H}^{\mathrm{T}} + \mathbf{M}\mathbf{R}\mathbf{M}^{\mathrm{T}})^{-1} \tag{10}$$

The control signal is generated using a stationary feedback gain $\mathbf{L}_s$:

$$u_k = -\mathbf{L}_s(\widehat{x}_k^{+} - \mathbf{v}) =: \mu(b_k), \; \mathbf{L}_s = (\mathbf{B}_s^{\mathrm{T}} S_s \mathbf{B}_s + \mathbf{W}_u)^{-1} \mathbf{B}_s^{\mathrm{T}} S_s \mathbf{A}_s \tag{11}$$

where S_s is the solution of the following discrete algebraic Riccati equation (DARE):

$$S_s = \mathbf{W}_x + \mathbf{A}_s^{\mathrm{T}} S_s \mathbf{A}_s - \mathbf{A}_s^{\mathrm{T}} S_s \mathbf{B}_s (\mathbf{B}_s^{\mathrm{T}} S_s \mathbf{B}_s + \mathbf{W}_u)^{-1} \mathbf{B}_s^{\mathrm{T}} S_s \mathbf{A}_s \tag{12}$$

Controllable and observable pairs: Consider an $n \times n$ matrix $\mathbf{A}$. A pair of matrices $(\mathbf{A}, \mathbf{B})$ is called a controllable pair if the controllability matrix $\mathfrak{C} = [\mathbf{B}, \mathbf{AB}, \mathbf{A}^2\mathbf{B}, \ldots, \mathbf{A}^{n-1}\mathbf{B}]$ has rank n (Bertsekas, 2007). A pair of matrices $(\mathbf{A}, \mathbf{H})$ is called observable if the pair $(\mathbf{A}^{\mathrm{T}}, \mathbf{H}^{\mathrm{T}})$ is controllable (Bertsekas, 2007).

Controllable and observable systems: Let us also define the matrices $\check{\mathbf{Q}}$ and $\check{\mathbf{W}}_x$ such that $\mathbf{G}\mathbf{Q}\mathbf{G}^{\mathrm{T}} = \check{\mathbf{Q}}\check{\mathbf{Q}}^{\mathrm{T}}$, $\mathbf{W}_x = \check{\mathbf{W}}_x^{\mathrm{T}}\check{\mathbf{W}}_x$. We next consider a class of linear systems and quadratic cost weights that satisfy the following property.

Property 1. *Pairs $(\mathbf{A}, \mathbf{B})$ and $(\mathbf{A}, \check{\mathbf{Q}})$ are controllable pairs, and pairs $(\mathbf{A}, \mathbf{H})$ and $(\mathbf{A}, \check{\mathbf{W}})$ are observable pairs.*

In the following, we present three lemmas, through which we can construct reachable SLQG-FIRM nodes for the systems that satisfy Property 1. However, approaches such as periodic LQG (PLQG)-based FIRM (Agha-mohammadi et al., 2012c) or dynamic feedback linearization (DFL)-based FIRM (Agha-mohammadi et al., 2012a) extend this class of systems by excluding the controllability part in Property 1, and thus consider a broader class of systems.

Lemma 1. *Consider the SLQG controller designed to drive the state of the system in Equation* (7) *to a point $\mathbf{v} \in \mathsf{X}$. Given that Property 1 is satisfied, in the absence of a stopping region, the belief b_k under SLQG controller converges to a unique stationary belief b_s, in distribution (i.d.). In other words, the distribution over belief converges to a unique distribution. That is,*

$$b_k \xrightarrow{i.d.} b_s \sim \mathcal{N}(b_c, \mathsf{C}) \tag{13}$$

Note that b_k is a random belief that converges to another random belief b_s. In the Gaussian setting, the distribution over the random belief b_s is $\mathcal{N}(b_c, \mathsf{C})$, where $b_c = \mathbb{E}[b_s] \equiv (\mathbf{v}, P_s)$. The stationary estimation covariance matrix P_s is characterized in Lemma 2, and the covariance C is characterized in Appendix C.

Proof. In Appendix C, we review the SLQG and prove Lemma 1. □

Lemma 2. *Given Property 1, the following DARE has a unique symmetric positive-definite solution (Bertsekas, 2007), denoted by P_s^-:*

$$P_s^- = \mathbf{G}\mathbf{Q}\mathbf{G}^{\mathrm{T}} + \mathbf{A}(P_s^- - P_s^-\mathbf{H}^{\mathrm{T}}(\mathbf{H}P_s^-\mathbf{H}^{\mathrm{T}} + \mathbf{R})^{-1}\mathbf{H}P_s^-)\mathbf{A}^{\mathrm{T}} \tag{14}$$

Moreover, the stationary covariance matrix P_s introduced in Lemma 1 is computed as:

$$P_s = P_s^- - P_s^-\mathbf{H}^{\mathrm{T}}(\mathbf{H}P_s^-\mathbf{H}^{\mathrm{T}} + \mathbf{R})^{-1}\mathbf{H}P_s^- \tag{15}$$

Proof. See Appendix C or Bertsekas (2007). □

Now we state the main result, through which we can construct *reachable* FIRM nodes under SLQG-based belief stabilizers:

Lemma 3. *Consider the SLQG controller designed to drive the state of the system in Equation* (7) *to a point* $\mathbf{v} \in \mathsf{X}$. *Suppose matrix* $\mathbf{H}$ *is full rank and Property 1 is satisfied. Then, any set* $B \subset \mathbb{B}$ *whose interior contains* $b_c \equiv (\mathbf{v}, P_s)$ *is reachable under the designed SLQG controller starting from any Gaussian distribution. Moreover, the estimation covariance* P_k *converges to the unique deterministic stationary covariance* P_s.

Proof. See Appendix D. □

Therefore, based on Lemma 3, SLQG can accomplish the belief reachability for appropriately chosen region B. In the next subsection we explicitly characterize region B.

5.3. Designing SLQG-FIRM nodes

Underlying PRM: As mentioned, to construct a FIRM we first construct an underlying PRM (Kavraki et al., 1996). In the SLQG-FIRM, nodes of the underlying PRM, denoted by $\{\mathbf{v}^j\}_{j=1}^N$, are sampled from the obstacle-free space. Considering linear systems or nonlinear systems that are locally well approximated by linearization, we linearize the system about every PRM node. Let us denote the linear (linearized) system about $\mathbf{v}^j$ as follows:

$$x_{k+1} = \mathbf{A}^j x_k + \mathbf{B}^j u_k + \mathbf{G}^j w_k, \quad w_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}^j) \tag{16a}$$

$$z_k = \mathbf{H}^j x_k + v_k, \quad v_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}^j) \tag{16b}$$

where w_k and v_k are motion and measurement noise, respectively, drawn from zero-mean Gaussian distributions with covariances $\mathbf{Q}^j$ and $\mathbf{R}^j$.

FIRM nodes: To design the jth FIRM node B^j, we first design the SLQG controller μ_s^j (see Equations (9) and (11)) corresponding to the system in Equation (16). The controller μ_s^j is called the jth node controller or the jth belief stabilizer. Given Property 1, based on Lemma 1, the limiting random belief $b_s^j \equiv (\widehat{x}_s^{+j}, P_s^j)$ exists, and $\widehat{x}_s^{+j}$ and P_s^j are the stationary estimation mean and covariance, respectively. Note that under SLQG, $\widehat{x}_s^{+j}$ is a random variable and P_s^j is a deterministic matrix. Moreover, in Lemma 1, it is shown that $b_c^j = \mathbb{E}[b_s^j] \equiv (\mathbf{v}^j, P_s^j)$, where P_s^j is shown to be unique and computed in Lemma 2.

Thus, we can characterize the jth node center:

$$b_c^j \equiv (\mathbf{v}^j, P_s^j) \tag{17}$$

As a result, considering B^j as a ball with an arbitrary radius $\epsilon > 0$ centered at b_c^j, the pair (B^j, μ_s^j) is a *proper pair*, based on Lemma 3; in other words, B^j is reachable under μ_s^j. Thus, one can define the jth FIRM node as $B^j = \{b : \|b - b_c^j\|_b < \delta\}$, where $\|\cdot\|_b$ denotes a suitable norm in belief space and δ defines the FIRM node size. A typical example of such a FIRM node in Gaussian belief space can be defined by considering mean and covariance separately:

$$B^j = \{b \equiv (x, P) : \|x - \mathbf{v}^j\| < \delta_1, \|P - P_s^j\|_m < \delta_2\} \tag{18}$$

where δ_1 and δ_2 are suitably small thresholds that determine the size of FIRM node B^j. Moreover, $\|\cdot\|$ is a suitable vector norm and $\|\cdot\|_m$ is a suitable matrix norm. We denote the set of all SLQG-FIRM nodes as $\mathbb{V} = \{B^i\}$.

5.4. Designing SLQG-FIRM edges

A FIRM edge is actually a local planner (local feedback controller). In SLQG-based FIRM, the local controller representing the (i,j)th edge is denoted by μ^{ij}. The role of μ^{ij} is to drive the belief from the node B^i to the node B^j. Based on Lemma 3, for a linear system, if we choose $\mu^{ij} = \mu_s^j$, as was done in Agha-mohammadi et al. (2011), the node B^j is reachable under μ^{ij}. However, to better cope with nonlinearities, we construct the local controller μ^{ij} by preceding the node controller with a time-varying LQG controller $\overline{\mu}_k^{ij}$, which is called an *edge controller* here. Time-varying LQG controllers are described in detail in Appendix B.

PRM edge: To design edge controllers, first the underlying PRM edges, denoted by $\mathcal{E} = \{\mathbf{e}^{ij}\}$, have to be constructed. For kinematics-based models there are many different methods in the PRM literature to construct such edges. For dynamical models, there are fewer choices. A few examples are in Van den Berg and Overmars (2007) and Agha-mohammadi et al. (2012c).

Edge controllers: An edge controller $\overline{\mu}_k^{ij}$ in SLQG-FIRM is built by linearizing the system along the (i,j)th PRM edge $\mathbf{e}^{ij}$ and designing a time-varying LQG controller to track it (see Appendix B). The edge controller has two major roles. First, it tries to track the PRM edge and thus exploits the available information on the PRM edges, such as some clearance from the obstacles. Second, in the case where the neighboring PRM nodes are not close to each other, it takes the belief into the valid linearization region of the jth belief stabilizer, where it hands the system over to the belief stabilizer, and the belief stabilizer in turn takes the system to the jth FIRM node.

Local controllers: Thus, overall, the (i,j)th local controller μ^{ij} is the concatenation of the (i,j)th edge controller $\overline{\mu}_k^{ij}$ and jth node controller (belief stabilizer) μ_s^j. We denote the set of all SLQG-FIRM edges by $\mathbb{M} = \{\mu^{ij}\}$ and the set of all SLQG-FIRM edges originating from B^i by $\mathbb{M}(i)$.

SLQG-FIRM: Formally, we define SLQG-FIRM as a graph with the set of nodes $\mathbb{V} = \{B^i\}$ and the set of edges (or local controllers) $\mathbb{M} = \{\mu^{ij}\}$. The set of controllers originating from B^i is denoted by $\mathbb{M}(i) \subset \mathbb{M}$.

5.5. Transition probabilities and edge costs

To find a feedback on a FIRM graph, we need to compute the cost associated with the graph edges. Moreover, we

include the constraint set F into the planning with FIRM by computing the probability of violating the constraint $(x,u) \notin F$ along the graph edges. Let us denote the cost of taking controller μ^{ij} at node B^i by $C^g(B^i,\mu^{ij})$. Superscript g refers to the 'global' (or 'graph-level') quantities, as these quantities are used to find the global policy (or policy on the graph). Similarly, let $\mathbb{P}^g(B^j|B^i,\mu^{ij})$ and $\mathbb{P}^g(F|B^i,\mu^{ij})$ denote the probabilities of the transition to B^j and F under μ^{ij}, respectively. These quantities are rigorously defined in Section 6 and their connection with the original POMDP is established. However, in this subsection, we just give an example of how such costs and transition probabilities can be computed.

Transition probabilities: Computing transition probabilities $\mathbb{P}^g(\cdot|B^i,\mu^{ij})$ in general can be computationally expensive. Here, we utilize particle-based methods to approximate the distributions and thus compute the collision probabilities. Basically, we can approximate the failure and reachability probabilities based on the number of particles that violate the constraints (hit the set F) and based on the number of particles that can reach the target node (hit the set B^j). The method is described in more detail with the experiments in Section 8.1.4. The dependency of collision events on each other in different time steps, which is ignored in most collision probability computation methods in the POMDP literature, can be taken into account rigorously in particle-based methods. Owing to the offline construction of FIRM, the high computational burden of particle-based approaches can be tolerated. However, any other method for computing transition probabilities can also be adopted, such as that in Patil et al. (2012).

Edge costs: The FIRM edge costs in general and their derivation based on the one-step costs of the original POMDP problem are defined are Section 6. However, roughly speaking, we can define the cost $C^g(B^i,\mu^{ij})$ as the sum of all one-step costs along the edge until the system reaches the target node B^j or hits the failure set F. Depending on the application, one can define a variety of cost functions. Here, we form a cost function based on a linear combination of the estimation accuracy and edge traverse time. This cost function aims to find paths for which the estimator (and hence the controller) can perform well, and also to find faster paths. An indicator of estimation error is the trace of estimation covariance. Thus, we define $\Phi^{ij} = \mathbb{E}[\sum_{k=1}^{\mathcal{T}} \mathrm{tr}(P_k^{ij})]$ along the edge. In SLQG, the covariance matrix evolves deterministically and thus the expectation operator can be omitted. However, if the filter of choice in the edge controller is the extended Kalman filter (EKF), the covariance matrix evolution is also stochastic, and this measure can take into account its stochasticity. Let us denote the mean stopping time under controller μ^{ij} as $\widehat{\mathcal{T}}^{ij}$. Then, the total edge cost is considered as a linear combination of estimation accuracy and expected stopping time, with suitable coefficients α_1 and α_2:

$$C^g(B^i,\mu^{ij}) = \alpha_1 \Phi^{ij} + \alpha_2 \widehat{\mathcal{T}}^{ij} \tag{19}$$

5.6. Graph feedback on SLQG-FIRM

Graph policy: Graph policy $\pi^g : \mathbb{V} \to \mathbb{M}$ is a function that returns an edge (local controller) for any given node of the graph. We denote the space of all graph policies by Π^g. To choose the best graph policy in Π^g we define the optimal graph cost-to-go J^g from every graph node.

Graph cost-to-go: The cost-to-go from a given node B^i is equal to the cost of the next taken controller, that is, $C^g(B^i,\pi^g(B^i))$, plus the expected cost-to-go from the next node or from the failure set. In other words, the DP equations for this graph are

$$J^g(B^i) = \min_{\mathbb{M}(i)} C^g(B^i,\mu^{ij}) + J^g(F)\,\mathbb{P}^g(F|B^i,\mu^{ij}) + J^g(B^j)\,\mathbb{P}^g(B^j|B^i,\mu^{ij}) \tag{20a}$$

$$\pi^g(B^i) = \arg\min_{\mathbb{M}(i)} C^g(B^i,\mu^{ij}) + J^g(F)\,\mathbb{P}^g(F|B^i,\mu^{ij}) + J^g(B^j)\,\mathbb{P}^g(B^j|B^i,\mu^{ij}) \tag{20b}$$

in which $J(F)$ is a suitably high user-defined cost-to-go for hitting the obstacles. The cost-to-go from goal node B^{goal} is defined to be zero, in other words, $J^g(B^{goal}) = 0$.

Solving SLQG-FIRM DP: The DP in equation (20) is a tractable DP as it is defined on a finite number of graph nodes. Computing the transition costs and probabilities offline, this DP can be solved online using standard techniques, such as value/policy iteration methods, for any submitted query. As a result, FIRM is indeed a multi-query roadmap in belief space. Moreover, if the goal node is fixed and only the starting point of the query changes, then this DP can be solved offline and π^g can be stored as a look-up table.

Offline construction of SLQG-FIRM: Algorithm 1 details the construction of SLQG-FIRM with a given goal node.

5.7. Planning with SLQG-FIRM (query phase)

Given that the FIRM graph is computed offline, the online phase of planning (and replanning) on the roadmap becomes very efficient, and thus feasible in real time. In this section, we assume that the goal node is fixed and we just input the start point as the query. However, as discussed in the previous subsection, one can easily submit queries with different goal locations by solving the DP online. If the initial belief b_0 of the submitted query does not belong to any B^i, we create a singleton set $B_0 = \{b_0\}$ as the initial FIRM node. To connect B_0 to the FIRM graph, we go back into the state space, where the underlying PRM is constructed. There, we add a new PRM node to the graph $\mathbf{v}_0$, which is the expected value of the robot state, in other words, $\mathbf{v}_0 = \mathbb{E}[x_0]$. Then, we connect $\mathbf{v}_0$ to the underlying PRM graph based on the connecting function of the adopted PRM. We denote the set of newly added edges originating from $\mathbf{v}_0$ by $\mathcal{E}(0)$. Then, corresponding to each edge in $\mathcal{E}(0)$, we design a local controller and call the set of them $\mathbb{M}(0)$.

Algorithm 1: Offline construction of SLQG-FIRM

input : Free space map, X_{free}
output : FIRM graph $\mathcal{G}$
Sample PRM nodes $\mathcal{V} = \{\mathbf{v}^j\}_{j=1}^{N}$ and construct its edges $\mathcal{E} = \{\mathbf{e}^{ij}\}$;
forall the *PRM nodes* $\mathbf{v}^j \in \mathcal{V}$ **do**
    Design the node controller (SLQG) μ_s^j about the node $\mathbf{v}^i$ using Equation (11);
    Compute associated b_c^j using Equation (17);
    Construct FIRM node B^j using Equation (18);
Construct $\mathbb{V} = \{B^i\}$;
forall the *PRM edges* $\mathbf{e}^{ij} \in \mathcal{E}$ **do**
    Design the edge controller (time-varying LQG) $\overline{\mu}_k^{ij}$ along the edge $\mathbf{e}^{ij}$ (detailed in Appendix B);
    Construct the local controller μ^{ij} by concatenating edge controller $\overline{\mu}_k^{ij}$ and node controller μ_s^j;
    Set $b_0 = b_c^i$;
    Generate sample belief paths $b_{0:\mathcal{T}}$ and ground truth paths $x_{0:\mathcal{T}}$ induced by controller μ^{ij} invoked at B^i;
    Compute the transition probabilities $\mathbb{P}^g(F|B^i,\mu^{ij})$ and $\mathbb{P}^g(B^j|B^i,\mu^{ij})$ and transition cost $C^g(B^i,\mu^{ij})$;
Construct $\mathbb{M} = \{\mu^{ij}\}$;
Compute the cost-to-go J^g and feedback π^g over the FIRM nodes by solving the DP in Equation (20);
$\mathcal{G} = (\mathbb{V},\mathbb{M},J^g,\pi^g)$;
return $\mathcal{G}$;

Algorithm 2: Online phase algorithm (planning or replanning with SLQG-FIRM)

input : Initial belief b_0, FIRM graph $\mathcal{G}$
if $\exists B^i \in \mathbb{V}$ *such that* $b_0 \in B^i$ **then**
    compute $\mu^{ij} = \pi^g(B^i)$;
else
    Compute $\mathbf{v}_0 = \mathbb{E}[x_0]$ based on b_0, and connect $\mathbf{v}_0$ to the PRM. Let $\mathcal{E}(0)$ denote the set of outgoing edges from $\mathbf{v}_0$;
    Set $B_0 = \{b_0\}$; design local controllers associated with edges in $\mathcal{E}(0)$. Call the set of these local controllers $\mathbb{M}(0)$;
    forall the $\mu \in \mathbb{M}(0)$ **do**
        Generate sample belief paths $b_{0:\mathcal{T}}$ and ground truth paths $x_{0:\mathcal{T}}$ induced by controller μ invoked at b_0;
        Compute the transition probabilities $\mathbb{P}^g(F|B_0,\mu)$ and $\mathbb{P}^g(B(\mu)|B_0,\mu)$ and transition costs $C^g(B_0,\mu)$;
    Set $i = 0$ and choose the best initial local controller μ^{ij} within the set $\mathbb{M}(0)$ using Equation (21);
while $B^i \neq B^{goal}$ **do**
    Set B^j as the target node of μ^{ij};
    while $b_k \notin B^j$ *and 'no collision'* **do**
        Apply the control $u_k = \mu^{ij}(b_k)$ to the system;
        Get the measurement z_{k+1} from sensors;
        if *Collision happens* **then return** Collision;
        Update belief as $b_{k+1} = \tau(b_k,\mu^{ij}(b_k),z_{k+1})$;
    Set $B^i = B^j$, then compute $\mu^{ij} = \pi^g(B^i)$;

Finally, we choose the best initial controller among the local controllers in $\mathbb{M}(0)$ using

$$\mu_0^*(\cdot) = \arg\min_{\mu\in\mathbb{M}(0)}\{C^g(B_0,\mu) + \mathbb{P}^g(B(\mu)|B_0,\mu)\,J^g(B(\mu)) + \mathbb{P}^g(F|B_0,\mu)\,J^g(F)\} \quad (21)$$

where $B(\mu)$ is the target node of the controller μ. Under the controller μ_0^*, belief evolves and enters one of FIRM nodes, if no collision occurs. From this FIRM node, a combination of the global graph policy π^g and the local edge policies $\{\mu^{ij}\}$ can take the belief to the goal node, as explained below.

Merging global and local feedbacks: After computing a global graph feedback π^g and local edge feedbacks $\{\mu^{ij}\}$, we can construct a full feedback π. Actually, at every time instance, π is equal to one of the local feedbacks, which is chosen by the global feedback in the last visited node. In other words, given the current FIRM node, we use policy π^g defined on FIRM nodes to find μ^* and pick μ^* to move the robot into $B(\mu^*)$. This process is continued until the system reaches the goal region or hits the failure set. Algorithm 2 illustrates this procedure.

Kidnapped robot problem: In robotics, the kidnapped robot problem commonly refers to a situation where an autonomous robot in operation is carried to an arbitrary location (Choset et al., 2005). Consider a kidnapped robot problem in a known environment. Just after the robot is kidnapped it would be risky to apply any control, because the robot may be close to an obstacle. Thus, in such a scenario, we first initialize the system belief with a Gaussian with large covariance and go into an 'information gathering' mode, where we do not apply any control signal and only gather measurements until the covariance shrinks to a reasonable covariance or it remains unchanged for a significant amount of time (i.e. when there is no additional information to reduce the uncertainty). Afterwards, we connect the resulting belief to the FIRM nodes and continue applying the FIRM policy to move the robot toward the goal region. A more efficient approach of handling this problem is detailed in Agha-mohammadi et al. (2013c) using innovation signals.

6. General FIRM framework

The goal of this section is to construct a general FIRM framework, assuming that there exists a mechanism to guarantee belief reachability. As a result, if for a certain class of systems one comes up with a controller that can accomplish

belief reachability, a graph in belief space directly follows according to this general framework.

To construct the general FIRM, we start by defining elements and assumptions needed in the FIRM construction. Accordingly, we transform the original intractable POMDP problem into an SMDP problem in belief space, inspired by sampling-based methods. Then, we construct an arbitrarily good approximation to the solution of this belief SMDP over finite subsets of belief space (FIRM nodes). Doing so, we end up with a tractable MDP, the so-called FIRM MDP. We discuss this derivation first for the obstacle-free case and then we add the obstacles to the planning framework. We characterize the quality of the solution obtained by FIRM via its success probability and provide a generic algorithm for planning with FIRM.

6.1. Feedback controllers and reachability

Belief transition probability: As discussed in Section 4, in partially observable environments the available data for decision-making at time step k can be compressed as the information state or belief b_k. As discussed, using dynamic estimation schemes, belief can be propagated as $b_{k+1} = \tau(b_k, u_k, z_{k+1})$ (see Equation (1)), which can be presented as a one-step transition pdf $p(b_{k+1}|b_k, u_k)$ or a one-step transition probability $\mathbb{P}(B|b_k, u_k) = \int_B p(b_{k+1}|b_k, u_k)$, where $B \subset \mathbb{B}$.

Feedback controllers and induced transition probability: In partially observable environments, at each stage, the decision-making process is performed based on the belief at that stage. Therefore, a controller is a mapping from the belief space to the control space, in other words, $\mu(\cdot): \mathbb{B} \to \mathbb{U}$. Accordingly, a controller μ induces a Markov chain with the one-step transition probability $\mathbb{P}_1(B|b, \mu) := \mathbb{P}(B|b, \mu(b))$ over the belief space.

Hitting time: Let $\mathcal{T}(D|b, \mu) \in [0, \infty]$ denote the hitting time on the set $D \subset \mathbb{B}$ under the controller μ starting from belief b. Formally it is defined as

$$\mathcal{T}(D|b, \mu) := \min\{k \geq 0, b_k \in D | b_0 = b, \mu\} \tag{22}$$

Stopping region: We call region $B \subset \mathbb{B}$ a stopping region of the controller μ if we force the controller to stop executing as the belief reaches the region B; in other words, for all $b \in B$, we impose $\mathbb{P}_1(B|b, \mu) = 1$.

n-step transition probability: We define the n-step transition probability as the probability of landing in the stopping region B in *at most* n steps:

$$\mathbb{P}_n(B|b, \mu) := \Pr(\mathcal{T}(B|b, \mu) \leq n) \tag{23}$$

Stationary transition probability: Consider the controller μ that starts executing from belief b and stops executing when the belief enters region B. Thus, we can define $\mathbb{P}(B|b, \mu)$ as the transition probability from b to B induced by μ, when the controller stops executing; in other words, $\mathbb{P}(B|b, \mu)$ would be the probability of landing in the stopping region B in a finite amount of time:

$$\mathbb{P}(B|b, \mu) := \Pr(\mathcal{T}(B|b, \mu) < \infty) \tag{24}$$

Reachability and accessibility: The stopping region B is called reachable under a controller μ starting from b if $\mathbb{P}(B|b, \mu) = 1$. The stopping region B is called accessible under a controller μ from b, if $\mathbb{P}(B|b, \mu) > 0$.

αT-reachability: The stopping region B is called αT-reachable under a controller μ starting from b if $\mathbb{P}_T(B|b, \mu) = \Pr(\mathcal{T}(B|b, \mu) \leq T) > \alpha$, in other words, if the controller can drive the system into B in fewer than T steps with a probability greater than α.

Reachability basin: The reachability basin $\breve{B}$ associated with the pair (μ, B) is the set of all beliefs from which B is reachable under μ in the absence of constraints. The reachability (and αT-reachability) basins are thus defined respectively as follows:

$$\breve{B} = \{b \in \mathbb{B} \ : \ \mathbb{P}(B|b, \mu) = 1\} \tag{25}$$

$$\breve{B}(\alpha, T) = \{b \in \mathbb{B} \ : \ \mathbb{P}_T(B|b, \mu) > \alpha\} \tag{26}$$

Clearly, $B \subset \breve{B}$, and in practical cases, B is much smaller than $\breve{B}$.

6.2. FIRM graph

In this section, we assume that there are no constraints (i.e. $F = \emptyset$), and we reduce planning over the entire belief space to planning over a representative graph constructed within the belief space. Doing so, we can reduce the MDP problem in (4) over the continuous space into a tractable MDP problem defined over the graph nodes.

Stabilizer sampling: The first step in the construction of the proposed framework is to sample a set of stabilizers $\{\mu^j\}$, where each stabilizer $\mu(\cdot)$ is a mapping from the belief space to the control space. Typically, every stabilizer is characterized by a d_v-vector of parameters $\mathbf{v} \in \mathbb{R}^{d_v}$; in other words, we can denote the jth stabilizer more rigorously as $\mu^j(\cdot; \mathbf{v}^j): \mathbb{B} \to \mathbb{U}$. As a result, we can sample the parameters $\mathcal{V} = \{\mathbf{v}^j\}$ and then construct a stabilizer corresponding to each parameter. One can view the set $\mathcal{V}$ as a set of underlying PRM nodes in the parameter space.

Sampling FIRM nodes: FIRM nodes $\{B_j\}$ are disjoint sets in the belief space, where the jth node has to be chosen such that it is reachable under the j-stabilizer; in other words, $\mathbb{P}(B^j|b, \mu^j) = 1$, with a sufficiently large $\breve{B}$. We discuss the size of $\breve{B}$ further below. Note that, for practical purposes, the reachability condition can be replaced by αT-reachability if needed.

Connecting samples: Consider a set of N samples $\{(\mu^i, B^i)\}_{i=1}^N$, where the reachability basin of the ith sample is denoted by $\breve{B}^i$. Now, consider $\{B^i\}_{i=1}^N$ as the nodes of a graph. The node B^i is connected to the node B^j if, starting from any $b \in B^i$, we can reach B^j using μ^j. In other words

B^i is connected to the node B^j if $B^i \subset \check{B}^j$. Again, the reachability condition can be replaced by the αT-reachability condition.

Checking connection condition: For simple systems (linear with Gaussian noise) and some controllers (such as SLQG), the connection condition can be checked analytically. However, in general, checking this connection condition analytically may be very difficult. In such cases, the Markov chain induced by the controller can be simulated numerically (e.g. using particle-based methods). Accordingly, we can approximate the reachability (or αT-reachability) probability and check if the condition is true or not. Since this process is done offline, the computational burden can be tolerated. However, as we will see further below, in many cases designing suitable edge controllers in practice increases the reachability probability such that practically one can assume the reachability is satisfied, and so there is no need to propagate the probability distribution.

Stopping region: By definition, the graph node B associated with the controller μ acts as the stopping region of the controller. However, if the process under the stabilizer hits another graph node before its corresponding graph node, we can stop the controller and pick the best controller from this intermediate node. Therefore, we can extend the stopping region for all controllers to the union of all nodes $\Psi := \cup_{l=1}^{N} B^l$. As a result, we will not necessarily have $\mathbb{P}(B^i|b,\mu^i) = 1$ since the process may hit some other node before B^i. However, we will have $\mathbb{P}(\Psi|b,\mu^i) = 1$ for all i in the absence of constraints.

Local controllers (simplified connecting strategy): To ease the connection step, and to have more distant nodes, we can precede each stabilizer by a time-varying controller (referred to as the edge controller). To illustrate this idea, consider two nodes B^i and B^j, where $B^i \nsubseteq \check{B}^j$; that is, B^i cannot be connected to B^j through μ^j. In this case, we can connect the underlying state nodes $\mathbf{v}^i$ and $\mathbf{v}^j$ in the state space by a finite trajectory e^{ij} (say of length ι) and then design a time-varying controller $\overline{\mu}_k^{ij}$, for $k = 0, 1, \ldots, \iota$, to track this finite trajectory. Therefore, if the node B^i is in the basin of reachability of the pair $(\overline{\mu}_k^{ij}, \check{B}^j)$, then obviously B^i would be in the basin of reachability of the pair (μ^{ij}, B^j), where the controller $\mu^{ij} = \{\overline{\mu}_{0:\iota}^{ij}, \mu^j\}$. We call μ^{ij} the (i,j)th local controller, as it connects the node B^i to the node B^j.

Graph: Formally, we define the constructed graph with the set of nodes $\mathbb{V} = \{B^i\}_{i=1}^N$ and the set of edges (or local controllers) $\mathbb{M} = \{\mu^{ij}\}$. The set of controllers available at B^i is denoted by $\mathbb{M}(i)$ (i.e. the set of edges starting from B^i). Similar to PRM, in which the path (final solution) is constructed as a concatenation of edges on the roadmap, in FIRM, the policy is constructed by the concatenation of the local policies. However, it is worth noting that with this construction we still perform planning in a continuous space and do not discretize the control space.

Local controllers versus macro-actions: By the term 'macro-action' we mean a sequence of controls (actions) (He et al., 2010, 2011). In other words, a macro-action is a sequence of open-loop policies. It is important to note that a local controller is *not* a macro-action, but rather a sequence of policies (macro-policy), each of which is a mapping from belief space to the continuous control space. Using macro-actions results in an open-loop policy, which cannot compensate for the belief-state deviation from the planned path. However, under local controllers (macro-policies), the effect of noise can be compensated for, due to the feedback nature of the controllers, and thus, the belief can be steered towards a stopping region.

6.3. Belief SMDP

In this section, we reduce planning over the entire belief space into planning over a subset of belief space, which is actually the union of FIRM graph nodes; that is, $\Psi = \cup_j B^j$.

SMDP transition costs: First, we generalize the concept of one-step cost $c(b,u): \mathbb{B} \times \mathbb{U} \to \mathbb{R}_{\geq 0}$ to the one-step SMDP cost $C^s(b,\mu): \mathbb{B} \times \mathbb{M} \to \mathbb{R}_{\geq 0}$, which represents the cost of invoking the local controller $\mu(\cdot)$ at belief state b; in other words,

$$C^s(b,\mu) := \sum_{t=0}^{\mathcal{T}} c(b_t, \mu(b_t)) | b_0 = b) \tag{27}$$

where $\mathcal{T} := \mathcal{T}(\Psi|b,\mu)$.

Belief SMDP: According to the above definitions, the original POMDP, formulated using DP in Equation (6), can be reduced to an SMDP (Sutton et al., 1999) in the belief space, referred to as a *belief SMDP*:

$$J^s(b) = \min_{\mu \in \mathbb{M}(i)} C^s(b,\mu) + \int_{\Psi} p(b'|b,\mu)\, J^s(b')\, db', \ \forall b \in B^i, \ \forall i \tag{28}$$

The integration over the entire belief space in Equation (6) is reduced to integration over the sampled nodes (that is, Ψ) in Equation (28), as μ stops executing.

6.4. FIRM MDP

Graph transitions: The DP in (28), though computationally more tractable than the original POMDP, is defined on the continuous neighborhoods B^i and thus is still formidable to solve. However, for sufficiently small B^i and sufficiently smooth cost functions, the cost-to-go of all beliefs in B^i are approximately equal. Thus, we can define the graph-level transition cost and probabilities $C^g : \mathbb{V} \times \mathbb{M} \to \mathbb{R}$ and $\mathbb{P}^g : \mathbb{V} \times \mathbb{V} \times \mathbb{M} \to [0,1]$ on the FIRM graph, in other words, over the finite space $\mathbb{V}$, such that $\mathbb{P}^g(B^j|B^i,\mu)$ is the transition probability from B^i to B^j under the local planner μ. Similarly, $C^g(B^i,\mu)$ denotes the cost of invoking local planner μ at the FIRM node B^i. Accordingly, $J^g : \mathbb{V} \to \mathbb{R}$ is the cost-to-go function over the FIRM nodes. These roadmap-level quantities are defined using the following 'piecewise constant approximation', which is an arbitrarily good approximation for smooth enough functions and

sufficiently small B^i:

$$\forall b \in B^i, \forall i \quad \begin{cases} J^g(B^i) := J^s(b_c^i) \approx J^s(b) \\ C^g(B^i,\mu) := C^s(b_c^i,\mu) \approx C^s(b,\mu) \\ \mathbb{P}^g(\cdot|B^i,\mu) := \mathbb{P}(\cdot|b_c^i,\mu) \approx \mathbb{P}(\cdot|b,\mu) \end{cases} \tag{29}$$

where b_c^i is a representative point in B^i. For example, if B^i is a ball, the typical value for b_c^i is the center of B^i. This approximation essentially states that any belief in the region B^i is represented by b_c^i for the purpose of decision-making.

Obstacle-free FIRM MDP: Given the approximation in Equation (29), the DP equation in (28) becomes

$$\begin{aligned} J^g(B^i) = J^s(b_c^i) &= \min_{\mu\in\mathbb{M}(i)} C^s(b_c^i,\mu) + \int_\Psi p(b'|b_c^i,\mu)\, J^s(b')\, db' \\ &= \min_{\mu\in\mathbb{M}(i)} C^s(b_c^i,\mu) + \sum_j \int_{B^j} p(b'|b_c^i,\mu)\, J^s(b')\, db' \\ &\approx \min_{\mu\in\mathbb{M}(i)} C^g(B^i,\mu) + \sum_j \int_{B^j} p(b'|b_c^i,\mu)\, J^g(B^j)\, db' \\ &= \min_{\mu\in\mathbb{M}(i)} C^g(B^i,\mu) + \sum_j J^g(B^j)\, \mathbb{P}(B^j|b_c^i,\mu) \\ &= \min_{\mu\in\mathbb{M}(i)} C^g(B^i,\mu) + \sum_j J^g(B^j)\, \mathbb{P}^g(B^j|B^i,\mu),\ \forall i \end{aligned} \tag{30}$$

Accordingly, we can get the graph feedback $\pi^g : \mathbb{V} \to \mathbb{M}$ through the following DP:

$$J^g(B^i) = \min_{\mu\in\mathbb{M}(i)} C^g(B^i,\mu) + \sum_j \mathbb{P}^g(B^j|B^i,\mu)\, J^g(B^j),\ \forall i \tag{31a}$$

$$\pi^g(B^i) = \arg\min_{\mu\in\mathbb{M}(i)} C^g(B^i,\mu) + \sum_j \mathbb{P}^g(B^j|B^i,\mu)\, J^g(B^j),\ \forall i \tag{31b}$$

Thus, the original POMDP over the entire belief space becomes a finite N-state MDP in Equation (31) defined on the finite set of FIRM nodes $\mathbb{V} = \{B^i\}_{i=1}^N$. We call the MDP in Equation (31) the FIRM MDP in the absence of obstacles. It is worth noting that $J^g(\cdot): \mathbb{V} \to \mathbb{R}$ is the cost-to-go function over the FIRM nodes, which assigns a cost-to-go for every FIRM node B^i, and the mapping $\pi^g(\cdot): \mathbb{V} \to \mathbb{M}$ is a mapping over the FIRM graph from FIRM nodes into the set of local controllers that returns the optimal local controller that has to be taken at any FIRM node. Given $C^g(B,\mu)$ for all (B,μ) pairs, the DP in Equation (31) can be solved offline using standard techniques such as the value/policy iteration to yield a feedback policy π^g over FIRM nodes $\{B^i\}$.

6.5. Incorporating obstacles into FIRM MDP

In the presence of obstacles (i.e. state or control constraints), we may not assume that the local controller $\mu^{ij}(\cdot)$ can drive any $b \in B^i$ into B^j with probability one. Instead, we have to specify the failure probabilities that the robot collides with an obstacle (hits the failure set F).

Let us generalize the transition probabilities by defining $\mathbb{P}(F|b,\mu)$ as the probability of hitting failure set F before hitting stopping region Ψ under μ starting from b. Similarly, we generalize $\mathbb{P}^g$ such that $\mathbb{P}^g(F|B^i,\mu) := \mathbb{P}(F|b_c^i,\mu)$. Finally, we generalize the cost-to-go function by adding F to its input set, that is, $J^g : \{\mathbb{V}, F\} \to \mathbb{R}_{\geq 0}$, such that $J^g(F)$ is a user-defined suitably high cost for hitting obstacles. Note that the cost-to-go from the goal node is zero, that is, $J^g(B^{goal}) = 0$. Therefore, we can modify Equation (31) to incorporate constraints by repeating the procedure in the previous subsection to get the FIRM MDP in the presence of obstacles:

$$\begin{aligned} J^g(B^i) = \min_{\mu\in\mathbb{M}(i)} & C^g(B^i,\mu) + J^g(F)\, \mathbb{P}^g(F|B^i,\mu) \\ & + \sum_j J^g(B^j)\, \mathbb{P}^g(B^j|B^i,\mu) \end{aligned} \tag{32a}$$

$$\begin{aligned} \pi^g(B^i) = \arg\min_{\mu\in\mathbb{M}(i)} & C^g(B^i,\mu) + J^g(F)\, \mathbb{P}^g(F|B^i,\mu) \\ & + \sum_j J^g(B^j)\, \mathbb{P}^g(B^j|B^i,\mu) \end{aligned} \tag{32b}$$

All that is required to solve the above DP equation is the values of the costs $C^g(B^i,\mu)$ and the transition probability functions $\mathbb{P}^g(\cdot|B^i,\mu)$. Thus, the main difference from the obstacle-free case is the addition of a 'failure' state to the FIRM MDP along with associated probabilities of failure from various nodes B^i.

6.6. Overall policy π

The overall feedback $\pi : \mathbb{B} \to \mathbb{U}$ is generated by combining the global policy π^g on the graph and local policies $\{\mu^{ij}\}$. Suppose at the kth time step the active local controller is shown by μ_k^*. It remains unchanged ($\mu_{k+1}^* = \mu_k^*$) and keeps generating control signals based on the belief b_k at each time step, until the belief reaches the corresponding stopping region, Ψ. Once the belief enters the stopping region $\Psi = \cup_j B^j$, it is in a graph node, say $B_k^* \in \mathbb{V}$. Accordingly, the global policy π^g chooses the next local controller, that is, $\mu_{k+1}^* = \pi^g(B_k^*)$. Thus, this hybrid policy is stated as follows:

$$u_k = \pi(b_k) = \begin{cases} \mu_k^*(b_k),\ \mu_k^* = \pi^g(B_{k-1}^*), & \text{if } b_k \in B_{k-1}^* \\ \mu_k^*(b_k),\ \mu_k^* = \mu_{k-1}^*, & \text{if } b_k \notin \Psi \end{cases} \tag{33}$$

Initial controller: Given the initial belief b_0, if b_0 is in one of the graph nodes, then we just choose the best local controller using π^g. However, if b_0 does not belong to any of the graph nodes, we first make a singleton set $B^0 = \{b_0\}$ and connect it to the graph nodes based on the connect methods discussed in Section 6.2. Denoting the outgoing edges (local controllers) from B^0 by $\mathbb{M}(0)$, we compute

the transition cost $C^g(B^0,\mu)$, the transition probabilities $\mathbb{P}^g(B^j|B^0,\mu)$ for all j, and failure probability $\mathbb{P}(F|B^0,\mu)$ for invoking local controllers $\mu \in \mathbb{M}(0)$ at B^0. Then, we choose the best initial controller μ_0^* as

$$\mu_0^* = \begin{cases} \arg\min_{\mu\in\mathbb{M}(0)} \{C^g(B^0,\mu) + \mathbb{P}^g(F|B^0,\mu)\, J^g(F) \\ \quad + \sum_j \mathbb{P}^g(B^j|B^0,\mu)\, J^g(B^j)\}, & \text{if } b_0 \notin \Psi \\ \pi^g(B^r), & \text{if } \exists r, \text{s.t. } b_0 \in B^r \end{cases} \quad (34)$$

It is worth noting that computing μ_0^* is the only part of the computation that depends on the initial belief b_0 and that has to be performed online; in other words, if a large deviation occurs, μ_0^* is the only part that needs to be reproduced for the new initial point. After μ_0^* drives the system to a graph node, from there on the optimal policy is already known. Computing μ_0^* is feasible in real time as $\mathbb{M}(0)$ contains a limited number of edges.

6.7. Success probability

We would also like to quantify the quality of the solution π in the presence of obstacles. To this end, we require the probability of success of the policy π^g at the higher-level Markov chain on FIRM nodes given by Equation (32b). Without loss of generality let us assume that the first node B^1 is the goal node B^{goal}. The DP in Equation (32) has $N+1$ states $\{F, B^{goal}, B^2, \ldots, B^N\}$ that can be decomposed into three disjoint classes: the failure class $\{F\}$, the goal class $\{B^{goal}\}$, and the transient class $\{B^2, B^3, \ldots, B^{N+1}\}$. The goal and failure classes are absorbing recurrent classes of this Markov chain. As a result, the transition probability matrix of this higher-level $N+1$ state Markov chain can be decomposed as follows (Norris, 1997):

$$\mathcal{P} = \begin{bmatrix} \mathcal{P}_f & 0 & 0 \\ 0 & \mathcal{P}_{goal} & 0 \\ \mathcal{R}_f & \mathcal{R}_{goal} & \mathcal{Q} \end{bmatrix} \quad (35)$$

where $\mathcal{P}_{goal} = \mathbb{P}^g(B^1|B^1,\cdot) = 1$ and $\mathcal{P}_f = \mathbb{P}^g(F|F,\cdot) = 1$, since goal and failure classes are the absorbing recurrent classes; in other words, the system stops once it reaches the goal or it fails. $\mathcal{Q}$ is a matrix that represents the transition probabilities between transient nodes in the transient class, whose (i,j)th element is $\mathcal{Q}[i,j] = \mathbb{P}^g(B^{i+1}|B^{j+1},\pi^g(B^{j+1}))$. Vectors $\mathcal{R}_{goal}$ and $\mathcal{R}_f$ are $(N-1)\times 1$ vectors that represent the probability of transient nodes $\mathbb{V}\setminus B^{goal}$ getting absorbed into the goal and failure node, respectively; that is, $\mathcal{R}_{goal}[j] = \mathbb{P}^g(B^1|B^{j+1},\pi^g(B^{j+1}))$ and $\mathcal{R}_f[j] = \mathbb{P}^g(F|B^{j+1},\pi^g(B^{j+1}))$. Then, it can be shown that the success probability from any desired node $B^i \in \mathbb{V}\setminus B^{goal}$ is as follows (Norris, 1997):

$$\begin{aligned} \mathbb{P}(\text{success}|B^i,\pi^g) &:= \mathbb{P}(B^{goal}|B^i,\pi^g) \\ &= \Gamma_{i-1}^{\mathrm{T}}(I-\mathcal{Q})^{-1}\mathcal{R}_{goal}, \quad \forall i \geq 2 \end{aligned} \quad (36)$$

where Γ_i is a column vector with all elements equal to zero except the ith element, which is set to one. Note that the vector $\mathcal{P}^s = (I-\mathcal{Q})^{-1}\mathcal{R}_{goal}$ includes the success probability from every graph node.

In the next section, we will discuss the success probability in more detail in the context of probabilistic completeness. However, according to the computed $\mathbb{P}(\text{success}|B^i,\pi^g)$, one can compute the success probability from any given initial belief b_0 as

$$\mathbb{P}(\text{success}|b_0,\pi) = \sum_j \mathbb{P}(B^j|b_0,\mu_0^*)\,\mathbb{P}(\text{success}|B^j,\pi^g) \quad (37)$$

where μ_0^* is given by Equation (34). Then, this success probability is compared with a minimum acceptable success probability, denoted by p_{min}. If the condition $\mathbb{P}(\text{success}|b_0,\pi) > p_{min}$ is not satisfied, then the number of nodes in the graph has to be increased until the condition is satisfied. If, from the initial point b_0, a successful policy in the class of admissible policies exists, then this procedure will eventually find a successful policy by increasing the number of nodes, due to the probabilistic completeness of the method, which is discussed in Section 7.1.

6.8. Generic FIRM algorithms

The generic algorithms for the offline construction of FIRM and online planning with FIRM are presented in Algorithms 3 and 4, respectively. Concrete instantiations of these algorithms for SLQG-FIRM are given in Algorithms 1 and 2, respectively.

Algorithm 3: Generic construction of the FIRM graph (offline)

1. Sample a set of stabilizer parameters $\mathcal{V} = \{\mathbf{v}^i\}$ and construct stabilizers $\mathbb{M} = \{\mu^i\}$ accordingly;
2. Sample a set of belief nodes $\mathbb{V} = \{B^i\}$ such that they satisfy the reachability condition;
3. Connect the belief nodes using local controllers μ^{ij};
4. For each B^i and $\mu \in \mathbb{M}(i)$, compute the transition cost $C^g(B^i,\mu)$ and transition probabilities $\mathbb{P}^g(B^j|B^i,\mu)$ and $\mathbb{P}^g(F|B^i,\mu)$ associated with invoking μ at B^i;
5. Solve the graph DP in Equation (32) to compute feedback π^g over graph nodes, and compute π accordingly;

Single-query versus multi-query: As mentioned earlier, most approaches for planning in belief space in continuous state, action, and observation spaces result in query-dependent plans. However, one of the contributions of FIRM is that its construction does not depend on the query. In Algorithms 3 and 4, it is assumed that the goal is fixed for all queries; in this case in the planning phase we are only robust to changes in the starting point of the query. However, to make the algorithms also robust to changes in the

Algorithm 4: Generic planning (or replanning) on FIRM (online)

Given an initial belief b_0, invoke the controller $\mu_0(\cdot)$ in Equation (34) to take the robot into some FIRM node B;
while $B \neq B^{goal}$ **do**
  Given the system is in FIRM node B, invoke the global feedback policy π^g to choose the local feedback policy $\mu(\cdot) = \pi^g(B)$;
  Let the local controller $\mu(\cdot)$ execute until the robot is absorbed into a FIRM node B' or until it hits the failure set;
  if *Collision happens* **then return** Collision;
  Update current node $B \leftarrow B'$;

goal belief, one can just move the last line of Algorithm 3 to the first line of Algorithm 4. Note that the computationally expensive part of Algorithm 3 is the computation of edge costs, which is independent of the start and goal locations of the submitted query.

6.9. Discussion

In summary, in FIRM we aim to transform the original POMDP problem into a belief SMDP problem and solve it on a subset of belief space. Given the smoothness of the cost function and transition probabilities, the solution of the FIRM MDP is arbitrarily close to the solution of the belief SMDP over FIRM nodes. The important characteristic of FIRM is that it is solved offline and thus performing the online phase of planning (or replanning) is computationally feasible in real time. To exploit the generic FIRM framework, one has to find (B, μ) pairs, where B is reachable (or αT-reachable) under μ, as FIRM nodes and edges. Also, transition costs and probabilities need to be computed. Finally, the corresponding FIRM MDP needs to be solved, which provides a global feedback policy on the graph that can be used in planning, as detailed in Algorithm 4. SLQG-FIRM, presented in Section 5, is an instance of FIRM, in which the design of local controllers μ^{ij} and FIRM nodes B^i is based on the properties of SLQG controllers.

7. Probabilistic completeness under uncertainty

In this section, we extend the concept of probabilistic completeness of planning algorithms for deterministic systems to the concept of probabilistic completeness of planning algorithms under uncertainty, based on Agha-mohammadi et al. (2012b). Accordingly, in the next subsection, we discuss the probabilistic completeness of the FIRM-based algorithms. We start by reviewing the definition of success and probabilistic completeness in the deterministic case, and then we extend these definitions to the stochastic case.

Success in the deterministic case: In the deterministic case, such as conventional PRM, the outcome of the planning algorithm is a path. Thus, success is defined for paths: for a given initial and goal point, a successful path is a path connecting the start point to the goal point which lies entirely in the obstacle-free space.

Probabilistic completeness in the deterministic case: In the absence of uncertainty, a sampling-based motion-planning algorithm is probabilistically complete if by increasing the number of samples, the probability of finding a successful path, if one exists, asymptotically approaches one.

A difference between the deterministic and the probabilistic case: In the presence of uncertainty, success cannot be defined for a path and has to be defined for a policy. Indeed, on a given path, different policies may result in different success probabilities. Moreover, under uncertainty, one can only assign a probability to reaching the goal. Thus, to define success for a policy we consider a threshold $p_{min} \in [0, 1]$ and decide about success or failure accordingly.

Successful policy: In the presence of uncertainty, the solution of the planning algorithm is a function, called a closed-loop policy or feedback. Therefore, success is defined for policies: for a given initial belief b_0 and goal region B^{goal}, a successful policy is a policy under which the probability of reaching the goal from the given initial point is greater than some predefined threshold p_{min}. In other words, π is successful for a given b_0 if $\mathbb{P}(\text{success}|b_0, \pi) := \mathbb{P}(B^{goal}|b_0, \pi) > p_{min}$.

Policy in sampling-based methods: In sampling-based methods, a policy is parametrized by a set of samples. These samples can be in the state or belief space, based on the algorithm. Let us denote these samples in a generic space by $\{\gamma_1, \gamma_2, \ldots, \gamma_N\}$. Thus, we can highlight the dependency of the sampling-based policy on the samples by the notation $\pi(\cdot; \{\gamma_1, \gamma_2, \ldots, \gamma_N\})$. The number of samples is denoted by N.

Strong probabilistic completeness under uncertainty (SPCUU): Suppose there exists a successful policy $\check{\pi}$. Then a sampling-based motion-planning algorithm is SPCUU if increasing the number of samples without bound causes the probability of finding a successful policy to approach one. In other words, if there exists a successful policy $\check{\pi}$, then we have the following property for the sampling-based policy π:

$$\lim_{N \to \infty} \mathbb{P}(B^{goal}|b_0, \pi) > p_{min} \tag{38}$$

where N is the number of samples in the sampling-based method.

Achieving an algorithm that is SPCUU requires searching in the entire space of policies, which is a computationally intractable task. Usually in solving POMDPs the space of admissible policies is restricted to a sufficiently rich subset of policy space, denoted by Π, within which the method searches for the best policy. Restricting the successful policy to the set Π, we define below a weaker notion of PCUU.

PCUU: Suppose that there exists a successful policy $\check{\pi} \in \Pi$. Then, a sampling-based motion-planning algorithm is PCUU if, when increasing the number of samples without bound, the probability of finding a successful policy approaches one. In other words, if there exists a successful policy $\check{\pi} \in \Pi$, then for the sampling-based policy π, we have $\lim_{N\to\infty} \mathbb{P}(B^{goal}|b_0,\pi) > p_{min}$.

As discussed in Section 6, in FIRM, inspired by the sampling-based PRM framework, this reduction from the entire function space to the restricted set of policies Π is performed by sampling feedback local planners and concatenating them. Therefore, the structure of local planners defines the set Π. Each local planner μ^{ij} is parametrized by its corresponding parameter $\mathbf{v}^j$. However, as mentioned in Section 6.2, we can consider the set $\mathcal{V} = \{\mathbf{v}^i\}$ as the set of underlying PRM nodes. Thus, any policy $\pi \in \Pi$ is parametrized by the set of underlying PRM nodes $\mathcal{V} = \{\mathbf{v}^i\}_{i=1}^N$. We highlight this dependency explicitly through the notation $\pi(\cdot;\mathcal{V})$. Therefore, the PCUU condition for FIRM can be written more explicitly as

$$\lim_{N\to\infty} \mathbb{P}(B^{goal}|b_0,\pi(\cdot;\mathcal{V})) > p_{min} \tag{39}$$

For a concrete instantiation of FIRM, we can explicitly characterize the set Π. For example, in SLQG-FIRM, Π is the set of all possible policies that can be generated by concatenating LQG controllers.

7.1. Probabilistic completeness of FIRM

Obviously, FIRM-based methods are not SPCUU algorithms. However, in this section, we show that under mild practical conditions, FIRM-based methods are PCUU algorithms. We first provide an analysis of the local planners in belief space, and then state the assumptions more rigorously.

Notation: The norm $\|\cdot\|$ is the supremum norm when it is applied to functions. The norm $\|\cdot\|_{op}$ is applied on operators and it stands for the operator norm (Keener, 2000). It is worth noting that in this section, by the word 'continuous', we mean 'Lipschitz continuous'. Finally, we assume that $\mathbb{X}_{free}$ is a compact set.

Hyper-state: $\mathcal{X} = (x,b) \in \mathbb{X}_h$ is referred to as hyper-state (or h-state), which is a state-belief pair. The space of all h-states is called hyper-state space (h-state space) $\mathbb{X}_h = \mathbb{X} \times \mathbb{B}$. Further, $p^{\mu}(\mathcal{X}'|\mathcal{X})$ denotes the one-step transition pdf induced by the local controller, μ, over the h-state space. Also, let $\mathbb{P}_n(S|\mathcal{X},\mu)$ denote the transition probability from h-state $\mathcal{X}$ into the set $S \subset \mathbb{X}_h$ in at most n steps.

Local planner and extended stopping region: The role of the (i,j)th local planner or local controller is to drive the belief from the region B^i to its stopping region B^j in the belief space (for ease of notation, we ignore the case where the controller can stop in any FIRM node, and we restrict its stopping region to B^j). In the presence of obstacles, we extend the concept of stopping region to include obstacles also. The stopping regions $\{B^j\}$ in the belief space and the stopping region F in the state space can both be extended to the h-state space, respectively denoted by $\{\mathcal{B}^j\}$ and $\mathcal{F}$, where $\mathcal{B}^j \subset \mathbb{X}_h$ and $\mathcal{F} \subset \mathbb{X}_h$ are defined as

$$\mathcal{B}^j := \{(X,b)\,|X \in \mathbb{X}_{free}, b \in B^j\} \tag{40}$$

$$\mathcal{F} := \{(X,b)\,|X \in F, b \in \mathbb{B}\} \tag{41}$$

$$\mathcal{S}^j := \mathcal{B}^j \cup \mathcal{F}, \quad \overline{\mathcal{S}}^j := \mathbb{X}_h \setminus \mathcal{S}^j \tag{42}$$

where $\mathcal{S}^j$ and $\overline{\mathcal{S}}^j$, respectively, denote the entire stopping region and transient region under the local controller μ^{ij}.

Absorption probability of local planners: If, under the dynamics induced by the local planner, the system reaches the target node $\mathcal{B}^j$, the local planner is considered to be successful; if the system hits an obstacle, the local planner is considered to have failed. The success probability of local planners (i.e. the absorption probability into FIRM nodes) is computed by solving the following integral equation that results from the law of total probability:

$$\begin{aligned}\mathbb{P}(\mathcal{B}^j|\mathcal{X},\mu^{ij}) &= \int_{\mathbb{X}_h} p^{\mu^{ij}}(\mathcal{X}'|\mathcal{X})\,\mathbb{P}(\mathcal{B}^j|\mathcal{X}',\mu^{ij})\,d\mathcal{X}' \\ &= \int_{\mathcal{B}^j} p^{\mu^{ij}}(\mathcal{X}'|\mathcal{X})\,d\mathcal{X}' \\ &\quad + \int_{\overline{\mathcal{S}}^j} p^{\mu^{ij}}(\mathcal{X}'|\mathcal{X})\,\mathbb{P}(\mathcal{B}^j|\mathcal{X}',\mu^{ij})\,d\mathcal{X}'\end{aligned} \tag{43}$$

where the second equality in Equation (43) follows from substituting the following conditions, inherited from FIRM construction, into the first integral:

$$\mathbb{P}(\mathcal{B}^j|\mathcal{X},\mu^{ij}) = \begin{cases} 1, & \text{if } \mathcal{X} \in \mathcal{B}^j \\ 0, & \text{if } \mathcal{X} \in \mathcal{F} \end{cases} \tag{44}$$

Henceforth, we drop indices i and j to unclutter expressions. Thus, we can write

$$\begin{aligned}\mathbb{P}(\mathcal{B}|\mathcal{X},\mu) &= \int_{\mathcal{B}} p^{\mu}(\mathcal{X}'|\mathcal{X})\,d\mathcal{X}' \\ &\quad + \int_{\overline{\mathcal{S}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\,\mathbb{P}(\mathcal{B}|\mathcal{X}',\mu)\,d\mathcal{X}' \\ &= R(\mathcal{X}) + \mathbf{T}_{\mathcal{S}}[\mathbb{P}(\mathcal{B}|\cdot,\mu)](\mathcal{X})\end{aligned} \tag{45}$$

where the operator $\mathbf{T}_{\mathcal{S}}$ and the function $R(\mathcal{X})$ are defined as

$$\begin{aligned}\mathbf{T}_{\mathcal{S}}[f(\cdot)](\mathcal{X}) &:= \int_{\overline{\mathcal{S}}} p^{\mu}(\mathcal{X}'|\mathcal{X}) f(\mathcal{X}')\,d\mathcal{X}', \\ R(\mathcal{X}) &:= \int_{\mathcal{B}} p^{\mu}(\mathcal{X}'|\mathcal{X})\,d\mathcal{X}'\end{aligned} \tag{46}$$

The solution of the integral equation in Equation (45) is expressed in the following as a Liouville–Neumann

series (Keener, 2000), similar to the solution of the inhomogeneous Fredholm equation of second type (Keener, 2000):

$$\mathbb{P}(\mathcal{B}|\mathcal{X},\mu)=\sum_{n=1}^{\infty}\mathbf{T}_{\mathcal{S}}^{n}\,[R(\cdot)\,](\mathcal{X}) \tag{47}$$

We show that the series in Equation (47) is a convergent series by resorting to the following assumption, which is a weaker version of the aforementioned FIRM condition on the design of nodes and local controllers.

Assumption 1. *We assume that there exists some time step N at which the controller stops with a positive probability. Mathematically, there exists an $N<\infty$ and a $\beta>0$ such that $\mathbb{P}_N(\mathcal{S}^j|\mathcal{X},\mu^{ij})\geq\beta>0$, for all $\mathcal{X}$.*

This assumption is almost always true, as it rephrases the role of the controller in driving the system toward the target region. For example, if we have Gaussian noise (as is the case in the SLQG-FIRM), the assumption is true at $N=1$ regardless of the utilized controller.

Lemma 4. *Given Assumption 1, we have*

$$\begin{cases}\|\mathbf{T}_{\mathcal{S}}^{n}\|_{op}\leq 1, & n<N\\ \|\mathbf{T}_{\mathcal{S}}^{n}\|_{op}\leq 1-\beta<1, & n\geq N\\ \sum_{n=0}^{\infty}\|\mathbf{T}_{\mathcal{S}}^{n}\|_{op}\leq c<\infty\end{cases} \tag{48}$$

Proof. See Appendix E. □

Corollary 1. *The series $\sum_{n=0}^{\infty}\mathbf{T}_{\mathcal{S}}^{n}[R]$ is a convergent series, and therefore we can define the resolvent operator $(I-\mathbf{T}_{\mathcal{S}})^{-1}[R]=\sum_{n=0}^{\infty}\mathbf{T}_{\mathcal{S}}^{n}[R]$, where $\|(I-\mathbf{T}_{\mathcal{S}})^{-1}\|_{op}\leq c<\infty$.*

Proof. See Appendix F. □

According to Corollary 1, the success probability of the local controller μ can be written, using the defined resolvent operator, as

$$\mathbb{P}(\mathcal{B}|\mathcal{X},\mu)=(I-\mathbf{T}_{\mathcal{S}})^{-1}\,[R(\cdot)\,](\mathcal{X}) \tag{49}$$

As the first result of this section (Proposition 1), we aim to show that this absorption probability varies continuously with respect to changes in parameters of the local planner. However, we will first state two assumptions.

Assumption 2. *We assume the local planning law and induced transition probabilities are smooth; in other words, we have assume the following.*

- *Local control laws are continuous in their parameters, that is, for the (i,j)th local controller, mapping $\mu^{ij}(\cdot;\mathbf{v}^j)\colon\ \mathbb{B}\rightarrow\mathbb{U}$ is a continuous function in its parameter $\mathbf{v}^j$.*
- *The transition pdf on h-state, that is, $p(\mathcal{X}'|\mathcal{X},u)$, is a continuous function of the control u; in other words, there exists a $c_1<\infty$ such that $\|p(\mathcal{X}'|\mathcal{X},u)-p(\mathcal{X}'|\mathcal{X},\check{u})\|\leq c_1\|u-\check{u}\|$.*

Finally, we state the following assumption, in which we emphasize the fact that, as $\mathbf{v}\rightarrow\check{\mathbf{v}}$, the transition probability induced by the local controller $\mu(\cdot;\mathbf{v})$ into the sets $\mathcal{B}$ and $\check{\mathcal{B}}$ also has to converge, which is a reasonable assumption for a smooth control law.

Assumption 3. *Consider the controllers $\mu(\cdot;\mathbf{v})$ and $\check{\mu}(\cdot;\check{\mathbf{v}})$, whose corresponding extended absorption regions are denoted by $\mathcal{B}$ and $\check{\mathcal{B}}$, respectively. We assume that there exist real numbers $r>0$ and $c'<\infty$ such that for $\|\mathbf{v}-\check{\mathbf{v}}\|\leq r$, we have*

$$\|\mathbb{P}_1(\mathcal{B}\ominus\check{\mathcal{B}}|\mathcal{X},\mu)\|\leq c'\|\mathbf{v}-\check{\mathbf{v}}\| \tag{50}$$

where $\ominus$ is the symmetric difference operator; in other words, $\mathcal{B}\ominus\check{\mathcal{B}}=(\mathcal{B}\setminus\check{\mathcal{B}})\cup(\check{\mathcal{B}}\setminus\mathcal{B})$.

Now we state the following proposition on the continuity of the success probability of local planners.

Proposition 1 (Continuity of absorption probabilities)**.** *Given Assumptions 1, 2, and 3, the absorption probability $\mathbb{P}(B^j|b,\mu^{ij})$ is continuous in parameter $\mathbf{v}^j$ for all i,j, and b.*

Proof. See Appendix G. □

Now we present the main result regarding the probabilistic completeness of FIRM-based methods.

Theorem 1. *Given Assumptions 1, 2, and 3, any planning algorithm under uncertainty that is generated based on the FIRM framework (i.e. guarantees belief node reachability and induces a roadmap in the belief space with independent edge costs) is PCUU.*

Proof. See Appendix H. □

The basic idea of probabilistic completeness under uncertainty stems from an idea similar to the one in the path-isolation-based analysis for planners in deterministic systems. Roughly speaking, in the path isolation argument for sampling-based planners in the absence of uncertainty, if there is a successful path and a non-zero neighborhood of this path, in which every path is successful, we can eventually find a path in this neighborhood by increasing the number of samples unboundedly. Similarly, in the presence of uncertainty, if there is a successful policy, it is parametrized by some parameters (set of PRM nodes, in FIRM). Thus, if there exists a non-zero measure neighborhood of these parameters, within which selected parameters lead to a successful policy, we can eventually reach a successful policy by increasing the number of samples unboundedly and choosing samples in the target neighborhoods.

8. Experimental results

In this section, we first illustrate theoretical results from the previous sections on a planar robot in a small 3D planning domain. Then, we present planning results on a larger 3D state space. Finally, we report the results of the method on

a dynamical model of an eight-arm manipulator (sixteen-degree-of-freedom state space). This section is followed by a brief comparison with other state-of-the-art methods in Section 9.

8.1. Planar 3D omnidirectional robot: Illustrating steps in construction and planning with SLQG-FIRM

In this subsection, we focus on an omni-directional robot. Its state is composed of its 2D position in the plane and its heading angle. The goal in this section is to illustrate the steps of constructing SLQG-FIRM and planning with it.

8.1.1. Motion model. A three-wheel omnidirectional mobile robot is used in experiments with the nonlinear kinematic model given in Kalmár-Nagy et al. (2004). The state vector is composed of a 2D location and heading angle $x = [{}^1x, {}^2x, \theta]^{\mathrm{T}}$ in a global world frame. Further, $u = [{}^1u, {}^2u, {}^3u]^{\mathrm{T}}$ is the vector of controls, where ${}^iu$ is the linear velocity of the ith wheel. The motion model for this robot, in its original continuous form, is (Kalmár-Nagy et al., 2004)

$$\dot{x} = \mathbf{f}_c(x, u, w) = T(x)\,u + w \tag{51}$$

where w is the motion noise, which is drawn from a zero-mean Gaussian distribution, and

$$T(x) = \begin{pmatrix} -\frac{2}{3}\sin(\theta) & -\frac{2}{3}\sin(\frac{\pi}{3}-\theta) & \frac{2}{3}\sin(\frac{\pi}{3}+\theta) \\ \frac{2}{3}\cos(\theta) & -\frac{2}{3}\cos(\frac{\pi}{3}-\theta) & -\frac{2}{3}\cos(\frac{\pi}{3}+\theta) \\ \frac{1}{3r} & \frac{1}{3r} & \frac{1}{3r} \end{pmatrix} \tag{52}$$

where r is the distance of the wheels from the robot's center of mass. The discrete motion model is shown by

$$x_k = \mathbf{f}(x_{k-1}, u_{k-1}, w_{k-1}) \tag{53}$$

where $w_k \sim \mathcal{N}(0, \mathbf{Q})$ is the motion noise at the kth time step, which is drawn from a zero-mean Gaussian distribution with covariance matrix $\mathbf{Q}$. It can be shown that if we linearize this system, the linearized motion model satisfies the controllability condition in Property 1.

8.1.2. Observation model. In experiments, the robot is equipped with exteroceptive sensors that provide range and bearing measurements from existing landmarks (radio beacons) in the environment. The 2D location of the jth landmark is denoted by L_j. Measuring L_j can be modeled as follows:

$$\begin{aligned} {}^jz &= {}^jh(x, {}^jv) \\ &= [\|{}^j\mathbf{d}\|, \mathrm{atan2}({}^jd_2, {}^jd_1) - \theta]^{\mathrm{T}} + {}^jv, \;\; {}^jv \sim \mathcal{N}(\mathbf{0}, {}^j\mathbf{R}) \end{aligned} \tag{54}$$

where ${}^j\mathbf{d} = [{}^jd_1, {}^jd_2]^{\mathrm{T}} := [{}^1x, {}^2x]^{\mathrm{T}} - L_j$. The vector ${}^jv$ is a state-dependent observation noise, with covariance

$${}^j\mathbf{R} = \mathrm{diag}\big((\eta_r\|{}^j\mathbf{d}\| + \sigma_b^r)^2, (\eta_\theta\|{}^j\mathbf{d}\| + \sigma_b^\theta)^2\big) \tag{55}$$

In other words, the uncertainty (standard deviation) of the sensor reading increases as the robot gets farther from the landmarks; $\eta_r = \eta_\theta = 0.3$ determines this dependence, and $\sigma_b^r = 0.01$ m and $\sigma_b^\theta = 0.5^\circ$ are the bias standard deviations. A similar model for range sensing is used in Prentice and Roy (2009). We assume the robot observes all N_L landmarks at all times and their observation noises are independent. Thus, the total measurement vector is denoted by $z = [{}^1z^{\mathrm{T}}, {}^2z^{\mathrm{T}}, \ldots, {}^{N_L}z^{\mathrm{T}}]^{\mathrm{T}}$, and, due to the independence of measurements of different landmarks, the observation model for all landmarks can be written as

$$z = h(x) + v, \;\; v \sim \mathcal{N}(\mathbf{0}, \mathbf{R}), \;\; \mathbf{R} = \mathrm{diag}({}^1\mathbf{R}, \ldots, {}^{N_L}\mathbf{R}) \tag{56}$$

It is straightforward to show that the linearized version of this observation model satisfies the observability condition in Property 1. Therefore, this entire system model (motion and sensing models) satisfies Property 1 and thus the SLQG-FIRM can be used for planning.

8.1.3. Construction of SLQG-FIRM nodes and edges. Figure 4(a) shows a sample environment, including obstacles, landmarks, and enumerated nodes in $({}^1x, {}^2x, \theta)$ space. Nodes are shown by blue triangles, which encode the position $({}^1x, {}^2x)$ and heading angle θ of the robot. Landmarks are shown by black stars. The corresponding FIRM nodes are computed and shown in Figure 4(b). All elements in Figure 4(b) are defined in $({}^1x, {}^2x, \theta)$ space but only the $({}^1x, {}^2x)$ portion of them is shown. Each $b_c^j \equiv (\mathbf{v}^j, P_s^j)$ is illustrated by a red dot representing $\mathbf{v}^j$, and a green ellipse representing the 3σ ellipse of covariance P_s^j. Each FIRM node B^j is a neighborhood around b_c^j. In the experiments, we define the node region using the component-wise version of Equation (18) to handle the error scale difference in position and orientation variables:

$$B^j = \{b \equiv (x, P) \mid \; |x - \mathbf{v}^j| \dot{<} \epsilon, |P - P_s^j| \dot{<} \Delta\} \tag{57}$$

where $|\cdot|$ and $\dot{<}$ stand for the absolute value and component-wise comparison operators, respectively. We also set $\epsilon = [0.07\text{ m}, 0.07\text{ m}, 1^\circ]^{\mathrm{T}}$ and $\Delta = \epsilon\epsilon^{\mathrm{T}}$ to quantify B^j. The projection of B^j onto the space of estimation mean, that is, $B_x^j = \{\widehat{x}^+ \;:\; |\widehat{x}^+ - \mathbf{v}^j| \dot{<} \epsilon\}$, is a neighborhood around $\mathbf{v}^j$, which is shown by a cyan rectangle centered at $\mathbf{v}^j$. Projection of B^j onto the space of estimation covariances, that is, $B_P^j = \{P \;:\; |P - P_s^j| \dot{<} \Delta\}$, is a neighborhood around P_s^j. However, in a 2D plot B_P^j cannot be shown due to its high dimension. Thus, we partially illustrate it only by two dashed green ellipses that represent 3σ covariances of $P_s^j - \Delta_d$ and $P_s^j + \Delta_d$, where Δ_d is the matrix Δ, whose off-diagonal elements are set to zero. For illustration purposes, both of these neighborhoods (i.e. B_x^j and B_P^j) are magnified 5x in Figure 4(b).

8.1.4. Transition costs and probabilities. After designing FIRM nodes and local controllers, the transition costs and

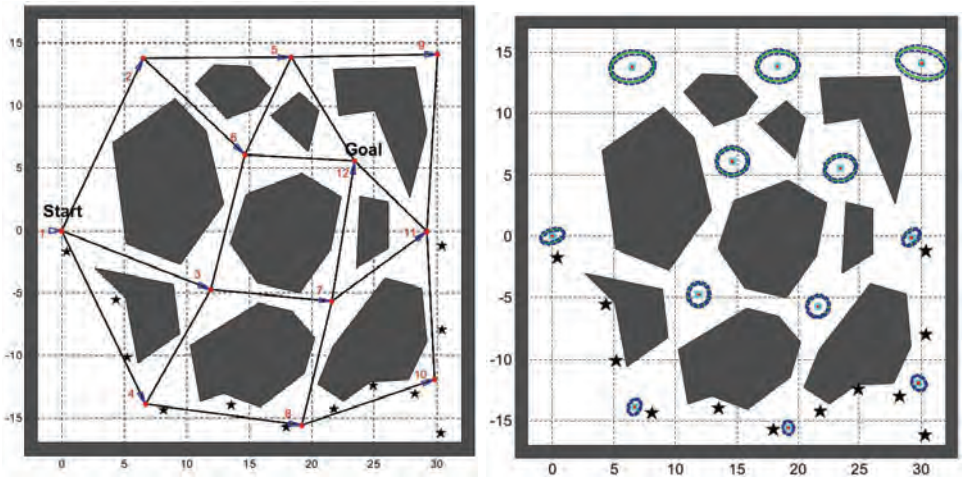


Fig. 4. (a) The underlying PRM graph. Gray polygons are the obstacles and black stars represent the landmarks' locations. (b) FIRM nodes corresponding to PRM nodes.

probabilities need to be computed. Based on the given task and needed accuracy, different approaches can be taken. Here, we use a particle-based approximation of the distribution to compute these quantities, where we use $M = 100$ particles. In other words, for every (B, μ) pair, we perform 100 runs. At every run, a sample path of state x, a sample path of estimation mean $\widehat{x}^+$, and a sample path of estimation covariance P is generated. If the filter of choice in the edge controller is the linearized Kalman filter (LKF) (Crassidis and Junkins, 2004; Simon, 2006), the covariance evolution is deterministic and there is no need to generate 100 different sample covariance paths. However, if the filter of choice in the edge controller is the EKF (Crassidis and Junkins, 2004; Simon, 2006), then we need to generate the sample covariance paths too, to take into account the stochasticity of the covariance matrix. Figure 5(a) depicts sample paths of the true state x and estimation mean $\widehat{x}^+$ in green and dark red, respectively, for $M = 100$ particles. Note that when a true state path (green path) collides with an obstacle, the process stops and failure happens. However, in this figure, for illustration purposes, we continue the process and ignore the obstacles to better show the uncertainty tube and information availability in different parts of the space. As seen in Figure 5(a), the behavior of the true state on the edges which have access to more accurate observations is remarkably close to the planned behavior. In contrast, on the edges that get less informative observations, the controller cannot effectively compensate for deviations of the ground truth from the nominal path, which can lead to collision with obstacles.

To avoid clutter, Figure 5(b) the depicts sample estimation covariance evolution only for a single particle. In this figure, we let the process and observation noise be zero, to keep the centers of the ellipses (i.e. estimation means) on the planned points. However, note that in general the estimation mean is affected by the noise (as is seen in Figure 5(a)). Indeed, Figure 5(b) can be seen as the maximum-likelihood estimation uncertainty tube over the roadmap.

Let us denote the qth sample path for the true state by $x^{(q)}_{0:\mathcal{T}^q}$, for the estimation mean by $\widehat{x}^{+(q)}_{0:\mathcal{T}^q}$, and for the estimation covariance by $P^{(q)}_{0:\mathcal{T}^q}$, where $\mathcal{T}^q$ is the stopping time of the qth particle in executing μ at B. Moreover, one can assign a weight to each particle q based on the probability of its occurrence. There are different ways proposed to compute these weights in the sequential Monte Carlo literature (Doucet et al., 2001). However, the main condition is that they have to sum to one, in other words, $\sum_{q=1}^{M} w^{(q)} = 1$. Here we simply consider $w^{(q)} = M^{-1}$. Note that if we particle μ^{ij} at B^i, all these quantities also need to have an ij superscript. Having these sample paths, we can compute the transition costs and probabilities associated with invoking μ^{ij} at B^i. For the collision probability, we have

$$\mathbb{P}^g(F|B^i, \mu^{ij}) = \mathbb{E}[\mathbb{I}_F|B^i, \mu^{ij}] \approx \sum_{q=1}^{M} w^{(q)} \mathbb{I}_F(x^{(q)}_{0:\mathcal{T}^q}) \quad (58)$$

$$\mathbb{P}^g(B^j|B^i, \mu^{ij}) = 1 - \mathbb{P}^g(F|B^i, \mu^{ij}) \quad (59)$$

where $\mathbb{I}_F$ is the failure indicator. $\mathbb{I}_F(x^{(q)}_{0:\mathcal{T}^q})$ is one if there exists a time step $k \leq \mathcal{T}^{(q)}$ such that $x_k \in F$; otherwise it is zero. Further, $\mathcal{T}^q$, or more rigorously $\mathcal{T}^{ij(q)}$, is the stopping time of the qth particle in executing μ^{ij} at B^i. To compute $\mathcal{T}^{ij(q)}$, we only need to check the condition $b \in B^j$ at every time step and find the first time step where belief b enters the stopping region B^j. Thus, we can compute the mean stopping time as

$$\widehat{\mathcal{T}}^{ij} = \mathbb{E}[\mathcal{T}^{ij}] \approx \sum_{q=1}^{M} w^{(q)} \mathcal{T}^{ij(q)} \quad (60)$$

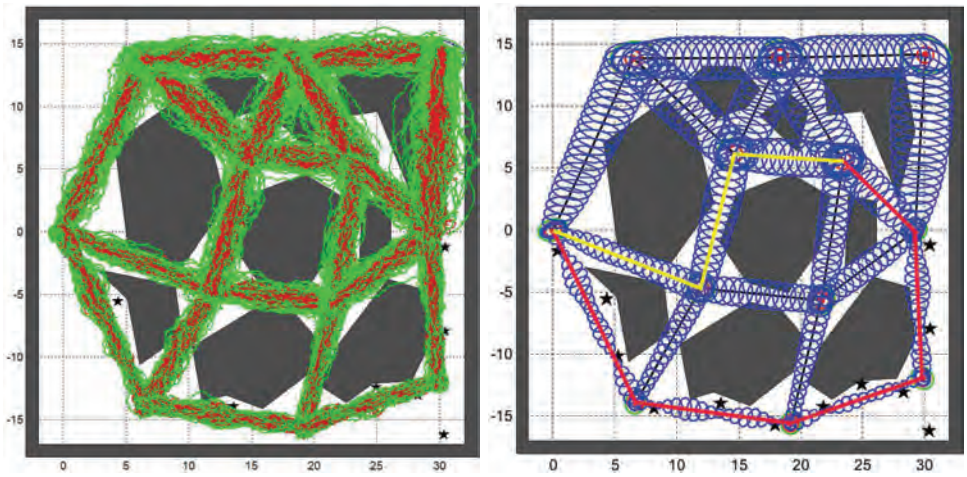


Fig. 5. Sample paths induced by controllers invoked at different nodes. (a) For $M = 100$ particles, sample ground truth paths and sample estimation mean paths are shown in green and dark red, respectively. (b) The most likely path under the optimal policy and the shortest path are shown in red and yellow respectively. The 3σ ML estimation uncertainty tube is drawn in blue.

To compute the filtering cost defined in Section 5.5, again we use the particle-based representation of belief,

$$\Phi^{ij} = \mathbb{E}\left[\sum_{k=1}^{\mathcal{T}^{ij}} \mathrm{tr}(P_k)\,|B^i, \mu^{ij}\right] \approx \sum_{q=1}^{M}\sum_{k=1}^{\mathcal{T}^q} w^{(q)}\mathrm{tr}(P_k^{(q)}) \quad (61)$$

where $P_k^{(q)}$ is the estimation covariance at the kth time step of the qth particle. Finally, the cost of taking μ^{ij} at B^j is as follows:

$$C^g(B^i, \mu^{ij}) = \alpha_1 \Phi^{ij} + \alpha_2 \widehat{\mathcal{T}}^{ij}$$

where we used the coefficients $\alpha_1 = 0.95$ and $\alpha_2 = 0.05$. Table 1 shows these quantities for several (B^i, μ^{ij}) pairs in FIRM corresponding to Figure 5.

8.1.5. Planning and replanning on FIRM. Plugging the computed transition costs and probabilities into Equation (32), we can solve the DP and compute the graph policy π^g. This process is performed once offline if the goal location is fixed. Figure 6(a) shows the policy π^g on the constructed FIRM in this example. Indeed, at every FIRM node B^i, the policy π^g decides which local controller has to be taken, which in turn aims to take the robot to the next FIRM node. Thus, the online part of the planning is significantly efficient and reduces to executing the controller and generating the control signal, which is almost an instantaneous computation.

Replanning: An important consequence of this framework is that replanning can be performed using FIRM efficiently. Suppose due to some unmodeled large disturbance, the robot's belief deviates significantly from the planned path; in other words, for some appropriate norm $\|\cdot\|$ on belief space we have $\|b_k - \mathbb{E}[b_k^p]\| > \varrho$, where b_k^p is the planned belief at the kth time step, and ϱ is the threshold for deciding if replanning is needed or not. In such cases, replanning occurs based on Algorithm 2. In Figure 6(b), we illustrate a simple replanning process. In this figure it is assumed that an unmodeled large disturbance affects the system such that the estimation mean significantly deviates from the planned path. The deviated mean is shown on the figure as the 'restart point'. Thus, based on Algorithm 2, we connect this point to PRM. In Figure 6(b) the newly added PRM edges (i.e. $\mathcal{E}(0)$) are shown by dashed green lines. Then, for every edge in $\mathcal{E}(0)$, we design a local controller. Call the set of newly constructed local controllers $\mathbb{M}(0)$. For every $\mu \in \mathbb{M}(0)$ compute corresponding transition costs and probabilities. Finally, according to Bellman's principle of optimality, we use the precomputed costs-to-go $J^g(\cdot)$ to decide which controller has to be taken at the 'restart point' using Equation (34). Taking this controller, the belief enters into a FIRM node, and from there again we can use the precomputed π^g to control the robot toward the goal region.

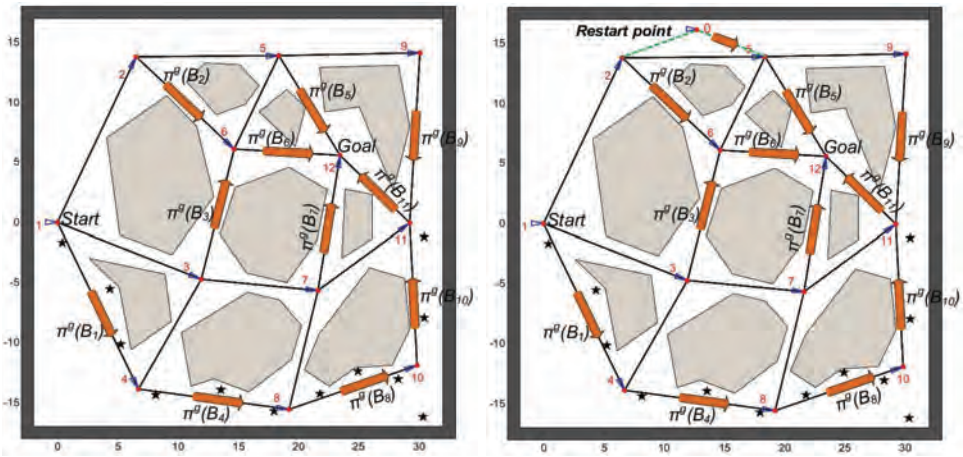
We show the most likely path under π^g in red in Figure 5(b). The shortest path is also illustrated in Figure 5(b) in yellow. It can be seen that the 'most likely path under the best policy' detours from the shortest path to a path along which the filtering uncertainty is smaller and it is easier for the controller to avoid collisions.

8.2. Larger environment

In this section, we consider the same omnidirectional robot with the same observation model, and we perform planning in a larger environment (shown in Figure 7) whose size is almost 10,000 m^2. Every grid square is a 10×10 area. The standard deviation of the process noise is assumed to be 1 m for the positional degrees of freedom and 7° for the angular degree of freedom. We start with a five-node FIRM and at every step we randomly sample five more nodes until

Table 1. Computed costs for several node-controller pairs in FIRM using 100 particles.

(B^i,μ^{ij}) pair	$B^1,\mu^{1,4}$	$B^4,\mu^{4,8}$	$B^8,\mu^{8,10}$	$B^{10},\mu^{10,11}$	$B^{11},\mu^{11,12}$	$B^1,\mu^{1,3}$	$B^3,\mu^{3,6}$	$B^6,\mu^{6,12}$
$\mathbb{P}^g(B^j\|B^i,\mu^{ij})$	97%	95%	99%	77%	79%	87%	55%	79%
Φ^{ij}	18.5967	11.2393	6.8229	15.1148	26.2942	23.6183	48.8189	43.6207
$\mathbb{E}[\mathcal{T}^{ij}]$	238.2	193.0	150.0	209.6	170.8	200.3	242.4	219.2
$\sigma[\mathcal{T}^{ij}]$	21.8	28.7	15.1	24.5	22.6	22.7	30.1	26.7



(a) (b)

Fig. 6. Planning and replanning on FIRM. (a) Policy π^g resulted from solving the DP in Equation (31) is shown by red arrows. Indeed, for every FIRM node, the policy π^g tells us which controller has to be taken. (b) In this figure it is assumed that an unmodeled large disturbance affects the system such that the estimation mean significantly deviates from the planned path. The deviated mean is denoted by 'restart point' in the figure.

we reach 500 nodes. Thus, overall, we construct 100 FIRM graphs in this environment, for each of which we measure the construction time (cumulative) and compute the success probability. Plots in Figure 8 show these quantities as a function of the number of nodes for a sample run on an Intel i5 dual-core 1.7 GHz machine with 4 GB memory. Further, 50 particles are used for collision-checking, and every node in the underlying PRM is connected to its three nearest neighbors.

Basically, FIRM construction is an anytime algorithm in the sense that one can increase the number of nodes and stop enlarging the graph when a termination condition is satisfied such as: (i) achieving a desirable success probability or a desirable cost-to-go, (ii) no change being observed in the success probability or in the cost-to-go for a significant time, or (iii) exceeding the maximum time allowed for offline computation.

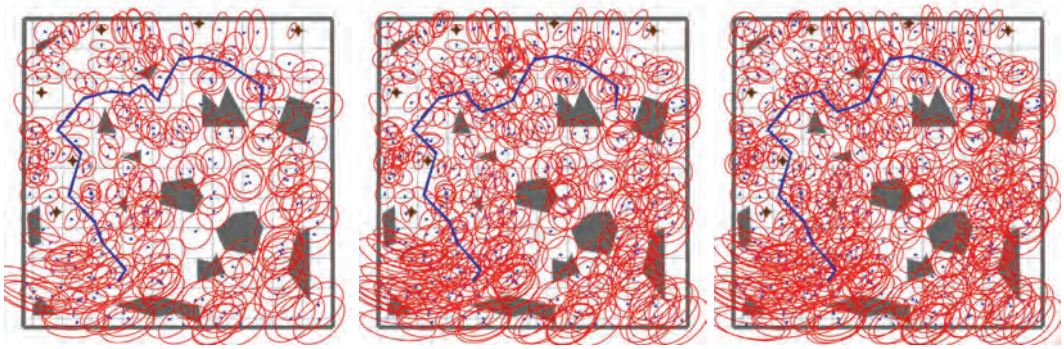
Again, as is seen in Figure 7, the highest-likelihood path under the optimal policy detours from the shortest path toward the more informative regions in the environment. As a result, it reduces the collision probability and at the same time increases the estimation accuracy and controller efficiency. However, it is important to note that the returned solution is not a single path, but it is a feedback law over the entire space. For the video of executing this plan (with fewer nodes to unclutter the video), see Extension 1.

We also conducted a simulation to illustrate the robustness of the method to large deviations. In this simulation, the robot is pushed away from the roadmap several times by some large disturbances, and replanning is performed online based on Algorithm 2. The video of this simulation is also available (see Extension 2).

8.3. Eight-arm manipulator

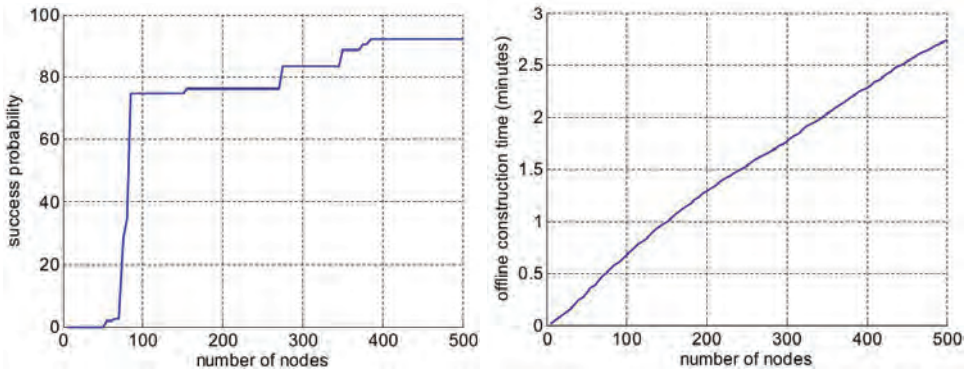
On a given graph, the number of paths between two given points grows exponentially with the size of graph. Thus, in the direct propagation of uncertainty on a roadmap, the number of edge costs and transition probabilities that need to be computed is exponential in the number of underlying PRM nodes (see Section 9 for a detailed analysis). As a result, when we deal with high-dimensional state spaces, where PRM needs to have many edges and nodes, it is not feasible to use the methods that perform direct uncertainty propagation. However, using FIRM, we only need to compute the costs and transition probabilities for as many edges as the underlying PRM has. Thus, we can easily increase the dimension to the level that PRM can handle, and the complexity of the algorithm is increased only by a constant factor (involving computation of costs and transition probabilities of a single edge). In the following experiment, we verify the effectiveness of FIRM in handling

(a) 50 nodes. (b) 75 nodes. (c) 105 nodes.



(d) 275 nodes. (e) 425 nodes. (f) 500 nodes.

Fig. 7. (a)–(f) Different snapshots of the roadmap for 50, 75, 105, 275, 425, and 500 nodes, respectively. The most likely paths under the optimal plan are also shown in blue. Stars show the landmarks. The means and covariances of the FIRM node centers are shown by small blue triangles and their associated red ellipses, respectively. Also, see Extensions 1 and 2 regarding the video of planning with FIRM in this environment.



(a) (b)

Fig. 8. A sample run showing (a) the success probability of the generated plan versus the number of nodes, as well as (b) the construction time (offline) for the plan.

high-dimensional systems through a simple example of an eight-arm manipulator. To the best of our knowledge, this is the first belief-space planner that can provide a plan over an entire roadmap for an eight-dimensional system while incorporating expensive costs in planning, such as computing collision probabilities. This experiment shows that FIRM can be used as a practical tool in many real-world problems.

8.3.1. Motion model. We consider an eight-arm manipulator with eight revolute joints in the plane. The state of the system is described by the angles of joints and their velocities $x = (\theta_1, \ldots, \theta_8, \dot{\theta}_1, \ldots, \dot{\theta}_8)^{\mathrm{T}}$, and the available control is considered to be the angular acceleration (or torque) of joints $u = (\alpha_1, \alpha_2, \ldots, \alpha_8)$. The process noise $w = (w_1, w_2, \ldots, w_8)$ is modeled as a zero-mean Gaussian noise on angular accelerations. Therefore, the continuous

motion model for every link is $\ddot{\theta}_i = \alpha_i + w_i$, whose discrete version for the entire state can be written as

$$x_{k+1} = Ax_k + Bu_k + Gw_k \tag{62}$$

where

$$A = \begin{pmatrix} I_8 & I_8\delta t \\ 0_8 & I_8 \end{pmatrix}, \quad B = \begin{pmatrix} 0_8 \\ I_8\delta t \end{pmatrix}, \quad G = \begin{pmatrix} 0_8 \\ I_8\sqrt{\delta t} \end{pmatrix} \tag{63}$$

where δt is the time interval between two consecutive time steps, and I_n and 0_n are the identity matrix and square zero matrix of dimension n, respectively.

8.3.2. Observation model. We use the light–dark environment setting as the observation model, which is also used in Platt et al. (2010, 2011). In the light–dark environment, the accuracy of sensory readings is encoded by a gray level, in which the regions that have access to more accurate sensory readings are lighter than the regions that do not have access to such informative sensory readings. In this experiment, we assume that we measure the state of the system, but this measurement is more accurate as we get closer to the left wall on which our sensor is mounted. (This model is adopted from Platt et al. (2010).) Thus, we have $z = h(x) = [z^1, \ldots, z^8]^T$, where

$$z^i = \theta_i + v^i, \quad v^i \sim \mathcal{N}(0, (\eta|\mathsf{x}^i - l| + \sigma_b)^2) \tag{64}$$

where x^i is the x coordinate of the ith joint location, l is the location of the vertical wall, η defines the dependency of the noise standard deviation on the distance from the wall, and σ_b is the bias standard deviation. Figure 9 shows an example of such an environment, in which $l = -1.5$, $\eta = 0.1$, and $\sigma_b = 10^{-4}$. The full observation model can be written as

$$z_k = h(x_k) = Hx_k + Mv_k \tag{65}$$

where $H = [I_8, 0_8]$ and $M = I_8$.

8.3.3. Sampling stabilizer parameters. The described system is a controllable and observable system, and thus we adopt the SLQG controller as the stabilizing controller. Therefore, the parameters of the controller are points in the equilibrium space, as explained in Section 5. In other words, to generate sample nodes in the state space, we need to sample the configuration space $(\theta_1, \ldots, \theta_8)$ and append zero angular velocities to it. To connect these samples in the state space we design simple trajectories between nodes, along which we accelerate the joints (angles) with constant acceleration until they are halfway to the next node, and after which we decelerate the joints until they reach the next node.

8.3.4. Construction of the SLQG-FIRM and planning with it. First, corresponding to sampled nodes in the state space, we compute corresponding FIRM nodes and then design local controllers according to Algorithm 1. In a similar procedure to the one in the previous experiment, we compute the transition costs and probabilities.

To solve the DP, we need to characterize the goal nodes. In Figure 9, the goal region for the tip location of the manipulator is shown by a purple circle. We mark all PRM samples whose tip locations are within the goal region as goal nodes. Setting the cost-to-go to zero for all goal nodes, we solve the DP and compute the optimal feedback on the graph according to Algorithm 1. Finally, we execute the plan based on Algorithm 2 and we illustrate the propagation of the covariance of the manipulator tip in Figure 9 in red. As can be seen in Figure 9, there are two passages among the obstacles to reach the goal region. Although the right passage is closer to the initial configuration of the manipulator, the manipulator detours to a longer path through the left passage, because there is more accurate sensory information available in the left passage than the right one. As is seen in this example, the feedback plan minimizes the collision probability and picks the safest path, while being robust to deviations. In other words, if for any reason the manipulator deviates significantly from the underlying PRM, the feedback plan connects the deviated belief to the best neighboring FIRM node in real time, and continues the pre-computed plan from this node.

9. Comparison and limitations

In this section, we perform a short comparison of SLQG-FIRM against the two most related methods in the literature: BRM (Prentice and Roy, 2009) and LQG-MP on roadmaps (Van den Berg et al., 2011). Both methods are belief-space planners that exploit roadmap-based ideas. We compare the methods in terms of the offline construction and online planning complexity, and also in terms of some other properties, all listed in Table 2. In the following, we go over the complexity analysis that leads to the entries in this table. Afterwards, we discuss limitations of the SLQG-FIRM.

Offline construction complexity: In a general graph, the number of paths between two given nodes is exponential in the number of nodes N. For example, if each node in a graph is connected to k nearest neighbor nodes on the graph, for a search depth of d edges on the graph, the corresponding search tree contains k^d paths. Notice that each of these paths has d edges on it. Thus, if we directly (without using belief stabilizers) propagate the uncertainty on a roadmap for a depth of d, we have to evaluate the cost on dk^d edges. So, the asymptotic complexity of the overall problem is of the order $\mathcal{O}(Nk^N)$. Now, if computing the cost and transition probabilities associated with each edge under uncertainty is a constant multiplier $\mathcal{O}(c)$ of computing its cost in a deterministic case, the overall complexity of the methods based on direct belief propagation is $\mathcal{O}(cNk^N)$. On the other hand, in any variant of FIRM, due to the edge independence, only the cost of $\mathcal{O}(Nk)$ edges needs to be constructed as in PRM,

Table 2. Belief space roadmap-based method comparison (without using a heuristic in search algorithms).

Algorithm	Offline construction complexity (no heuristic)	Replanning (online planning) complexity	Future observations	System requirement	Valid region of plan	Collision probabilities
Generic PRM	$\mathcal{O}(Nk)$	$\mathcal{O}(k)$	———	Assumes a controller exists to drive the system from node to node	On the graph only	———
BRM	$\mathcal{O}(cNk^N)$	$\mathcal{O}(c\frac{N}{l}k^N)$ or $\mathcal{O}(cNk^N)$	Maximum likelihood observation	Well linearizable systems	Vicinity of the nominal path	Not considered
LQG-MP on roadmaps	$\mathcal{O}(cNk^N)$	$\mathcal{O}(cNk^N)$	All observations	Well linearizable systems	Vicinity of the nominal path	Simplified measures are used
Generic FIRM	$\mathcal{O}(cNk)$	$\mathcal{O}(ck)$	———	Assumes a controller exists to drive the system from node to node	Union of convergence regions of local controllers	———
SLQG-FIRM	$\mathcal{O}(cNk)$	$\mathcal{O}(ck)$ or $\mathcal{O}(1)$	All observations	Well linearizable, and linear controllable and observable systems	Vicinity of whole PRM (entire space for a dense PRM)	Computed

and thus the overall complexity of offline construction of FIRM is $\mathcal{O}(cNk)$.

Online planning (replanning) complexity: If the system deviates from the valid region of the plan, in direct propagation methods, edge costs need to be recomputed for all edges. So, in BRM and LQG-MP on roadmaps, the replanning complexity will be of the order $\mathcal{O}(Nk^N)$. If the cost of each edge is defined in such a way that it only depends on the belief at the start and end of the edge (i.e. does not depend on the belief along the edge), BRM can reduce the computation complexity to $\mathcal{O}(c(N/l)k^N)$ through covariance factorization techniques, where l is assumed to be the length (number of steps) of each edge. In FIRM, in the case of replanning (submitting a query with new starting point), it is only necessary to connect the deviated belief to k neighboring FIRM nodes. Thus, we only need to compute the cost for the k new edges. It is worth noting that if the underlying PRM is dense enough that the valid region of the local controllers covers the space, edge-cost computation in the replanning phase reduces to zero, because if the system deviates out of a valid region of a local planner, it will fall into the valid region of some other planner.

To reduce the complexity of the search algorithm in BRM and LQG-MP on roadmaps, it is assumed that the costs on different edges of the roadmap are independent. This heuristic can reduce the complexity of the algorithm, but it may still be significantly high compared to the PRM or FIRM. Moreover, this heuristic (edge-independent assumption) is not true without having belief stabilizers, and thus search algorithms relying on such a heuristic may result in solutions arbitrarily different from the true solution of the search algorithm. Assuming that no such heuristic is used in the search algorithm, Table 2 summarizes the complexity of these algorithms.

The huge reduction in the computational complexity of the planning algorithm (in particular, in the online phase) opens many possibilities in utilizing POMDP solvers in real-world applications. Moreover, due to its sampling-based nature, it ameliorates the curse of dimensionality just as PRM does in the deterministic case. In other words, if the dimension of the system increases, we need a greater number of nodes N in the underlying PRM to capture the free space connectivity, in which case we cannot use direct methods due to their complexity. However, FIRM can tolerate the increase in the dimension since its complexity is only a constant multiplier of the PRM complexity.

9.1. *Limitations of the SLQG-FIRM and future directions*

In this section, we discuss limitations of the proposed method. It is important to distinguish which limitation is associated with the generic FIRM framework, and which limitation is associated with the particular presented instantiation of the FIRM, that is, the SLQG-FIRM. In some cases, we also propose ways to remedy these limitations as future research directions.

Stabilization time: The FIRM framework introduces the usage of belief stabilizers. However, the time needed for the belief stabilization procedure is added to the overall execution time. If the number of time steps along the nominal path is l, and the number of time steps needed for stabilization is τ, the extra time τ is usually negligible compared

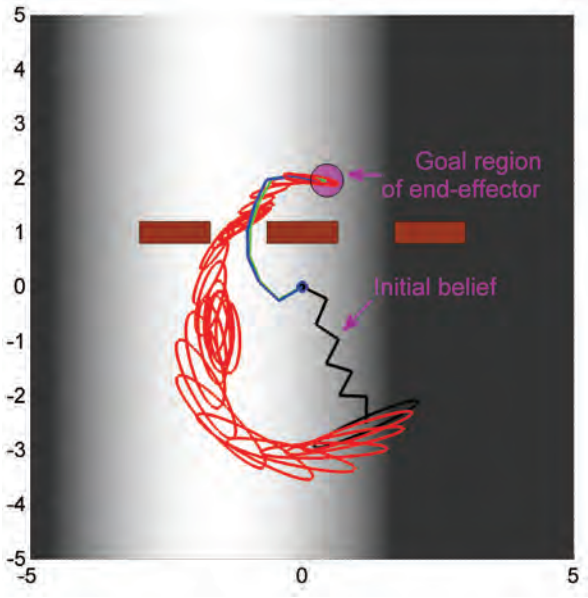


Fig. 9. A result of executing the FIRM plan for an eight-arm manipulator in a light–dark (sensing) environment. The manipulator is attached to the origin $(0, 0)$ and the purple region is the goal region for the manipulator tip. To unclutter the figure, we only show the uncertainty of the manipulator tip (end-effector). The initial mean and covariance are shown in black, and the evolution of the tip covariance during the plan execution is shown in red. The final estimation mean and the true configuration of the manipulator are shown in blue and green, respectively. Obstacles are shown in brown. The manipulator follows a longer but safer path to the goal region through the left passage, compared to the shorter but risky (with high collision probability) path through the right passage.

to l. However, τ can increase as the sensing uncertainty increases. In such a situation, one can consider two cases: if obstacles are close to the robot, it is indeed unsafe to move with a poor estimate, and it is indeed better to lose some time to gain more information, and then start moving. On the other hand, if there is no obstacle close to the robot, then one can increase the size of the corresponding FIRM node, and thus decrease the extra stopping time. Moreover, efficient sampling-based methods, which are aware of available information at different locations of the environment, and thus aware of the mean stabilization time, can be used to efficiently sample the nodes in the locations with lower mean stabilization times. These issues open up new directions for future research. However, if an application is very sensitive to the extra time, FIRM may not be a good choice for it, and methods such as BRM or LQG-MP can result in better guarantees on execution time.

Controllability and observability: As mentioned in Section 5, SLQG-FIRM works for systems that satisfy Property 1, which basically requires the linearized system about the PRM nodes to be controllable and observable. Although this includes a large class of systems, it excludes some important systems, such as non-holonomic systems that are not linearly controllable about any point. It is worth noting that this is not a limitation of the generic FIRM framework, but a limitation of the SLQG-FIRM. More recent instantiations of FIRM, such as PLQG-based FIRM (Agha-mohammadi et al., 2012c) or DFL-based FIRM (Agha-mohammadi et al., 2012a), aim to relax the controllability requirements in Property 1 and thus can include non-holonomic systems as well. However, relaxing the observability assumption is still an open problem.

Gaussian beliefs: The reachability argument in the SLQG-FIRM is restricted to Gaussian beliefs. In other words, we cannot guarantee reachability to some predefined non-Gaussian beliefs with SLQG controllers. This issue is a subject of future research.

Increasing the uncertainty: Although it may rarely happen in practice, it is possible to have a situation that leads to an uncertainty growth during the belief-stabilization process. However, this issue can be addressed easily. Notice that FIRM nodes are known a priori. Thus, at the beginning of each stabilization procedure, we can compare the current belief with the stationary belief of the stabilizer. If the current belief has more information than the stationary belief (e.g. if all eigenvalues of the estimation covariance are strictly less than the corresponding eigenvalues of the stationary estimation covariance), we replan from the current belief based on Algorithm 2. Therefore, uncertainty will not be increased during the stabilization procedure.

Locally linearizable systems: If a linear representation of the system of interest cannot be obtained (e.g. if the system state lives in a discrete set of states), the class of methods that use the linearized system as a local approximation of the true system will not work. In this case, another class of methods can be adopted which can handle these systems much better, such as those in Smith and Simmons (2005), Kurniawati et al. (2008), and Kurniawati et al. (2011). Coming up with belief stabilizers that work in discrete state space settings to design a discrete-state variant of FIRM is also an area for future research.

Velocity reduction in dynamical systems: To apply SLQG-FIRM to dynamical systems, the underlying PRM samples need to be selected from the equilibrium space, in other words, they need to have zero velocity. As a result a reduction in the system's velocity is expected while trying to reach the FIRM nodes. However, in many applications, reducing the speed at nodes to gain the robustness, reliability, and scalability offered by FIRM may be a useful trade-off. Nevertheless, this reduction in speed may not be desirable for some applications where the system cannot (or should not) decrease its velocity. For such systems, Agha-mohammadi et al. (2013a) propose a FIRM variant based on periodic controllers which does not require a reduction in the system's velocity. However, designing more efficient variants of FIRM that can sample points with non-zero velocities without introducing periodicity in the system's motion is an interesting future research direction.

10. Conclusion

In this paper, we have proposed the FIRM framework for solving the motion-planning problem under motion and sensing uncertainties. This problem is originally a POMDP, whose solution is computationally intractable. Exploiting feedback controllers, we reduced it to a tractable FIRM MDP that can be solved using standard DP techniques. FIRM utilizes feedback controllers to create reachable node regions in belief space. An important consequence is that FIRM preserves the optimal substructure property on the roadmap and thus overcomes the curse of history in the original POMDP problem. Finally, by computing the collision probabilities, obstacles are also appropriately taken into account in planning on FIRM. We showed an instantiation of the abstract FIRM framework using SLQG controllers and illustrated the construction and planning results on it. By extending the probabilistic completeness concept to planners under uncertainty, we also showed that FIRM is probabilistically complete under uncertainty. We believe that FIRM provides an important step toward solving POMDPs and utilizing them as a practical tool for robot motion planning under uncertainty.

Acknowledgements

The authors are grateful to the anonymous reviewers for their helpful suggestions as well as Aditya Mahadevan and Daniel Tomkins for fruitful discussions and their help with experiments.

Funding

This work was supported in part by the NSF (award number RI-1217991). In addition, the work of Agha-mohammadi and Chakravorty is supported in part by the AFOSR (grant FA9550-08-1-0038), and the work of Agha-mohammadi and Amato is supported in part by the NSF (award numbers CNS-0551685, CCF-0833199, CCF-0830753, IIS-0917266, IIS-0916053 and EFRI-1240483), by the NSF/DNDO (award number 2008-DN-077-ARI018-02), by the NIH (award number NCI R25 CA090301-11), by the DOE (award numbers DE-FC52-08NA28616, DE-AC02-06CH11357, B575363 and B575366), by the THECB (NHARP award number 000512-0097-2009), by Samsung, by Chevron, by IBM, by Intel, by Oracle/Sun, and by the King Abdullah University of Science and Technology (award number KUS-C1-016-04).

References

Agha-mohammadi A, Chakravorty S and Amato N (2011) FIRM: Feedback controller-based Information-state RoadMap – a framework for motion planning under uncertainty. In: *International conference on intelligent robots and systems (IROS)*.

Agha-mohammadi A, Chakravorty S and Amato N (2012a) Nonholonomic motion planning in belief space via dynamic feedback linearization-based FIRM. In: *International conference on intelligent robots and systems (IROS)*.

Agha-mohammadi A, Chakravorty S and Amato N (2012b) On the probabilistic completeness of the sampling-based feedback motion planners in belief space. In: *IEEE international conference on robotics and automation (ICRA)*.

Agha-mohammadi A, Chakravorty S and Amato N (2012c) Periodic-feedback motion planning in belief space for nonholonomic and/or non-stoppable robots. Technical Report no. TR12-003, Texas A&M University, USA.

Agha-mohammadi A, Chakravorty S and Amato N (2013a) Online replanning in belief space for dynamical systems: Towards handling discrete changes of goal location. In: *IEEE international conference on robotics and automation (ICRA): Workshop on combining task and motion planning*.

Agha-mohammadi A, Chakravorty S and Amato N (2013b) Sampling-based stochastic control with constraints: A unified approach in state and information spaces. In: *The American control conference (ACC)*.

Agha-mohammadi A, Agarwal S, Mahadeval A, et al. (2013c) Dynamic real-time replanning in belief space: an experimental study on physical mobile robots. Technical report TR 13-007, Texas A&M University, USA.

Alterovitz R, Siméon T and Goldberg K (2007) The stochastic motion roadmap: A sampling framework for planning with Markov motion uncertainty. In: *Proceedings of robotics: Science and systems (RSS)*.

Amato N, Bayazit B, Dale L, et al. (1998) OBPRM: An obstacle-based PRM for 3D workspaces. In: *International workshop on the algorithmic foundations of robotics*, pp. 155–168.

Astrom K (1965) Optimal control of Markov decision processes with incomplete state estimation. *Journal of Mathematical Analysis and Applications* 10: 174–205.

Bai H, Hsu D, Lee WS, et al. (2010) Monte Carlo value iteration for continuous-state POMDPs. In: Hsu D, Isler V, Latombe J-C, et al. (eds) *Algorithmic Foundations of Robotics IX* (*Springer Tracts in Advanced Robotics*, vol. 68). New York, NY: Springer, pp. 175–191.

Bertsekas D (1976) *Dynamic Programming and Stochastic Control*. Waltham, MA: Academic Press.

Bertsekas D (2007) *Dynamic Programming and Optimal Control*, 3rd edn. Cambridge, MA: Athena Scientific.

Bohlin R (2002) *Robot path planning*. PhD Thesis, Chalmers University of Technology, Sweden.

Bry A and Roy N (2011) Rapidly-exploring random belief trees for motion planning under uncertainty. In: *International conference on robotics and automation (ICRA)*, pp. 723–730.

Censi A, Calisi D, Luca AD, et al. (2008) A Bayesian framework for optimal motion planning with uncertainty. In: *Proceedings of the IEEE international conference on robotics and automation (ICRA)*, Pasadena, CA.

Chakravorty S and Erwin RS (2011) Information space receding horizon control. In: *IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL)*.

Chakravorty S and Kumar S (2009) Generalized sampling based motion planners with application to nonholonomic systems. In: *Proceedings of the IEEE international conference on systems, man, and cybernetics*, San Antonio, TX.

Chakravorty S and Kumar S (2011) Generalized sampling-based motion planners. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 41(3): 855–866.

Choset H, Lynch KM, Hutchinson S, et al. (2005) *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA: MIT Press.

Cormen TH, Leiserson CE, Rivest RL, et al. (2001) *Introduction to Algorithms*. 2nd edn. Cambridge, MA: MIT Press.

Crassidis J and Junkins J (2004) *Optimal Estimation of Dynamic Systems*. Boca Raton, FL: Chapman & Hall/CRC.

Doucet A, de Freitas J and Gordon N (2001) *Sequential Monte Carlo Methods in Practice*. New York, NY: Springer.

Guibas L, Hsu D, Kurniawati H, et al. (2008) Bounded uncertainty roadmaps for path planning. In: *International workshop on algorithmic foundations of robotics (WAFR)*.

He R, Brunskill E and Roy N (2010) PUMA: Planning under uncertainty with macro-actions. In: *Proceedings of the twenty-fourth conference on artificial intelligence (AAAI)*, Atlanta, GA.

He R, Brunskill E and Roy N (2011) Efficient planning under uncertainty with macro-actions. *Journal of Artificial Intelligence Research* 40: 523–570.

Hsu D (2000) *Randomized single-query motion planning in expansive spaces*. PhD Thesis, Stanford University, CA.

Huynh V and Roy N (2009) icLQG: Combining local and global optimization for control in information space. In: *IEEE international conference on robotics and automation (ICRA)*.

Kaelbling LP, Littman ML and Cassandra AR (1998) Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101: 99–134.

Kalmár-Nagy T, D'Andrea R and Ganguly P (2004) Near-optimal dynamics trajectory generation and control of an omnidirectional vehicle. *Robotics and Autonomous Systems* 46(1): 47–64.

Karaman S and Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research* 30(7): 846–894.

Kavraki L, Kolountzakis M and Latombe J (1998) Analysis of probabilistic roadmaps for path planning. *IEEE Transactions on Robotics and Automation* 14: 166–171.

Kavraki L, Latombe J, Motwani R, et al. (1995) Randomized query processing in robot motion planning. In: *Proceedings of the ACM symposium on the theory of computing*, pp. 353–362.

Kavraki L, Švestka P, Latombe J, et al. (1996) Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation* 12(4): 566–580.

Keener JP (2000) *Principles of Applied Mathematics: Transformation and Approximation*. 2nd edn. Boulder, CO: Westview Press.

Kumar PR and Varaiya PP (1986) *Stochastic Systems: Estimation, Identification, and Adaptive Control*. Englewood Cliffs, NJ: Prentice-Hall.

Kurniawati H, Bandyopadhyay T and Patrikalakis N (2012) Global motion planning under uncertain motion, sensing, and environment map. *Autonomous Robots* 33(3): 255–272.

Kurniawati H, Du Y, Hsu D, et al. (2010) Motion planning under uncertainty for robotic tasks with long time horizons. *The International Journal of Robotics Research* 30: 308–323.

Kurniawati H, Du Y, Hsu D, et al. (2011) Motion planning under uncertainty for robotic tasks with long time horizons. *The International Journal of Robotics Research* 30(3): 308–323.

Kurniawati H, Hsu D and Lee W (2008) SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In: *Proceedings of robotics: Science and systems (RSS)*.

Ladd A and Kavraki L (2004) Measure theoretic analysis of probabilistic path planning. *IEEE Transactions on Robotics and Automation* 20(2): 229–242.

Lavalle S and Kuffner J (2001) Randomized kinodynamic planning. *The International Journal of Robotics Research* 20: 378–400.

Lozano-Perez T (1983) Spatial planning: A configuration space approach. *IEEE Transactions on Computers* 100(2): 108–120.

Madani O, Hanks S and Condon A (1999) On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In: *Proceedings of the sixteenth conference on artificial intelligence (AAAI)*, pp. 541–548.

Melchior N and Simmons R (2007) Particle RRT for path planning with uncertainty. In: *International conference on intelligent robots and systems (IROS)*.

Meyn S and Tweedie RL (2009) *Markov Chains and Stochastic Stability*, 2nd edn. Cambridge: Cambridge University Press.

Missiuro P and Roy N (2006) Adapting probabilistic roadmaps to handle uncertain maps. In: *International conference on robotics and automation (ICRA)*.

Nakhaei A and Lamiraux F (2008) A framework for planning motions in stochastic maps. In: *IEEE international conference on robotics and automation (ICRA)*.

Norris JR (1997) *Markov Chains*. Cambridge: Cambridge University Press.

Ong SCW, Png SW, Hsu D, et al. (2010) Planning under uncertainty for robotic tasks with mixed observability. *The International Journal of Robotics Research* 29(8): 1053–1068.

Papadimitriou C and Tsitsiklis JN (1987) The complexity of Markov decision processes. *Mathematics of Operations Research* 12(3): 441–450.

Patil S, van den Berg J and Alterovitz R (2012) Estimating probability of collision for safe planning under Gaussian motion and sensing uncertainty. In: *IEEE international conference on robotics and automation (ICRA)*.

Pineau J, Gordon G and Thrun S (2003) Point-based value iteration: An anytime algorithm for POMDPs. In: *International joint conference on artificial intelligence*, pp. 1025–1032.

Pineau J, Gordon G and Thrun S (2006) Anytime point based approximations for large POMDPs. *Journal of Artificial Intelligence Research* 27: 335–380.

Platt R, Kaelbling L, Lozano-Perez T, et al. (2011) Efficient planning in non-Gaussian belief spaces and its application to robot grasping. In: *Proceedings of the international symposium of robotics research (ISRR)*.

Platt R, Tedrake R, Kaelbling L, et al. (2010) Belief space planning assuming maximum likelihood observatoins. In: *Proceedings of robotics: Science and systems (RSS)*.

Porta JM, Vlassis N, Spaan MTJ, et al. (2006) Point-based value iteration for continuous POMDPs. *Journal of Machine Learning Research* 7: 2329–2367.

Prentice S and Roy N (2009) The belief roadmap: Efficient planning in belief space by factoring the covariance. *The International Journal of Robotics Research* 28(11–12): 1448–1465.

Simon D (2006) *Optimal State Estimation: Kalman, H-infinity, and Nonlinear Approaches*. New York, NY: John Wiley and Sons.

Smallwood RD and Sondik EJ (1973) The optimal control of partially observable Markov processes over a finite horizon. *Operations Research* 21(5): 1071–1088.

Smith T and Simmons R (2005) Point-based POMDP algorithms: Improved analysis and implementation. In: *Proceedings of the international conference on uncertainty in artificial intelligence (UAI)*.

Sniedovich M (2006) Dijkstra's algorithm revisited: The dynamic programming connexion. *Control and Cybernetics* 35(3): 599–620.

Spaan M and Vlassis N (2005) Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research* 24: 195–220.

Sutton R, Precup D and Singh S (1999) Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112: 181–211.

Švestka P and Overmars M (1997) Motion planning for car-like robots using a probabilistic learning approach. *The International Journal of Robotics Research* 16(2): 119–143.

Thrun S, Burgard W and Fox D (2005) *Probabilistic Robotics*. Cambridge, MA: MIT Press.

Toit ND and Burdick JW (2010) Robotic motion planning in dynamic, cluttered, uncertain environments. In: *International conference on robotics and automation (ICRA)*.

Van den Berg J, Abbeel P and Goldberg K (2010) LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. In: *Proceedings of robotics: Science and systems (RSS)*.

Van den Berg J, Abbeel P and Goldberg K (2011) LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *The International Journal of Robotics Research* 30(7): 895–913.

Van den Berg J and Overmars M (2007) Kinodynamic motion planning on roadmaps in dynamic environments. In: *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 4253–4258.

Van den Berg J, Patil S and Alterovitz R (2011) Motion planning under uncertainty using differential dynamic programming in belief space. In: *Proceedings of the international symposium of robotics research (ISRR)*.

Van den Berg J, Patil S and Alterovitz R (2012) Efficient approximate value iteration for continuous Gaussian POMDPs. In: *Proceedings of the AAAI conference on artificial intelligence (AAAI)*.

Vitus MP and Tomlin CJ (2011) Closed-loop belief space planning for linear, Gaussian systems. In: *International conference on robotics and automation (ICRA)*, pp. 2152–2159.

Appendices

A. Index to Multimedia Extensions

The multimedia extension page is found at http://www.ijrr.org

Table of Multimedia Extensions

Extension	Type	Description
1	Video	Executing the FIRM plan in the environment shown in Figure 7
2	Video	Real-time replanning with FIRM, which shows the robustness of the method to large disturbances

B. Time-varying LQG controller

The time-varying LQG controller is often used to track a pre-planned trajectory (also called a nominal, desired, or open-loop trajectory) in the presence of process and observation noise. In principal it is designed (and optimal) for linear systems with Gaussian noise, but it can also be utilized for stabilizing nonlinear systems locally around the planned trajectory. An LQG controller is composed of a KF as an estimator and an LQR as a controller. At every time step k, the KF provides the *a posteriori* distribution (belief) b_k over the system state, and the LQR generates control u_k based on b_k.

In this appendix, we first discuss the system linearization and planned nominal trajectory, and then discuss the KF, LQR, and LQG corresponding to this nominal trajectory. Consider the nonlinear partially observable state-space equations of the system as follows:

$$x_{k+1} = f(x_k, u_k, w_k), \quad w_k \sim \mathcal{N}(0, Q_k) \tag{66a}$$

$$z_k = h(x_k, v_k), \quad v_k \sim \mathcal{N}(0, R_k) \tag{66b}$$

A planned nominal trajectory for this system is a sequence of planned states $(x_k^p)_{k\geq 0}$ and planned controls $(u_k^p)_{k\geq 0}$ such that it is consistent with the noiseless dynamics model; in other words, we have

$$x_{k+1}^p = f(x_k^p, u_k^p, 0) \tag{67}$$

The planned trajectory can be a finite sequence of some length N. The role of a closed-loop stochastic controller, during the trajectory tracking, is to compensate for the robot's deviations from the planned trajectory and to keep the robot close to the planned trajectory in the sense of minimizing the following quadratic cost:

$$J = \mathbb{E}\left[\sum_{k\geq 0} (x_k - x_k^p)^{\mathrm{T}} W_x (x_k - x_k^p) + (u_k - u_k^p)^{\mathrm{T}} W_u (u_k - u_k^p)\right] \tag{68}$$

where W_x and W_u are positive-definite weight matrices for the state and control costs, respectively.

Since the state space is not fully observable and is only partially observable, we do not have access to the perfect value of the state x_k, and thus, we provide the estimate x_k^+ of the state x_k based on the available observations $z_{0:k}$ from the beginning up to the current time step. Then, based on the separation principle (Kumar and Varaiya, 1986; Bertsekas, 2007), it can be shown that in a linear system with Gaussian noise, the above minimization in terms of the error $x_k - x_k^p$ is equivalent to performing two separate minimizations based on the estimation error $x_k - \widehat{x}_k^+$ and the controller error $\widehat{x}_k^+ - x_k^p$, whose summation is the same as the original main error $x_k - x_k^p = (x_k - \widehat{x}_k^+) + (\widehat{x}_k^+ - x_k^p)$, where $\widehat{x}_k^+ = \mathbb{E}[x_k^+] = \mathbb{E}[x_k | z_{0:k}]$. As a major consequence, the design of the stochastic controller with a partially observable state space (LQG) reduces to designing a controller

with a fully observable state (LQR) and designing an estimator (KF), separately. In the following, we first discuss the linearization of a nonlinear model. Then we discuss how a KF and an LQR can be designed for this linearized system. Finally, we combine them to construct a time-varying LQG controller.

Model linearization: Given a nominal trajectory $(x_k^p, u_k^p)_{k\geq 0}$, we linearize the dynamics and observation model in Equation (66) as follows:

$$x_{k+1} = f(x_k^p, u_k^p, 0) + A_k(x_k - x_k^p) + B_k(u_k - u_k^p) + G_k w_k, \quad w_k \sim \mathcal{N}(0, Q_k) \tag{69a}$$

$$z_k = h(x_k^p, 0) + H_k(x_k - x_k^p) + M_k v_k, \quad v_k \sim \mathcal{N}(0, R_k) \tag{69b}$$

where

$$A_k = \frac{\partial f}{\partial x}(x_k^p, u_k^p, 0), \quad B_k = \frac{\partial f}{\partial u}(x_k^p, u_k^p, 0),$$
$$G_k = \frac{\partial f}{\partial w}(x_k^p, u_k^p, 0), H_k = \frac{\partial h}{\partial x}(x_k^p, 0),$$
$$M_k = \frac{\partial h}{\partial v}(x_k^p, 0) \tag{70}$$

Now, let us define the following errors:

- LQG error (main error): $e_k = x_k - x_k^p$
- KF error (estimation error): $\widetilde{e}_k = x_k - \widehat{x}_k^+$
- LQR error (estimation of LQG error): $\widehat{e}_k^+ = \widehat{x}_k^+ - x_k^p$

Note that these errors are linearly dependent: $e_k = \widehat{e}_k^+ + \widetilde{e}_k$. Also, defining $\delta u_k = u_k - u_k^p$ and $\delta z_k = z_k - z_k^p := z_k - h(x_k^p, 0)$, we can rewrite the above linearized models as follows:

$$e_{k+1} = A_k e_k + B_k \delta u_k + G_k w_k \tag{71a}$$

$$\delta z_k = H_k e_k + M_k v_k \tag{71b}$$

KF: In Kalman filtering, we aim to provide an estimate of the system's state based on the available partial information we have obtained up to time k, that is, $z_{0:k}$. The state estimate is a random vector denoted by x_k^+, whose distribution is the conditional distribution of the state on the obtained observations so far, which is called belief and is denoted by b_k:

$$b_k = p(x_k^+) = p(x_k | z_{0:k}) \tag{72}$$

$$\widehat{x}_k^+ = \mathbb{E}[x_k | z_{0:k}] \tag{73}$$

$$P_k = \mathbb{C}[x_k | z_{0:k}] \tag{74}$$

where $\mathbb{E}[\cdot|\cdot]$ and $\mathbb{C}[\cdot|\cdot]$ are the conditional expectation and conditional covariance operators, respectively. In the Gaussian case, we have $b_k = \mathcal{N}(\widehat{x}_k^+, P_k)$; in other words, the belief can be characterized only by its mean and covariance, that is, $b_k \equiv (\widehat{x}_k^+, P_k)$.

Kalman filtering consists of two steps at every time stage: a prediction step and an update step. In the prediction step, the mean and covariance of prior x_k^- are computed. For the system in Equation (71), the prediction step is

$$\widehat{e}_{k+1}^- = A_k \widehat{e}_k^+ + B_k \delta u_k \tag{75}$$

$$P_{k+1}^- = A_k P_k^+ A_k^{\mathrm{T}} + G_k Q_k G_k^{\mathrm{T}} \tag{76}$$

In the update step, the mean and covariance of posterior x_k^+ are computed. For the system in Equation (71), the update step is

$$K_k = P_k^- H_k^{\mathrm{T}} (H_k P_k^- H_k^{\mathrm{T}} + M_k R_k M_k^{\mathrm{T}})^{-1} \tag{77}$$

$$\widehat{e}_{k+1}^+ = \widehat{e}_{k+1}^- + K_{k+1}(\delta z_{k+1} - H_{k+1}\widehat{e}_{k+1}^-) \tag{78}$$

$$P_{k+1}^+ = (I - K_{k+1}H_{k+1}) P_{k+1}^- \tag{79}$$

Note that

$$\widehat{x}_k^+ = \mathbb{E}[x_k | z_{0:k}] = x_k^p + \widehat{e}_k^+ = x_k^p + \mathbb{E}[e_k | z_{0:k}] \tag{80}$$

$$P_k = \mathbb{C}[x_k | z_{0:k}] = P_k^+ = \mathbb{C}[e_k | z_{0:k}] \tag{81}$$

LQR controller: Once we obtain the belief from the filter, a controller can generate an optimal control signal accordingly. In other words, we have a time-varying mapping μ_k from the belief space into the control space that generates an optimal control based on the given belief $u_k = \mu_k(b_k)$ at every time step k. The LQR controller is of this kind and it is optimal in the sense of minimizing the following cost:

$$J_{LQR} = \mathbb{E}\left[\sum_{k\geq 0} (\widehat{x}_k^+ - x_k^p)^{\mathrm{T}} W_x (\widehat{x}_k^+ - x_k^p) + (u_k - u_k^p)^{\mathrm{T}} W_u (u_k - u_k^p)\right]$$
$$= \mathbb{E}\left[\sum_{k\geq 0} (\widehat{e}_k^+)^{\mathrm{T}} W_x (\widehat{e}_k^+) + (\delta u_k)^{\mathrm{T}} W_u (\delta u_k)\right] \tag{82}$$

The linear control law that minimizes this cost function for a linear system is of the form

$$\delta u_k = -L_k \widehat{e}_k^+ \tag{83}$$

where the time-varying feedback gains L_k can be computed recursively as follows:

$$L_k = (B_k^{\mathrm{T}} S_{k+1} B_k + W_u)^{-1} B_k^{\mathrm{T}} S_{k+1} A_k \tag{84}$$

$$S_k = W_x + A_k^{\mathrm{T}} S_{k+1} A_k - A_k^{\mathrm{T}} S_{k+1} B_k L_k \tag{85}$$

If the nominal path is of length N, then $S_N = W_x$ is the initial condition of the above recursion, which is solved backwards in time. Note that the full control is $u_k = u_k^p + \delta u_k$.

LQG controller: Plugging the obtained LQR control law into the Kalman filtering equations, we obtain the following error dynamics for the defined errors:

$$\begin{pmatrix} e_{k+1} \\ \widetilde{e}_{k+1} \end{pmatrix} = \begin{pmatrix} A_k - B_k L_k & B_k L_k \\ 0 & A_k - K_{k+1} H_{k+1} A_k \end{pmatrix} \begin{pmatrix} e_k \\ \widetilde{e}_k \end{pmatrix} + \begin{pmatrix} G_k & 0 \\ G_k - K_{k+1} H_{k+1} G_k & -K_{k+1} M_{k+1} \end{pmatrix} \begin{pmatrix} w_k \\ v_{k+1} \end{pmatrix} \tag{86}$$

or equivalently,

$$\begin{pmatrix} e_{k+1} \\ \widehat{e}_{k+1}^+ \end{pmatrix} = \begin{pmatrix} A_k & -B_k L_k \\ K_{k+1} H_{k+1} A_k A_k - B_k L_k - K_{k+1} H_{k+1} A_k \end{pmatrix} \begin{pmatrix} e_k \\ \widehat{e}_k^+ \end{pmatrix} + \begin{pmatrix} G_k & 0 \\ K_{k+1} H_{k+1} G_k & K_{k+1} M_{k+1} \end{pmatrix} \begin{pmatrix} w_k \\ v_{k+1} \end{pmatrix} \tag{87}$$

Defining $\zeta_k := (e_k, \widehat{e}_k^+)^{\mathrm{T}}$ and $q_k := (w_k, v_{k+1})^{\mathrm{T}}$, we can rewrite Equation (87) in a more compact form as

$$\zeta_{k+1} = \overline{F}_k \zeta_k + \overline{G}_k q_k, \quad q_k \sim \mathcal{N}(0, \overline{Q}_k), \quad \overline{Q}_k = \begin{pmatrix} Q_k & 0 \\ 0 & R_{k+1} \end{pmatrix} \tag{88}$$

with appropriate definitions for $\overline{F}_k$ and $\overline{G}_k$.

The above equation, along with the equation on estimation covariance propagation,

$$P_{k+1} = (I - K_{k+1}H_{k+1})(A_k P_k A_k^{\mathrm{T}} + G_k Q_k G_k^{\mathrm{T}}) \tag{89}$$

characterize the evolution of state x_k and belief $b_k \equiv (\widehat{x}_k^+, P_k)$ under the time-varying LQG controller.

C. SLQG controller

The SLQG controller is often used to regulate (or stabilize) the system state to a pre-planned point (also called the set-point, nominal, or desired point) in the presence of process and observation noise. In principal it is designed (and optimal) for linear systems with Gaussian noise, but it can also be utilized for stabilizing nonlinear systems locally around the planned point. The SLQG controller is composed of a stationary Kalman filter (SKF) as an estimator and a stationary linear quadratic regulator (SLQR) as a controller. At every time step k, the SKF provides the *a posteriori* distribution (belief) b_k over the system state, and the SLQR generates control u_k based on b_k.

In this appendix, we first discuss the system linearization around the planned point, and then discuss the SKF, SLQR, and SLQG corresponding to this nominal point. Consider the nonlinear partially observable state-space equations of the system as follows:

$$x_{k+1} = f(x_k, u_k, w_k), \quad w_k \sim \mathcal{N}(0, Q_k) \tag{90a}$$

$$z_k = h(x_k, v_k), \quad v_k \sim \mathcal{N}(0, R_k) \tag{90b}$$

and consider a planned state point x^p, to whose vicinity the controller has to drive the system. If the system state reaches the point x^p, it is assumed that the system remains there with zero control, $u^p = 0$, in other words,

$$x^p = f(x^p, 0, 0) \tag{91}$$

The role of a closed-loop stochastic controller during the state regulation is to compensate for robot deviations from the desired point due to noise effects, and to drive the robot close to the desired point in the sense of minimizing the following quadratic cost:

$$J = \mathbb{E}\left[\sum_{k\geq 0} (x_k - x^p)^{\mathrm{T}} W_x (x_k - x^p) + (u_k)^{\mathrm{T}} W_u (u_k)\right] \tag{92}$$

where W_x and W_u are positive-definite weight matrices for the state and control costs, respectively.

Again, similar to the time-varying case, since we only have imperfect information about the state x_k, we have to make the estimate x_k^+ about the state based on the available observations $z_{0:k}$. Accordingly, the controller generates the control signal based on the estimated value of the state; i.e., belief. Based on the separation principle (Bertsekas, 2007), in a linear system with Gaussian noise, minimization of the cost in Equation (92) is equivalent to performing two separate minimizations that lead to the separate design of the SKF and SLQR. In the following, we first discuss the linearization of a nonlinear model, and then we discuss how the SKF and the SLQR can be designed for this linearized system, and finally, we combine them to construct an SLQG controller.

Model linearization: Given a desired point x^p, we linearize the dynamics and observation model in Equation (90) as follows:

$$x_{k+1} = f(x^p, 0, 0) + A_s(x_k - x^p) + B_s(u_k - 0) + G_s w_k, \quad w_k \sim \mathcal{N}(0, Q_s) \tag{93a}$$

$$z_k = h(x^p, 0) + H_s(x_k - x^p) + M_s v_k, \quad v_k \sim \mathcal{N}(0, R_s) \tag{93b}$$

where

$$A_s = \frac{\partial f}{\partial x}(x^p, 0, 0), \quad B_s = \frac{\partial f}{\partial u}(x^p, 0, 0), \quad G_s = \frac{\partial f}{\partial w}(x^p, 0, 0)$$
$$H_s = \frac{\partial h}{\partial x}(x^p, 0), \quad M_s = \frac{\partial h}{\partial v}(x^p, 0) \tag{94}$$

Now, let us define the following errors:

- SLQG error (main error): $e_k = x_k - x^p$
- SKF error (estimation error): $\widetilde{e}_k = x_k - \widehat{x}_k^+$, where $\widehat{x}_k^+ = \mathbb{E}[x_k^+] = \mathbb{E}[x_k | z_{0:k}]$
- SLQR error (estimation of SLQG error): $\widehat{e}_k^+ = \widehat{x}_k^+ - x^p$

Note that these errors are linearly dependent: $e_k = \widehat{e}_k^+ + \widetilde{e}_k$. Defining $\delta u_k := u_k$ and $\delta z_k := z_k - z^p = z_k - h(x^p, 0)$, we can rewrite the above linearized models as follows:

$$e_{k+1} = A_s e_k + B_s \delta u_k + G_s w_k \tag{95a}$$

$$\delta z_k = H_s e_k + M_s v_k \tag{95b}$$

SKF: In SKF, we aim to provide an estimate of the system's state based on the available partial information we have obtained up to time k, that is, $z_{0:k}$. The state estimate is a random vector denoted by x_k^+, whose distribution is the conditional distribution of the state on the obtained observations so far, which is called belief and is denoted by $b_k = p(x_k^+) = p(x_k | z_{0:k})$. In the Gaussian case, the belief can only be characterized by its mean and covariance, that is, $b_k \equiv (\widehat{x}_k^+, P_k)$. Thus, in the Gaussian case, we can write

$$b_k = p(x_k^+) = p(x_k | z_{0:k}) = \mathcal{N}(\widehat{x}_k^+, P_k) \Leftrightarrow b_k \equiv (\widehat{x}_k^+, P_k) \tag{96}$$

$$\widehat{x}_k^+ = \mathbb{E}[x_k | z_{0:k}], \quad P_k = \mathbb{C}[x_k | z_{0:k}] \tag{97}$$

where $\mathbb{E}[\cdot|\cdot]$ and $\mathbb{C}[\cdot|\cdot]$ are the conditional expectation and conditional covariance operators, respectively.

SKF consists of two steps at every time stage: a prediction step and an update step. In the prediction step, the mean and covariance of prior x_k^- are computed. For the system in Equation (95) the prediction step is

$$\widehat{e}_{k+1}^- = A_s \widehat{e}_k^+ + B_s \delta u_k \tag{98}$$
$$P_{k+1}^- = A_s P_k^+ A_s^{\mathrm{T}} + G_s Q_s G_s^{\mathrm{T}} \tag{99}$$

In the update step, the mean and covariance of posterior x_k^+ are computed. For the error system in Equation (95), the update step is

$$K_k = P_k^- H_s^{\mathrm{T}} (H_s P_k^- H_s^{\mathrm{T}} + M_s R_s M_s^{\mathrm{T}})^{-1} \tag{100}$$
$$\widehat{e}_{k+1}^+ = \widehat{e}_{k+1}^- + K_{k+1}(\delta z_{k+1} - H_s \widehat{e}_{k+1}^-) \tag{101}$$
$$P_{k+1}^+ = (I - K_{k+1} H_s) P_{k+1}^- \tag{102}$$

Note that

$$\widehat{x}_k^+ = x^p + \widehat{e}_k^+, \quad P_k = P_k^+ \tag{103}$$

In SKF, if (A_s, H_s) is an observable pair and $(A_s, \check{Q}_s)$ is a controllable pair, where $G_s Q_s G_s^{\mathrm{T}} = \check{Q}_s \check{Q}_s^{\mathrm{T}}$, then the prior and posterior covariances P_k^- and P_k and the filter gain K_k all converge to their stationary values, denoted by P_s^-, P_s, and K_s, respectively (Bertsekas, 2007). P_s^- can be computed by solving the following DARE in Equation (104). Having P_s^-, the stationary gain K_s and estimation covariance P_s are computed as follows:

$$P_s^- = G_s Q_s G_s^{\mathrm{T}} + A_s (P_s^- - P_s^- H_s^{\mathrm{T}} (H_s P_s^- H_s^{\mathrm{T}} + M_s R_s M_s^{\mathrm{T}})^{-1} H_s P_s^-) A_s^{\mathrm{T}} \tag{104}$$
$$K_s = P_s^- H_s^{\mathrm{T}} (H_s P_s^- H_s^{\mathrm{T}} + M_s R_s M_s^{\mathrm{T}})^{-1} \tag{105}$$
$$P_s = (I - K_s H_s) P_s^- \tag{106}$$

SLQR controller: In the SLQR we have a stationary mapping μ_s from the belief space to the control space that generates an optimal control based on the given belief $u_k = \mu_s(b_k)$ at every time step k. The SLQR controller is optimal in the sense of minimizing the following cost:

$$J_{SLQR} = \mathbb{E}\left[\sum_{k\geq 0} (\widehat{x}_k^+ - x^p)^{\mathrm{T}} W_x (\widehat{x}_k^+ - x^p) + (u_k)^{\mathrm{T}} W_u (u_k)\right]$$
$$= \mathbb{E}\left[\sum_{k\geq 0} (\widehat{e}_k^+)^{\mathrm{T}} W_x (\widehat{e}_k^+) + (\delta u_k)^{\mathrm{T}} W_u (\delta u_k)\right] \tag{107}$$

If (A_s, B_s) is a controllable pair and $(A_s, \check{W}_x)$ is an observable pair, where $\check{W}_x^{\mathrm{T}} \check{W}_x = W_x$, then the stationary linear control law that minimizes the cost function J_{SLQR} for a linear system is of the form

$$\delta u_k = -L_s \widehat{e}_k^+ \tag{108}$$

where the stationary feedback gain L_s can be computed as follows:

$$L_s = (B_s^{\mathrm{T}} S_s B_s + W_u)^{-1} B_s^{\mathrm{T}} S_s A_s \tag{109}$$
$$S_s = W_x + A_s^{\mathrm{T}} S_s A_s - A_s^{\mathrm{T}} S_s B_s L_s \tag{110}$$

where the second equation is indeed a DARE that can be efficiently solved for S_s. Plugging S_s into Equation (109), we get the feedback gain L_s.

SLQG controller: Plugging the obtained control law of SLQR into the SKF equations, we can get the following stationary dynamics for the defined errors:

$$\begin{pmatrix} e_{k+1} \\ \widetilde{e}_{k+1} \end{pmatrix} = \begin{pmatrix} A_s - B_s L_s & B_s L_s \\ 0 & A_s - K_s H_s A_s \end{pmatrix} \begin{pmatrix} e_k \\ \widetilde{e}_k \end{pmatrix} + \begin{pmatrix} G_s & 0 \\ G_s - K_s H_s G_s & -K_s M_s \end{pmatrix} \begin{pmatrix} w_k \\ v_{k+1} \end{pmatrix} \tag{111}$$

or equivalently,

$$\begin{pmatrix} e_{k+1} \\ \widehat{e}_{k+1}^+ \end{pmatrix} = \begin{pmatrix} A_s & -B_s L_s \\ K_s H_s A_s & A_s - B_s L_s - K_s H_s A_s \end{pmatrix} \begin{pmatrix} e_k \\ \widehat{e}_k^+ \end{pmatrix} + \begin{pmatrix} G_s & 0 \\ K_s H_s G_s & K_s M_s \end{pmatrix} \begin{pmatrix} w_k \\ v_{k+1} \end{pmatrix} \tag{112}$$

Defining $\zeta_k := (e_k, \widehat{e}_k^+)^{\mathrm{T}}$ and $q_k := (w_k, v_{k+1})^{\mathrm{T}}$, we can rewrite Equation (112) in a more compact form as

$$\zeta_{k+1} = \overline{F}_s \zeta_k + \overline{G}_s q_k, \quad q_k \sim \mathcal{N}(0, \overline{Q}_s), \quad \overline{Q}_s = \begin{pmatrix} Q_s & 0 \\ 0 & R_s \end{pmatrix} \tag{113}$$

with appropriate definitions for $\overline{F}_s$ and $\overline{G}_s$.

It can be shown that if $\overline{F}_s$ is a stable matrix (i.e. $\lim_{\kappa\to\infty}(\overline{F}_s)^\kappa = 0$), ζ_k converges i.d. to $\zeta_s \sim \mathcal{N}(0, \mathcal{P}_s)$. Stationary covariance $\mathcal{P}_s$ is the solution of the following Lyapunov equation:

$$\mathcal{P}_s = \overline{F}_s \mathcal{P}_s \overline{F}_s^{\mathrm{T}} + \overline{G}_s \overline{Q}_s \overline{G}_s^{\mathrm{T}} \tag{114}$$

Note that $\mathcal{P}_s$ can be decomposed into four blocks,

$$\mathcal{P}_s = \begin{pmatrix} \mathcal{P}_{s,11} & \mathcal{P}_{s,12} \\ \mathcal{P}_{s,21} & \mathcal{P}_{s,22} \end{pmatrix} \tag{115}$$

such that $\mathcal{P}_{s,11} = \lim_{k\to\infty} \mathbb{C}[e_k]$ and $\mathcal{P}_{s,22} = \lim_{k\to\infty} \mathbb{C}[\widehat{e}_k^+]$. Therefore, since $\widehat{x}_k^+ = x^p + \widehat{e}_k^+$, the estimation mean also converges to a stationary random variable, denoted by $\widehat{x}_s^+$:

$$\widehat{x}_s^+ := \lim_{k\to\infty} \widehat{x}_k^+ \sim \mathcal{N}(x^p, \mathcal{P}_{s,22}) \tag{116}$$

Due to the linear relation $e_k = \widehat{e}_k^+ + \widetilde{e}_k$, we can also conclude $\lim_{k\to\infty} \mathbb{C}[\widetilde{e}_k] = \mathcal{P}_{s,11} + \mathcal{P}_{s,22} - 2\mathcal{P}_{s,12}$. It can be proven that in SLQG, the stability of matrix $\overline{F}_s$ is a direct consequence of the controllability of pair (A_s, B_s) and the observability of pair (A_s, H_s) (Bertsekas, 1976, 2007).

Thus, collecting all the conditions, if (A_s, B_s) and $(A_s, \breve{Q}_s)$ are controllable pairs, where $G_s Q_s G_s^{\mathrm{T}} = \breve{Q}_s \breve{Q}_s^{\mathrm{T}}$, and if (A_s, H_s) and $(A_s, \breve{W}_x)$ are observable pairs, where $W_x = \breve{W}_x^{\mathrm{T}} \breve{W}_x$, then the belief b_k converges in distribution to a stationary belief under the SLQG:

$$b_s := \lim_{k\to\infty} b_k = \mathcal{N}(\widehat{x}_s^+, P_s^+) \tag{117}$$

where P_s^+ is a deterministic quantity and we can characterize the distribution over the stationary belief as

$$b_s \equiv (\widehat{x}_s^+, P_s^+) \sim \mathcal{N}\left(\begin{pmatrix} x^p \\ P_s^+ \end{pmatrix}, \begin{pmatrix} \mathcal{P}_{s,22} & 0 \\ 0 & 0 \end{pmatrix}\right) \tag{118}$$

D. Proof of Lemma 3

Proof. Let us consider the state-space model of the linear system of interest as follows:

$$x_{k+1} = \mathbf{A}x_k + \mathbf{B}u_k + \mathbf{G}w_k, \quad w_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}) \tag{119a}$$

$$z_k = \mathbf{H}x_k + v_k, \quad v_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}) \tag{119b}$$

Based on Lemma 1, if $(\mathbf{A}, \mathbf{B})$ and $(\mathbf{A}, \breve{\mathbf{Q}})$ are controllable pairs, where $\mathbf{G}\mathbf{Q}\mathbf{G}^{\mathrm{T}} = \breve{\mathbf{Q}}\breve{\mathbf{Q}}^{\mathrm{T}}$, and if $(\mathbf{A}, \mathbf{H})$ and $(\mathbf{A}, \breve{\mathbf{W}}_x)$ are observable pairs, where $\mathbf{W}_x = \breve{\mathbf{W}}_x^{\mathrm{T}} \breve{\mathbf{W}}_x$, then the estimation covariance deterministically tends to a stationary covariance P_s. Therefore, for any $\epsilon > 0$, after a deterministic finite time, P_k enters the ϵ-neighborhood of the stationary covariance, denoted by P_s.

The estimation mean dynamics, however, are stochastic and are as follows for the system in Equation (119):

$$\begin{aligned} \widehat{x}_{k+1}^+ &= \mathbf{v} + (\mathbf{A} - \mathbf{B}\mathbf{L} - \mathbf{K}_{k+1}\mathbf{H}\mathbf{A})(\widehat{x}_k^+ - \mathbf{v}) \\ &\quad + \mathbf{K}_{k+1}\mathbf{H}\mathbf{A}(x_k - \mathbf{v}) + \mathbf{K}_{k+1}\mathbf{H}\mathbf{G}w_k + \mathbf{K}_{k+1}v_{k+1} \\ &= \mathbf{v} - (\mathbf{A} - \mathbf{B}\mathbf{L})\mathbf{v} + (\mathbf{A} - \mathbf{B}\mathbf{L} - \mathbf{K}_{k+1}\mathbf{H}\mathbf{A})\widehat{x}_k^+ \\ &\quad + \mathbf{K}_{k+1}\mathbf{H}\mathbf{A}x_k + \mathbf{K}_{k+1}\mathbf{H}\mathbf{G}w_k + \mathbf{K}_{k+1}v_{k+1} \end{aligned} \tag{120}$$

where the Kalman gain $\mathbf{K}_k$ is

$$\mathbf{K}_k = P_k^- \mathbf{H}^T (\mathbf{H} P_k^- \mathbf{H}^T + \mathbf{R})^{-1} \tag{121}$$

Since $\mathbf{K}$ is full rank (due to the condition on the rank of $\mathbf{H}$), and since v and w represent Gaussian noise, Equation (120) induces an irreducible Markov process over the state space (Meyn and Tweedie, 2009). Thus, if we have a stopping region for the estimation mean with size $\epsilon > 0$, the estimation mean process will hit this stopping region in finite time (Meyn and Tweedie, 2009), with probability one.

Based on the estimation mean dynamics in Equation (120) and the state dynamics in Appendix C, in the absence of a stopping region, if the estimation mean process and state process start from $\widehat{x}_0^+$ and x_0 respectively, such that $\mathbb{E}[\widehat{x}_0^+] = \mathbf{v}$ and $\mathbb{E}[x_0] = \mathbf{v}$ (which indeed is the case in FIRM due to the usage of edge controllers), 'the mean of estimation mean' remains on $\mathbf{v}$. That is, $\mathbb{E}[\widehat{x}_k^+] = \mathbf{v}$, for all k. As a result, if we center the stopping region for the estimation mean at $\mathbf{v}$, the probability of hitting the stopping region is maximized and the stopping time is minimized.

Combining the results for estimation covariance and estimation mean, if we define the region B as a set in the Gaussian belief space with a non-empty interior centered at $(\mathbf{v}, P_s)$, the belief $b_k \equiv (\widehat{x}_k^+, P_k)$ enters region B in finite time with probability one. Thus, the pair (B, μ) is a proper pair over $\mathbb{GB}$; in other words, B is reachable under μ starting from any Gaussian distribution. □

E. Proof of Lemma 4

Before proving Lemma 4, we state and prove the following lemma.

Lemma 5. *Consider the bounded function* $0 \le f(\mathcal{X}) \le 1$, *and kernel* $k(\mathcal{X}', \mathcal{X}) \ge 0$. *Then, for any set* $\mathcal{A}$, *we have*

$$\| \int_{\mathcal{A}} k(\mathcal{X}', \mathcal{X}) f(\mathcal{X}')\, d\mathcal{X}' \| \le \| \int_{\mathcal{A}} k(\mathcal{X}', \mathcal{X})\, d\mathcal{X}' \| \tag{122}$$

Proof. Given the properties of $f(\cdot)$ and $k(\cdot,\cdot)$, we have $k(\mathcal{X}', \mathcal{X}) f(\mathcal{X}') \le k(\mathcal{X}', \mathcal{X})$, for all $\mathcal{X}$ and $\mathcal{X}'$. Taking the integral from both sides with respect to $\mathcal{X}'$ and then taking the supremum norm with respect to $\mathcal{X}$, the result follows. □

Now we prove Lemma 4.

Proof. If we denote the domain of operator $\mathbf{T}_{\mathcal{S}}$ by $\mathcal{D}$, we know that for all $f \in \mathcal{D}$ we have $0 \le f(\mathcal{X}) \le 1$, because $f(\mathcal{X})$ is the probability of reaching given set $\mathcal{S}$ under some given controller invoked at point $\mathcal{X}$. Thus, it cannot be negative or greater than one, and based on Lemma 5, we have

$$\begin{aligned} \mathbf{T}_{\mathcal{S}}[f] &= \int_{\overline{\mathcal{S}}} p^{\mu}(\mathcal{X}'|\mathcal{X}) f(\mathcal{X}')\, d\mathcal{X}' \\ &\le \int_{\overline{\mathcal{S}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\, d\mathcal{X}' = \mathbb{P}_1(\overline{\mathcal{S}}|\mathcal{X}, \mu) \le 1 \end{aligned} \tag{123}$$

Therefore, based on the definition of operator norm, we have

$$\|\mathbf{T}_{\mathcal{S}}\|_{op} = \sup_{f(\cdot)} \{\|\mathbf{T}_{\mathcal{S}}[f]\| \ : \ \forall f \in \mathcal{D}, \|f\| \le 1\} \le 1 \tag{124}$$

According to Assumption 1, there exists a finite number N such that

$$\inf_{\mathcal{X}} \mathbb{P}_n(\mathcal{S}|\mathcal{X}, \mu) = \beta > 0 \ \ \forall n > N \tag{125}$$

where 'inf' and 'sup' denote the infimum and supremum, respectively. Thus, we have

$$\begin{aligned} \|\mathbb{P}_n(\overline{\mathcal{S}}|\mathcal{X}, \mu)\| &= \sup_{\mathcal{X}} (1 - \mathbb{P}_n(\mathcal{S}|\mathcal{X}, \mu)) \\ &= 1 - \inf_{\mathcal{X}} \mathbb{P}_n(\mathcal{S}|\mathcal{X}, \mu) \\ &= 1 - \beta < 1 \ \ \forall n > N \end{aligned} \tag{126}$$

Let us denote the nth iterated kernel of $\mathbf{T}_{\mathcal{S}}$ as $p_n(\mathcal{X}'|\mathcal{X},\mu)$. Since this iterated kernel is a pdf, we have $p_n(\mathcal{X}'|\mathcal{X},\mu) \geq 0,\ \forall \mathcal{X}, \forall \mathcal{X}', \forall n$. We can write

$$\|\mathbf{T}_{\mathcal{S}}^N[f]\| = \|\int_{\overline{\mathcal{S}}} p_N(\mathcal{X}'|\mathcal{X},\mu) f(\mathcal{X}')\, d\mathcal{X}'\|$$
$$\leq \|\int_{\overline{\mathcal{S}}} p_N(\mathcal{X}'|\mathcal{X},\mu)\, d\mathcal{X}'\| = \|\mathbb{P}_N(\overline{\mathcal{S}}|\mathcal{X},\mu)\| \leq \alpha < 1 \tag{127}$$

where $\alpha = 1-\beta$, and similar to Equation (124), we get $\|\mathbf{T}_{\mathcal{S}}^N\|_{op} \leq \alpha < 1$. From the operator norm properties, we have

$$\|\mathbf{T}_{\mathcal{S}}^{N+1}\|_{op} \leq \|\mathbf{T}_{\mathcal{S}}^N\|_{op}\|\mathbf{T}_{\mathcal{S}}\|_{op} \leq \alpha < 1$$

and similarly for all $n \geq N$, we have

$$\|\mathbf{T}_{\mathcal{S}}^n\|_{op} \leq \alpha < 1 \ \ \forall n \geq N$$

Now, consider the series: $\sum_{i=1}^{\infty} \|\mathbf{T}_{\mathcal{S}}^n\|_{op}$. We can split the sum into smaller pieces as follows:

$$\sum_{n=1}^{\infty} \|\mathbf{T}_{\mathcal{S}}^n\|_{op} = \sum_{n=1}^{N} \|\mathbf{T}_{\mathcal{S}}^n\|_{op} + \sum_{i=1}^{\infty} \sum_{n=iN+1}^{(i+1)N} \|\mathbf{T}_{\mathcal{S}}^n\|_{op}$$

But because $\|\mathbf{T}_{\mathcal{S}}^{n+1}\|_{op} \leq \|\mathbf{T}_{\mathcal{S}}^n\|_{op}$ for all $n \geq N$, we have

$$\sum_{n=iN+1}^{(i+1)N} \|\mathbf{T}_{\mathcal{S}}^n\|_{op} \leq N\|\mathbf{T}_{\mathcal{S}}^{iN}\|_{op}$$

Also, we know

$$\|\mathbf{T}_{\mathcal{S}}^{iN}\|_{op} \leq \|\mathbf{T}_{\mathcal{S}}^N\|_{op}^i \leq \alpha^i$$

and thus, we have

$$\sum_{n=1}^{\infty} \|\mathbf{T}_{\mathcal{S}}^n\|_{op} = \underbrace{\sum_{n=1}^{N} \|\mathbf{T}_{\mathcal{S}}^n\|_{op}}_{\leq N} + \sum_{i=1}^{\infty} \sum_{n=iN+1}^{(i+1)N} \|\mathbf{T}_{\mathcal{S}}^n\|_{op}$$
$$\leq N + \sum_{i=1}^{\infty} N\alpha^i = N + \frac{N}{1-\alpha} = c < \infty$$

□

F. Proof of Corollary 1

Proof. We know $\|R\| \leq 1$, and thus we can write

$$\|\sum_{n=0}^{\infty} \mathbf{T}_{\mathcal{S}}^n[R]\| \leq \sum_{n=0}^{\infty} \|\mathbf{T}_{\mathcal{S}}^n\|_{op}\|R\| \leq \sum_{n=0}^{\infty} \|\mathbf{T}_{\mathcal{S}}^n\|_{op} \leq c < \infty$$

Thus, series $\sum_{n=0}^{\infty} \mathbf{T}_{\mathcal{S}}^n[R]$ is a convergent series and we can define the operator $(I-\mathbf{T}_{\mathcal{S}})^{-1}[R] = \sum_{n=0}^{\infty} \mathbf{T}_{\mathcal{S}}^n[R]$. We have

$$\|(I-\mathbf{T}_{\mathcal{S}})^{-1}\|_{op} = \|\sum_{n=0}^{\infty} \mathbf{T}_{\mathcal{S}}^n\|_{op} \leq c < \infty \tag{128}$$

□

G. Proof of Proposition 1

We first state and prove the following lemma on the continuity of the transition probability in the local controller's parameter.

Lemma 6. *Given Assumption 2, there exists a $c_2 < \infty$ such that*

$$\|p(\mathcal{X}'|\mathcal{X},\mu(b;\mathbf{v})) - p(\mathcal{X}'|\mathcal{X},\check{\mu}(b;\check{\mathbf{v}}))\| \leq c_2\|\mathbf{v}-\check{\mathbf{v}}\| \tag{129}$$

Proof. The result directly follows by combining the two parts of Assumption 2. □

Now we are ready to prove Proposition 1.

Proof. To show $\mathbb{P}(\mathcal{B}|\mathcal{X},\mu)$ is continuous in $\mathbf{v}$, we perturb $\mathbf{v}$ to some $\check{\mathbf{v}}$, such that $\|\mathbf{v}-\check{\mathbf{v}}\| < r$. The local controller associated with node $\check{\mathbf{v}}$ is referred to as $\check{\mu}$, whose successful absorption region is denoted by $\check{\mathcal{B}}$ and whose stopping region is $\check{\mathcal{S}}$. Similarly, the corresponding transient operator and recurrent function are referred to as $\check{\mathbf{T}}_{\check{\mathcal{S}}}$ and $\check{R}$ respectively. Finally, the success probability associated with the perturbed node $\check{\mathbf{v}}$ is $\mathbb{P}(\check{\mathcal{B}}|\mathcal{X},\check{\mu})$. To shorten the statements, we refer to $\mathbb{P}(\mathcal{B}|\mathcal{X},\mu)$ and $\mathbb{P}(\check{\mathcal{B}}|\mathcal{X},\check{\mu})$ respectively as $\mathfrak{P}(\mathcal{X})$ and $\check{\mathfrak{P}}(\mathcal{X})$. As a result of node perturbation, the success probability is perturbed as

$$\mathbb{P}(\mathcal{B}|\mathcal{X},\mu) - \mathbb{P}(\check{\mathcal{B}}|\mathcal{X},\check{\mu}) := \mathfrak{P} - \check{\mathfrak{P}} = R + \mathbf{T}_{\mathcal{S}}[\mathfrak{P}] - \check{R} - \check{\mathbf{T}}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}]$$
$$= R - \check{R} + \mathbf{T}_{\mathcal{S}}[\mathfrak{P}] - \mathbf{T}_{\mathcal{S}}[\check{\mathfrak{P}}] + \mathbf{T}_{\mathcal{S}}[\check{\mathfrak{P}}] - \mathbf{T}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}] + \mathbf{T}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}] - \check{\mathbf{T}}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}]$$
$$= (R-\check{R}) + \mathbf{T}_{\mathcal{S}}[\mathfrak{P}-\check{\mathfrak{P}}] + (\mathbf{T}_{\mathcal{S}} - \mathbf{T}_{\check{\mathcal{S}}})[\check{\mathfrak{P}}] + (\mathbf{T}_{\check{\mathcal{S}}} - \check{\mathbf{T}}_{\check{\mathcal{S}}})[\check{\mathfrak{P}}]$$

where

$$\mathbf{T}_{\check{\mathcal{S}}}[f(\cdot)](\mathcal{X}) := \int_{\overline{\check{\mathcal{S}}}} p^{\mu}(\mathcal{X}'|\mathcal{X}) f(\mathcal{X}')\, d\mathcal{X}' \tag{130}$$

Let us define the operators $\mathbf{T}_{\Delta\mathcal{S}} := (\mathbf{T}_{\mathcal{S}} - \mathbf{T}_{\check{\mathcal{S}}})$ and $\Delta\mathbf{T}_{\check{\mathcal{S}}} := (\mathbf{T}_{\check{\mathcal{S}}} - \check{\mathbf{T}}_{\check{\mathcal{S}}})$. Now, based on Corollary 1, we can write

$$\mathfrak{P} - \check{\mathfrak{P}} = (I-\mathbf{T}_{\mathcal{S}})^{-1}\left[R - \check{R} + \mathbf{T}_{\Delta\mathcal{S}}[\check{\mathfrak{P}}] + \Delta\mathbf{T}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}]\right] \tag{131}$$

and thus the following inequality holds on the supremum norm of the perturbation of the absorption probability:

$$\|\mathfrak{P} - \check{\mathfrak{P}}\|$$
$$\leq \|(I-\mathbf{T}_{\mathcal{S}})^{-1}\|_{op}\left(\|R-\check{R}\| + \|\mathbf{T}_{\Delta\mathcal{S}}[\check{\mathfrak{P}}]\| + \|\Delta\mathbf{T}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}]\|\right)$$
$$\leq c\left(\|R-\check{R}\| + \|\mathbf{T}_{\Delta\mathcal{S}}[\check{\mathfrak{P}}]\| + \|\Delta\mathbf{T}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}]\|\right)$$
$$= c\left(\|K_1(\mathcal{X})\| + \|K_2(\mathcal{X})\| + \|K_3(\mathcal{X})\|\right) \tag{132}$$

where $K_1(\mathcal{X}) := R(\mathcal{X}) - \check{R}(\mathcal{X})$, $K_2(\mathcal{X}) := \mathbf{T}_{\Delta\mathcal{S}}[\check{\mathfrak{P}}(\cdot)](\mathcal{X})$, and $K_3(\mathcal{X}) := \Delta\mathbf{T}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}(\cdot)](\mathcal{X})$. In the following we bound K_1, K_2, and K_3, and thus bound $\|\mathfrak{P} - \check{\mathfrak{P}}\|$, accordingly.

G.1. Bound for $K_1(\mathcal{X})$

The supremum norm of $K_1(\mathcal{X})$ is

$$\begin{aligned}
&\|K_1(\mathcal{X})\| = \|R(\mathcal{X}) - \check{R}(\mathcal{X})\| \\
&= \| \int_{\mathcal{B}} p^{\mu}(\mathcal{X}'|\mathcal{X})\, d\mathcal{X}' - \int_{\check{\mathcal{B}}} p^{\check{\mu}}(\mathcal{X}'|\mathcal{X})\, d\mathcal{X}' \| \\
&= \| \int_{\mathcal{B}\cap\check{\mathcal{B}}} [p^{\mu}(\mathcal{X}'|\mathcal{X}) - p^{\check{\mu}}(\mathcal{X}'|\mathcal{X})]\, d\mathcal{X}' \\
&+ \int_{\mathcal{B}-\check{\mathcal{B}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\, d\mathcal{X}' - \int_{\check{\mathcal{B}}-\mathcal{B}} p^{\check{\mu}}(\mathcal{X}'|\mathcal{X})\, d\mathcal{X}' \| \\
&\leq \int_{\mathcal{B}\cap\check{\mathcal{B}}} \|p^{\mu}(\mathcal{X}'|\mathcal{X}) - p^{\check{\mu}}(\mathcal{X}'|\mathcal{X})\|\, d\mathcal{X}' \\
&+ \| \int_{\mathcal{B}-\check{\mathcal{B}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\, d\mathcal{X}' + \int_{\check{\mathcal{B}}-\mathcal{B}} p^{\check{\mu}}(\mathcal{X}'|\mathcal{X})\, d\mathcal{X}' \| \\
&\overset{\text{from (129)}}{\leq} \int_{\mathcal{B}\cap\check{\mathcal{B}}} c_2\|\mathbf{v}-\check{\mathbf{v}}\| d\mathcal{X}' + \|\mathbb{P}_1(\mathcal{B}\ominus\check{\mathcal{B}}|\mathcal{X},\mu)\| \\
&\qquad + \|\mathbb{P}_1(\check{\mathcal{B}}\ominus\mathcal{B}|\mathcal{X},\check{\mu})\| \\
&\overset{\text{from (50)}}{\leq} c_2'\|\mathbf{v}-\check{\mathbf{v}}\| + 2c'\|\mathbf{v}-\check{\mathbf{v}}\| = \gamma_1\|\mathbf{v}-\check{\mathbf{v}}\| \qquad (133)
\end{aligned}$$

where $c_2' < \infty$ and $\gamma_1 = c_2' + 2c' < \infty$. In the penultimate inequality, we also used the fact that $\mathbb{P}_1(\check{\mathcal{B}}-\mathcal{B}|\mathcal{X},\check{\mu}) \leq \mathbb{P}_1(\check{\mathcal{B}}\ominus\mathcal{B}|\mathcal{X},\check{\mu})$ and $\mathbb{P}_1(\mathcal{B}-\check{\mathcal{B}}|\mathcal{X},\mu) \leq \mathbb{P}_1(\mathcal{B}\ominus\check{\mathcal{B}}|\mathcal{X},\mu)$ because $\check{\mathcal{B}}-\mathcal{B} \subseteq \check{\mathcal{B}}\ominus\mathcal{B}$ and $\mathcal{B}-\check{\mathcal{B}} \subseteq \mathcal{B}\ominus\check{\mathcal{B}}$.

G.2. Bound for $K_2(\mathcal{X})$

We have

$$\begin{aligned}
&\|K_2(\mathcal{X})\| = \|\mathbf{T}_{\Delta\mathcal{S}}[\check{\mathfrak{P}}]\| = \|\mathbf{T}_{\mathcal{S}}[\check{\mathfrak{P}}] - \mathbf{T}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}]\| \\
&= \| \int_{\overline{\mathcal{S}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\check{\mathfrak{P}}(\mathcal{X}')\, d\mathcal{X}' - \int_{\overline{\check{\mathcal{S}}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\check{\mathfrak{P}}(\mathcal{X}')\, d\mathcal{X}' \| \\
&= \| \int_{\overline{\mathcal{S}}-\overline{\check{\mathcal{S}}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\check{\mathfrak{P}}(\mathcal{X}')\, d\mathcal{X}' - \int_{\overline{\check{\mathcal{S}}}-\overline{\mathcal{S}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\check{\mathfrak{P}}(\mathcal{X}')\, d\mathcal{X}' \| \\
&\leq \| \int_{\overline{\mathcal{S}}-\overline{\check{\mathcal{S}}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\check{\mathfrak{P}}(\mathcal{X}')\, d\mathcal{X}' + \int_{\overline{\check{\mathcal{S}}}-\overline{\mathcal{S}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\check{\mathfrak{P}}(\mathcal{X}')\, d\mathcal{X}' \| \\
&= \| \int_{\overline{\mathcal{S}}\ominus\overline{\check{\mathcal{S}}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\check{\mathfrak{P}}(\mathcal{X}')\, d\mathcal{X}' \| \overset{\text{from (122)}}{\leq} \| \int_{\overline{\mathcal{S}}\ominus\overline{\check{\mathcal{S}}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\, d\mathcal{X}' \| \\
&= \|\mathbb{P}_1(\overline{\mathcal{S}}\ominus\overline{\check{\mathcal{S}}}|\mathcal{X},\mu)\| \leq \|\mathbb{P}_1(\overline{\mathcal{B}}\ominus\overline{\check{\mathcal{B}}}|\mathcal{X},\mu)\| \\
&= \|\mathbb{P}_1(\mathcal{B}\ominus\check{\mathcal{B}}|\mathcal{X},\mu)\| \overset{\text{from (50)}}{\leq} \gamma_2\|\mathbf{v}-\check{\mathbf{v}}\| \qquad (134)
\end{aligned}$$

where $\gamma_2 = c' < \infty$. The penultimate inequality and equality follow from the relations $\overline{\mathcal{S}}\ominus\overline{\mathcal{S}'} \subseteq \overline{\mathcal{B}}\ominus\overline{\mathcal{B}'}$ and $\overline{\mathcal{B}}\ominus\overline{\mathcal{B}'} = \mathcal{B}\ominus\mathcal{B}'$, respectively.

G.3. Bound for $K_3(\mathcal{X})$

We have

$$\begin{aligned}
&\|K_3(\mathcal{X})\| = \|\Delta\mathbf{T}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}]\| = \|\mathbf{T}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}] - \check{\mathbf{T}}_{\check{\mathcal{S}}}[\check{\mathfrak{P}}]\| \\
&= \| \int_{\overline{\check{\mathcal{S}}}} p^{\mu}(\mathcal{X}'|\mathcal{X})\check{\mathfrak{P}}(\mathcal{X}')\, d\mathcal{X}' - \int_{\overline{\check{\mathcal{S}}}} p^{\check{\mu}}(\mathcal{X}'|\mathcal{X})\check{\mathfrak{P}}(\mathcal{X}')\, d\mathcal{X}' \| \\
&= \| \int_{\overline{\check{\mathcal{S}}}} \left(p^{\mu}(\mathcal{X}'|\mathcal{X}) - p^{\check{\mu}}(\mathcal{X}'|\mathcal{X})\right)\check{\mathfrak{P}}(\mathcal{X}')\, d\mathcal{X}' \| \\
&\leq \int_{\overline{\check{\mathcal{S}}}} \|p^{\mu}(\mathcal{X}'|\mathcal{X}) - p^{\check{\mu}}(\mathcal{X}'|\mathcal{X})\| \|\check{\mathfrak{P}}(\mathcal{X}')\|\, d\mathcal{X}' \\
&\overset{\text{from (129)}}{\leq} \int_{\overline{\check{\mathcal{S}}}} c_2\|\mathbf{v}-\check{\mathbf{v}}\|\, d\mathcal{X}' = \gamma_3\|\mathbf{v}-\check{\mathbf{v}}\| \qquad (135)
\end{aligned}$$

where $\gamma_3 < \infty$.

Therefore, based on Equations (132)–(135), we can conclude that

$$\|\mathbb{P}(\mathcal{B}|\mathcal{X},\mu) - \mathbb{P}(\check{\mathcal{B}}|\mathcal{X},\check{\mu})\| \leq \gamma\|\mathbf{v}-\check{\mathbf{v}}\| \qquad (136)$$

where $\gamma = c(\gamma_1+\gamma_2+\gamma_3) < \infty$, which completes the proof that the absorption probability under the controller μ is continuous in the PRM node $\mathbf{v}$. □

H. Proof of Theorem 1

Before starting the proof of Theorem 1, we state the following proposition that concludes the continuity of the success probability of π (overall planner) given the continuity of the success probability of μ^{ij} (the individual local planners).

Proposition 2. (Continuity of success probability of π) The success probability $\mathbb{P}(\text{success}|b_0,\pi)$ is continuous in $\mathcal{V}$ if the absorption probabilities $\mathbb{P}(B^j|b,\mu^{ij})$ are continuous in $\mathbf{v}^j$ for all i, j, and b.

Proof. Given that $\mathbb{P}(B^j|b,\mu^{ij})$ is continuous in $\mathbf{v}^j$, for all i,j, we want to show that $\mathbb{P}(\text{success}|\pi,b_0)$ is continuous in all $\mathbf{v}^j$. First, let us look at the structure of the success probability:

$$\mathbb{P}(\text{success}|b_0,\pi) = \mathbb{P}(B(\mu_0)|b_0,\mu_0)\,\mathbb{P}(\text{success}|B(\mu_0),\pi^g) \qquad (137)$$

where μ_0 is computed using Equation (34). The term $\mathbb{P}(B(\mu_0)|b_0,\mu_0)$ on the right-hand side of Equation (137) is continuous because the continuity of $\mathbb{P}(B^j|b,\mu^{ij})$ for all i,j is assumed in this proposition. Thus, we only need to show the continuity of the second term in Equation (137). Without loss of generality we can consider $B^i = B(\mu_0)$. Then, we need to show that $\mathbb{P}(\text{success}|B^i,\pi^g)$ is continuous in $\mathbf{v}^i$ for all i.

As we saw in Section 6.7, the probability of success from the ith FIRM node is as follows:

$$\mathbb{P}(\text{success}|B^i,\pi^g) = \Gamma_i^{\mathrm{T}}(I-\mathcal{Q})^{-1}\mathcal{R}_g \qquad (138)$$

Moreover, we can consider $B^{goal} = B^N$ without loss of generality; then, the (i,j)th element of matrix $\mathcal{Q}$ is $\mathcal{Q}[i,j] = \mathbb{P}(B^i|B^j, \pi^g(B^j))$, and the jth element of vector $\mathcal{R}_g$ is $\mathcal{R}_g[j] = \mathbb{P}(B^N|B^j, \pi^g(B^j))$.

Since we considered B^j as the stopping region of the local controller μ^{ij}, we have

$$\mathbb{P}(B^j|B^i, \mu^{il}) = 0, \text{ if } l \neq j \tag{139}$$

Therefore, all non-zero elements in the matrices $\mathcal{R}_g$ and $\mathcal{Q}$ are of the form $\mathbb{P}(B^j|B^i, \mu^{ij})$. Thus, given the continuity of $\mathbb{P}(B^j|b, \mu^{ij})$, the transition probability $\mathbb{P}(B^j|B^i, \mu^{ij})$ is continuous and the matrices $\mathcal{R}_g$ and $\mathcal{Q}$ are continuous. Therefore, $\mathbb{P}(\text{success}|B^i, \pi^g)$ and thus $\mathbb{P}(\text{success}|b_0, \pi)$ are continuous in underlying PRM nodes. □

Now we are ready to prove Theorem 1.

Proof. Based on the definition of probabilistic completeness under uncertainty, if there exists a successful policy $\check{\pi}$, FIRM has to find a successful policy π as the number of FIRM nodes increases unboundedly. Thus, we start by assuming that there exists a successful policy $\check{\pi} \in \Pi$ for a given initial belief b_0. Since each policy in Π is parametrized by a PRM graph, there exists a PRM with nodes $\check{\mathcal{V}} = \{\check{\mathbf{v}}^i\}_{i=1}^N$ that parametrizes the policy $\check{\pi}$. Since $\check{\pi}$ is a successful policy, we know $\mathbb{P}(\text{success}|b_0, \check{\pi}) > p_{min}$. Thus, we can define $\epsilon^* = \mathbb{P}(\text{success}|b_0, \check{\pi}) - p_{min} > 0$.

Given Assumptions 1, 2, and 3, and based on Propositions 1 and 2, we know that $\mathbb{P}(\text{success}|b_0, \pi)$ is continuous with respect to the parameters of the local planners. In other words, for any $\epsilon > 0$, there exists a $\delta > 0$ such that if $\|\mathcal{V} - \check{\mathcal{V}}\| < \delta$, then $|\mathbb{P}(\text{success}|b_0, \pi(\cdot; \mathcal{V})) - \mathbb{P}(\text{success}|b_0, \check{\pi}(\cdot; \check{\mathcal{V}}))| < \epsilon$. The notation $\|\mathcal{V} - \check{\mathcal{V}}\| < \delta$ means that $\|\mathbf{v}^i - \check{\mathbf{v}}^i\| < \delta$, for all i, or equivalently, $\mathbf{v}^i \in \check{\Omega}_i$, for all i, where $\check{\Omega}_i$ is a ball with radius δ, centred at $\check{\mathbf{v}}^i$.

Therefore, for the introduced ϵ^*, there exists a δ^* and corresponding regions $\{\check{\Omega}_i\}_{i=1}^N$ such that if we have a PRM whose nodes (or a subset of whose nodes: a subset of nodes is sufficient, because the success probability is a non-decreasing function in terms of the number of nodes) satisfy the condition $\mathbf{v}_i^* \in \check{\Omega}_i$ for all $i = 1, \ldots, N$, then the planner π parametrized by this PRM has a success probability greater than p_{min}, that is, $\mathbb{P}(\text{success}|b_0, \pi(\cdot; \mathcal{V})) > p_{min}$, and hence π is successful.

Since $\delta > 0$, the regions $\check{\Omega}_i$ have non-empty interiors. Consider a PRM with a sampling algorithm under which there is a non-zero probability of sampling in $\check{\Omega}_i$, such as uniform sampling. Then, starting with any PRM, if we increase the number of nodes, a PRM node will eventually be chosen at every $\check{\Omega}_i$ with probability one. Therefore the policy constructed based on these nodes will have a success probability greater than p_{min}; in other words, we eventually get a successful policy if one exists. Thus, FIRM is probabilistically complete under uncertainty. □

Periodic-node Graph-based Framework for Stochastic Control of Small Aerial Vehicles

Ali-akbar Agha-mohammadi, Saurav Agarwal, and Suman Chakravorty

Abstract

This paper presents a strategy for stochastic control of small aerial vehicles under uncertainty using graph-based methods. In planning with graph-based methods, such as the Probabilistic Roadmap Method (PRM) in state space or the Information RoadMaps (IRM) in information-state (belief) space, the local planners (along the edges) are responsible to drive the state/belief to the final node of the edge. However, for aerial vehicles with minimum velocity constraints, driving the system belief to a sampled belief is a challenge. In this paper, we propose a novel method based on periodic controllers, in which instead of stabilizing the belief to a predefined probability distribution, the belief is stabilized to an orbit (periodic path) of probability distributions. Choosing nodes along these orbits, the node reachability in belief space is achieved and we can form a graph in belief space that can handle higher-order-dynamics or non-stoppable systems (whose velocity cannot be zero), such as fixed-wing aircraft. The proposed method takes obstacles into account and provides a query-independent graph, since its edge costs are independent of each other. Thus, it satisfies the principle of optimality. Therefore, dynamic programming can be utilized to compute the best feedback on the graph. We demonstrate the method's performance on a unicycle robot and a six degrees of freedom small aerial vehicle.

I. Introduction

This paper is concerned with the stochastic control problem for two classes of systems: *(i)* systems with dynamics, i.e., systems whose state is composed of position and its higher order derivatives such as velocity and acceleration, and *(ii)* systems with kinodynamical constraints, in particular, systems, whose velocity cannot fall below a certain threshold (referred to as non-stoppable systems in this paper). For example, consider a control problem where the system state is composed of the position and velocity $(x, \dot{x})$ of an object. Stabilizing this system to a state $(x = a, \dot{x} = b)$ where $b \neq 0$ is not possible since to stabilize the x part to a, the $\dot{x}$ must go to zero. As an example for non-stoppable systems, consider a system whose state only consists of position x, but it has constraints on its velocity $\dot{x} > b > 0$. All fixed-wing aircraft fall into this category as their velocity cannot fall below some threshold to maintain the lift requirement. Thus, stabilizing such systems to a fixed state is a challenge. This challenge gets even more difficult when this stabilization has to be achieved under uncertainty. In this paper, we propose a framework that circumvents the need for point stabilization in graph-based (roadmap-based) methods by means of stabilization to suitably designed periodic maneuvers.

Motion planning under uncertainty (MPUU) is an instance of the problem of sequential decision making under uncertainty. Considering the uncertainty in an object's motion, the problem can be framed as a stochastic control with perfect state information. In the presence of uncertainty in sensory readings, i.e., measurement noise, the state of the system is no longer available for decision making. In such a situation, a state estimation module can provide a probability distribution (referred to as information-state or belief) over all possible states of the system, and therefore decision making has to be performed in belief space. Planning in belief space in its most general form is formulated as a Partially Observable Markov Decision Process (POMDP) problem [8], [18]. However, in general solving POMDPs in continuous state, control, and observation spaces, where many robotic problems reside, is a formidable challenge.

Sampling-based motion planning methods have shown great success in dealing with many deterministic motion planning problems in complex environments and are divided into two main classes: *(i)* roadmap-based (graph-based) methods such as the Probabilistic Roadmap Method (PRM) and its variants [6], [19], [20] and *(ii)* tree-based methods such as methods in [16], [19], [22]. In deterministic settings, tree-based methods are usually single-query, i.e., their solution is valid for a given initial point whereas roadmap-based methods are mainly multi-query, i.e., the generated roadmap structure is independent of the initial point. In this sense, roadmap-based methods are a suitable choice for extension to belief space because the solution of a POMDP is feedback over the entire belief space and it does not depend on the initial belief. Accordingly, restricting the attention to a representative graph (roadmap) in the space, the feedback can be defined as a mapping from its nodes to its edges.

Similar to motion planning in state space, in belief space motion planning, the basic motion tasks can be defined as *point-to-point motion*, which deals with driving the belief of the moving object from a given belief to another given belief, and *trajectory following*, which deals with following a trajectory in belief space. Depending on the kinematics/dynamics of the system, these tasks might be very challenging in the state space. However, they often are more challenging in belief space even for simple kinematics/dynamics. To construct a query-independent roadmap in state/belief space, point-to-point motion in state/belief space is required. Feedback-based Information RoadMap (FIRM) [3], [5] extends graph-based methods to

A. Agha-mohammadi is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, 02139. S. Agarwal and S. Chakravorty are with the Dept. of Aerospace Engineering, Texas A&M University, College Station, TX, 77843, USA. Email addresses: aliagha@mit.edu, sauravag@tamu.edu, schakrav@tamu.edu.

belief space by embedding the point-to-point motion behavior in belief space using belief stabilizers (i.e. stationary feedback controllers), which was a missing behavior in pioneering works such as [17], [24], [28].

As a result of embedding the point-to-point motion behavior in belief space, FIRM generates a graph in belief space that is query independent and only needs to be constructed once offline. Establishing a connection between its solution and the original POMDP [5], it is shown that FIRM is probabilistically complete [4]. In [5] first FIRM is presented as an abstract framework for graph-based planning in belief space and then Stationary Linear Quadratic Gaussian-FIRM (SLQG-FIRM) is presented as a concrete instantiation of the abstract FIRM framework. The performance of FIRM has been demonstrated on physical mobile robots in changing environments [2]. However, SLQG-FIRM is limited to the systems that are stabilizable to stationary fixed points (with zero velocity) in the state space. This excludes the class of systems we consider in this paper.

The main contributions of this paper are:

- Proposing a graph-based solution for controlling small aerial vehicles in the presence of uncertainty and constraints. We accomplish this goal by proposing a concrete instantiation of the FIRM framework that can handle non-stoppable systems (i.e., class of dynamical systems that are not stabilizable to a point with zero-velocity), such as fixed-wing aircraft.
- Accordingly, transforming the intractable constrained POMDP to a tractable dynamic programming over a graph corresponding to non-stoppable systems.
- Designing the periodic-node PRM in state space.
- Investigating the cyclostationary behavior of the belief under Periodic Linear Quadratic Gaussian (PLQG) controllers and designing a belief stabilizer for non-stoppable systems.

This paper is organized as follows. We start by introducing the concept of periodic-node graph in state space, whose nodes lie on periodic trajectories referred to as orbit. In Section III, we review the problem of stochastic optimal control with imperfect observations. Section IV constructs the abstract FIRM framework based on the underlying periodic-node graph. In this section, we show how constraints are incorporated in the construction phase of the planner. Then, in Section V we analyze the behavior of PLQG controllers as belief stabilizers and accordingly we propose an approach to characterize and select the reachable regions in belief space under PLQG controllers. As a result we extend the periodic-node PRM from state space to a corresponding graph in belief space. We provide algorithms for offline construction of this graph and online (re)planning with this graph. Finally, in Section VI, we demonstrate the performance of the proposed method on a planar unicycle model with minimum allowable velocity and on a simplified 6DoF aerial vehicle model.

II. Periodic-Node PRM

An implicit assumption in graph-based methods such as PRM [20] is that on every edge there exists a controller to drive the robot from the start node of the edge to the end node of the edge or to an ϵ-neighborhood of the end node, for a sufficiently small $\epsilon > 0$. For a linearly controllable robot, a linear controller can locally track a PRM edge and drive the robot to its endpoint node. Obviously, controlling non-stoppable robots on a PRM roadmap is a challenge, since they have constraints on their controls and cannot reduce their velocity below a specific threshold u_{min}, and hence, stabilization is not feasible for them. This task becomes more challenging if the system is also nonholonomic. In a nonholonomic robot such as a unicycle, the linearized model at any point is not controllable, and hence, a linear controller cannot stabilize the robot to the PRM nodes. Consider the discrete unicycle model:

$$x_{k+1} = f(x_k, u_k, w_k) = \begin{pmatrix} \mathsf{x}_k + (V_k + n_v)\delta t \cos\theta_k \\ \mathsf{y}_k + (V_k + n_v)\delta t \sin\theta_k \\ \theta_k + (\omega_k + n_\omega)\delta t \end{pmatrix}, \quad w_k \sim \mathcal{N}(0, \mathbf{Q}_k) \tag{1}$$

where $x_k = (\mathsf{x}_k, \mathsf{y}_k, \theta_k)^T$ describes the robot state, in which $(\mathsf{x}_k, \mathsf{y}_k)^T$ is the 2D position of the robot and θ_k is the heading angle of the robot, at time step k. The vector $u_k = (V_k, \omega_k)^T$ is the control vector consisting of linear velocity V_k and angular velocity ω_k. The motion noise vector is denoted by $w_k = (n_v, n_\omega)^T$. Linearizing this system about the point (node) $\mathbf{v} = (\mathsf{x}^p, \mathsf{y}^p, \theta^p)$, nominal control $u^p = (V^p, \omega^p)$, and zero noise, we get:

$$x_{k+1} = f(x_k, u_k, w_k) \approx f(\mathbf{v}, u^p, 0) + \mathbf{A}(x_k - \mathbf{v}) + \mathbf{B}(u_k - u^p) + \mathbf{G}w_k \tag{2}$$

where $\mathbf{A} = \frac{\partial f}{\partial x}(\mathbf{v}, u^p, 0)$, $\mathbf{B} = \frac{\partial f}{\partial u}(\mathbf{v}, u^p, 0)$, $\mathbf{G} = \frac{\partial f}{\partial w}(\mathbf{v}, u^p, 0)$. Checking the rank of the controllability matrix of the linearized system, we get: $rank([\mathbf{B}, \mathbf{AB}, \mathbf{A^2B}]) = 2 + \mathbb{I}(V^p > 0)$, where $\mathbb{I}$ is the indicator function, which is one if $V^p > 0$ and is zero, otherwise. Therefore, if the nominal control is zero, i.e., $u^p = (V^p, \omega^p)^T = (0, 0)^T$, which is the case when we stabilize the robot to a PRM node, the resulting linear system is not controllable, since $rank([\mathbf{B}, \mathbf{AB}, \mathbf{A^2B}]) = 2 < 3$. Thus, a linear controller cannot stabilize the unicycle to a PRM node. Moreover, based on the necessary condition in Brockett's paper [12], even a smooth time-invariant nonlinear control law cannot drive the unicycle to a PRM node, and the stabilizing controller has to be either discontinuous and/or time-varying.

On roadmaps in belief space, the situation is even more complicated, since the controller has to drive the probability distribution over the state to the ϵ-neighborhood of a belief node in belief space. Again, if the linearized system in (2)

is controllable, using a linear stochastic controller such as the stationary LQG controller, one can drive the robot belief to the belief node [5]. However, if the system is non-stoppable and/or its linearized model is not controllable, the belief stabilization, if possible, is much more difficult than state stabilization.

A. Periodic-node PRM

In this paper, we circumvent the problem of stabilization to graph nodes by designing a variant of PRM, referred to as Periodic-Node PRM (PNPRM). Although there are different ways to address this problem in state space, the critical property of PNPRM is that it can be extended to belief space to form a graph whose nodes are beliefs that are reachable without a point-stabilization process. Let us denote the motion model with $x_{k+1} = f(x_k, u_k, w_k)$, where state, control, and process noise at the k-th time step are denoted by x_k, u_k, and w_k, respectively.

Similar to traditional PRM, PNPRM also consists of nodes and edges. However, in PNPRM, the nodes lie on small T-periodic trajectories (trajectories with period T) in the state space, referred to as orbits. Each orbit satisfies the control constraints and non-holonomic constraints of the moving robot. To construct a PNPRM, we first sample a set of orbits in the state space, and then on each orbit, a number of state nodes are selected. Let us denote the j-th orbit trajectory by $O^j := (x_k^{p^j}, u_k^{p^j})_{k\geq 0}$, where $x_{k+1}^{p^j} = f(x_k^{p^j}, u_k^{p^j}, 0)$, $x_{k+T}^{p^j} = x_k^{p^j}$, and $u_{k+T}^{p^j} = u_k^{p^j}$. The set of PNPRM nodes that are chosen on O^j is denoted by $\mathbb{V}^j = \{\mathbf{v}_1^j, \mathbf{v}_2^j, \cdots, \mathbf{v}_m^j\}$ where $\mathbf{v}_\alpha^j = x_{k_\alpha}^{p^j}$ for some $k_\alpha \in \{1, \cdots, T\}$. Edges in PNPRM do not connect nodes to nodes, but they connect orbits to orbits in a way that respects all the control constraints and nonholonomic constraints. Thus, the (i, j)-th edge denoted by $\mathbf{e}^{ij}$ connects O^i to O^j.

As a result, a node $\mathbf{v}_\alpha^i$ is connected to the node $\mathbf{v}_\gamma^j$ through concatenation of three path segments: *i)* the first segment is a part of O^i that connects $\mathbf{v}_\alpha^i$ to the starting point of $\mathbf{e}^{ij}$. This part is called *pre-edge* and is denoted by $\mathbf{e}^{i_\alpha j}$, *ii)* the second segment is the edge $\mathbf{e}^{ij}$ itself that connects O^i to O^j, and *iii)* the third segment is a part of O^j that connects the ending point of $\mathbf{e}^{ij}$ to $\mathbf{v}_\gamma^j$. This part is called *post-edge* and is denoted by $\mathbf{e}^{ij_\gamma}$.

One form of constructing orbits is based on circular periodic trajectories, where the edges are the lines that are tangent to the orbits. Figure 1 shows a simple PNPRM with three orbits O^i, O^r, and O^j. On each orbit four nodes are selected which are drawn (dots) with different colors. Edges $\mathbf{e}^{ij}$ and $\mathbf{e}^{rj}$ connect the corresponding orbits.

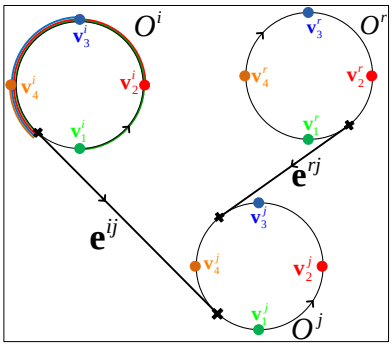


Fig. 1. A simple PNPRM with three orbits, twelve nodes, and two edges.

III. Planning in Information Space

Partially Observable Markov Decision Processes (POMDPs) are the most general formulation for motion planning problems under motion and sensing uncertainties. Note that in this paper, the environment map is assumed to be known. The solution of the POMDP problem is an optimal feedback law (mapping) π, which maps the information (belief) space to the control space. Let us denote the state, control and observation at time step k by x_k, u_k, and z_k, respectively, which belong to spaces $\mathbb{X}$, $\mathbb{U}$, and $\mathbb{Z}$, respectively. The belief in stochastic setting is defined as the probability distribution function (pdf) of the system state conditioned on the obtained measurements and applied controls up to the k-th time step, i.e., $b_k := p(x_k | z_{0:k}; u_{0:k-1})$ and $\mathbb{B}$ denotes the belief space, containing all possible beliefs. Note that $z_{0:k} = \{z_1, z_2, \cdots, z_k\}$ and $u_{0:k-1} = \{u_1, u_2, \cdots, u_{k-1}\}$. It is well known that the POMDP problem can be posed as a Markov Decision Process (MDP) in belief space [9], [27], whose solution π is computed by solving the following Dynamic Programming (DP) equation:

$$J(b) = \min_u \{c(b, u) + \int_{\mathbb{B}} p(b'|b, u) J(b') db'\}, \quad \forall b \in \mathbb{B} \tag{3a}$$

$$\pi(b) = \arg\min_u \{c(b, u) + \int_{\mathbb{B}} p(b'|b, u) J(b') db'\}, \quad \forall b \in \mathbb{B} \tag{3b}$$

where $J(\cdot) : \mathbb{B} \to \mathbb{R}$ is the optimal cost-to-go function, $p(b'|b, u)$ is the belief transition pdf under control u, and $c(b, u)$ is the one-step cost of taking control u at belief b.

IV. FIRM Framework Based on PNPRM

It is well known that the above DP equation is exceedingly difficult to solve since it is defined over an infinite-dimensional belief space. In this section, inspired by sampling-based methods, we build a graph in belief space by sampling beliefs from belief space and connecting them to each other. Hence we reduce the intractable DP in (3) to a tractable DP over this graph.

Graph nodes: Let us denote the nodes and edges of the underlying PNPRM by $\mathcal{V} = \cup_{j=1}^{n} \mathbb{V}^j = \cup_{j=1}^{n} \{\mathbf{v}_\alpha^j\}_{\alpha=1}^{m}$ and $\mathcal{E} = \{\mathbf{e}_{ij}\}$, respectively. Corresponding to each PNPRM node $\mathbf{v}_\alpha^i$, we have a unique belief $\grave{b}_\alpha^j$ whose reachability can be guaranteed utilizing appropriate feedback controllers. A concrete example of designing such a controller and computing $\grave{b}_\alpha^j$ will be provided in Section V. We define the j-th graph node in belief space (or FIRM node) B^j as a neighborhood around $\grave{b}_\alpha^j$; i.e., $B_\alpha^j = \{b : \|b - \grave{b}_\alpha^j\| \leq \epsilon\}$. The set of FIRM nodes that correspond to the i-th orbit is denoted by $\mathbb{V}^i = \{B_\alpha^i\}_{\alpha=1}^{m}$ and the set of all FIRM nodes is $\mathbb{V} = \cup_{i=1}^{n} \mathbb{V}^i$. Figure 2 shows an example set of Gaussian $\grave{b}_\alpha^j$'s corresponding to the PNPRM nodes in Fig. 1. In Gaussian case each belief b is characterized by its mean $\hat{x}^+$ and covariance P, denoted by $b \equiv (\hat{x}^+, P)$. In Fig. 2, the mean part of $\grave{b}_\alpha^j$'s is assumed to coincide with the underlying PNPRM node and the covariance part is shown by its 3σ ellipse. Also FIRM node B_2^j (neighborhood of $\grave{b}_2^j$) is shown.

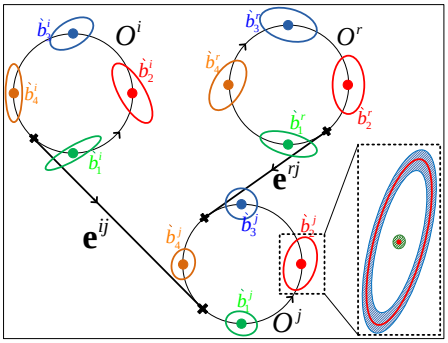


Fig. 2. $\grave{b}_\alpha^l \equiv (\mathbf{v}_\alpha^l, \check{P}_{k_\alpha}^l)$ is the center of belief nodes corresponding to the nodes shown in Fig. 1, where $\check{P}_{k_\alpha}^l$'s are shown by their 3σ-ellipse. As an example of a FIRM node, the magnified version of B_2^j, which is a small neighborhood centered at $\grave{b}_2^j$, is shown in the dotted box, where the blue shaded region depicts the covariance neighborhood and green shaded region depicts the mean neighborhood.

Graph edges: Each graph edge in belief space is a local feedback controller $\mu(\cdot) : \mathbb{B} \to \mathbb{U}$. The role of (i_α, j)-th local controller, denoted by $\mu^{\alpha,ij}$, is to take the belief from the FIRM node B_α^i to a FIRM node on orbit O^j, i.e., to $\cup_\gamma B_\gamma^j$. Thus, we define $\mathcal{T}^{\alpha,ij} := \min\{k \geq 0, b_k \in \cup_\gamma B_\gamma^j | b_0 = \grave{b}_\alpha^j, \mu^{\alpha,ij}\} \in [0, \infty]$ as the stopping time of the controller $\mu^{\alpha,ij}$. The stopping time is a random variable that defines the time it takes for the controller to drive the belief from the initial node to the target orbit. Also, let the $\mathbb{P}(A|b, \mu)$ be the probability of reaching set A in finite time under the local controller μ starting from belief b. Therefore, for a local controller $\mu^{\alpha,ij}$ to act as a graph edge, it has to satisfy $\mathbb{P}(\cup_\gamma B_\gamma^j | \grave{b}_\alpha^i, \mu^{\alpha,ij}) = \Pr(\mathcal{T}^{\alpha,ij} < \infty) = 1$ in the absence of obstacles. In other words, in a constraint-free environment, the feedback controller $\mu^{\alpha,ij}(\cdot)$ has to drive the system's belief from B_α^i into a $B \in \mathbb{V}^j$ in finite time with probability one.

In this section, it is assumed that a set of edges (local controllers) that satisfy the mentioned reachability property is given. In Section V we show that the above property can be accomplished using periodic LQG controller for the class of non-stoppable/nonholonomic systems, such as small aerial vehicles. Accordingly, we provide concrete algorithms to construct local controllers and their corresponding reachable nodes.

Graph in belief space: Formally, we define the constructed graph as $G = (\mathbb{V}, \mathbb{M})$ with the set of nodes $\mathbb{V} = \{B_\alpha^j\}$ and the set of edges $\mathbb{M} = \{\mu^{\alpha,ij}\}$. The set of edges available (i.e., outgoing) at FIRM node B_α^i is denoted by $\mathbb{M}(i, \alpha) := \{\mu^{\alpha,ij} \in \mathbb{M} | \exists \mathbf{e}_{ij} \in \mathcal{E}\}$. It is worth noting that the planning is still performed over continuous state, control, and observation spaces and we do not discretize any of those.

Graph transition cost and probabilities: We generalize the one-step transition costs $c(b, u)$ and probabilities to the cost of taking a controller in a graph node and its corresponding transition probabilities along the graph edges:

$$C^g(B_\alpha^i, \mu^{\alpha,ij}) := \sum_{k=0}^{\mathcal{T}^{\alpha,ij}} c(b_k, \mu^{\alpha,ij}(b_k) | b_0 = \grave{b}_\alpha^i) \approx \sum_{k=0}^{\mathcal{T}^{\alpha,ij}} c(b_k, \mu^{\alpha,ij}(b_k) | b_0 = b), \quad \forall b \in B_\alpha^i \tag{4a}$$

$$\mathbb{P}^g(B_\gamma^j | S_\alpha^i, \mu^{\alpha,ij}) := \mathbb{P}(B_\gamma^j | \grave{b}_\alpha^i, \mu^{\alpha,ij}) \approx \mathbb{P}(B_\gamma^j | b, \mu^{\alpha,ij}), \quad \forall b \in B_\alpha^i \tag{4b}$$

The "piecewise constant approximation" in (4) is an arbitrarily good approximation for sufficiently small B_α^i and smooth cost function and transition probabilities.

Graph policy: Graph policy $\pi^g : \mathbb{V} \to \mathbb{M}$ is a function that returns a local controller for any given node of the graph. We denote the space of all graph policies by Π^g.

Graph cost-to-go: To choose the best graph policy, we define the graph cost-to-go J^g from every graph node. Let $B_{\overline{k}}$ be the $\overline{k}$-th FIRM node visited along the plan. Then, we can formally define the cost-to-go from any node $B_0 \in \mathbb{V}$ as:

$$\begin{aligned} J^g(B_0;\pi^g) &= \sum_{\overline{k}=0}^{\infty} \mathbb{E}\left[C^g(B_{\overline{k}}, \pi^g(B_{\overline{k}}))\right] \\ s.t. \quad & B_{\overline{k+1}} \sim \mathbb{P}^g(B_{\overline{k+1}}|B_{\overline{k}}, \pi^g(B_{\overline{k}})) \end{aligned} \tag{5}$$

Accordingly, the MDP defined on the graph is as follows:

$$\begin{aligned} \pi^{g^*} &= \arg\min_{\Pi^g} \sum_{\overline{k}=0}^{\infty} \mathbb{E}\left[C^g(B_{\overline{k}}, \pi^g(B_{\overline{k}}))\right] \\ s.t. \quad & B_{\overline{k+1}} \sim \mathbb{P}^g(B_{\overline{k+1}}|B_{\overline{k}}, \pi^g(B_{\overline{k}})) \end{aligned} \tag{6}$$

Obstacle-free graph DP: Since the graph MDP is defined on a finite number of FIRM nodes, we can form a tractable Dynamic Programming (DP) to find the optimal graph policy:

$$J^g(B^i_\alpha) = \min_{\mu^{\alpha,ij}\in\mathbb{M}(i,\alpha)} C^g(B^i_\alpha, \mu^{\alpha,ij}) + \sum_{\gamma=1}^{m} J^g(B^j_\gamma)\mathbb{P}^g(B^j_\gamma|B^i_\alpha, \mu^{\alpha,ij}),\ \forall\alpha,i,j \tag{7a}$$

$$\pi^g(B^i_\alpha) = \arg\min_{\mu^{\alpha,ij}\in\mathbb{M}(i,\alpha)} C^g(B^i_\alpha, \mu^{\alpha,ij}) + \sum_{\gamma=1}^{m} J^g(B^j_\gamma)\mathbb{P}^g(B^j_\gamma|B^i_\alpha, \mu^{\alpha,ij}),\ \forall\alpha,i,j \tag{7b}$$

where $J^g(\cdot) := \min_{\pi^g} J(\cdot;\pi^g)$ is the optimal cost-to-go.

Incorporating obstacles into planning: In the presence of obstacles, we cannot assure that the local controller $\mu^{\alpha,ij}(\cdot)$ can drive any $b \in B^i_\alpha$ into $\cup_\gamma B^j_\gamma$ with probability one. Instead, we specify the failure probabilities that the robot collides with an obstacle. Let us denote the failure set on $\mathbb{X}$ by F (i.e., $F = \mathbb{X} - \mathbb{X}_{free}$). Let $\mathbb{P}(F|B^i_\alpha, \mu^{\alpha,ij}) := \mathbb{P}(F|\grave{b}^i_\alpha, \mu^{\alpha,ij})$ denote the probability of hitting the failure set under local controller $\mu^{\alpha,ij}$ starting from B^i_α. Similarly, we generalize the cost-to-go function by defining $J^g(F)$ as a user-defined suitably high cost for hitting obstacles. Therefore, we can modify (7) to incorporate obstacles in the state space as follows:

$$\begin{aligned} J^g(B^i_\alpha) = \min_{\mu^{\alpha,ij}\in\mathbb{M}(i,\alpha)} & C^g(B^i_\alpha, \mu^{\alpha,ij}) + J^g(F)\mathbb{P}^g(F|B^i_\alpha, \mu^{\alpha,ij}) \\ & + \sum_{\gamma=1}^{m} J^g(B^j_\gamma)\mathbb{P}^g(B^j_\gamma|B^i_\alpha, \mu^{\alpha,ij}),\ \forall\alpha,i,j \end{aligned} \tag{8a}$$

$$\begin{aligned} \pi^g(B^i_\alpha) = \arg\min_{\mu^{\alpha,ij}\in\mathbb{M}(i,\alpha)} & C^g(B^i_\alpha, \mu^{\alpha,ij}) + J^g(F)\mathbb{P}^g(F|B^i_\alpha, \mu^{\alpha,ij}) \\ & + \sum_{\gamma=1}^{m} J^g(B^j_\gamma)\mathbb{P}^g(B^j_\gamma|B^i_\alpha, \mu^{\alpha,ij}),\ \forall\alpha,i,j \end{aligned} \tag{8b}$$

Thus, all that is required to solve the above DP equation are the values of the costs $C^g(B^i_\alpha, \mu^{\alpha,ij})$ and transition probability functions $\mathbb{P}^g(\cdot|B^i_\alpha, \mu^{\alpha,ij})$, which are discussed in Section V.

Overall policy π: The overall feedback π is generated by combining the policy π^g on the graph and the local controllers $\mu^{\alpha,ij}$s. However, this combination leads to a non-Markov policy. More rigorously, the resulting policy is a semi-Markov policy [26]. In other words, the current action depends on the current belief as well as the last visited FIRM node. Thus, the overall feedback $\pi : \mathbb{V}\times\mathbb{B} \to \mathbb{U}$ can be written as:

$$\pi(B, b) = \pi^g(B)(b) = \mu(b). \tag{9}$$

Initial controller: Now, let us consider the first step of planning where the system has not visited any FIRM node yet. Given the initial belief is b_0, if b_0 is in a FIRM node B, then we can just generate the control signal as $\pi(B, b_0)$ based on Eq. 9. However, if b_0 does not belong to any of the FIRM nodes, we consider a singleton FIRM node $B_0 = \{b_0\}$ and connect it to the graph. Let us denote the set of newly added local controllers by $\mathbb{M}(0)$. Computing the transition cost $C(b_0, \mu^{ij})$, and probabilities $\mathbb{P}(B^j_\gamma|b_0, \mu^{ij})$, and $\mathbb{P}(F|b_0, \mu^{ij})$, for invoking local controllers $\mu^{ij} \in \mathbb{M}(0)$ at b_0, we choose the best initial controller μ^0_* as:

$$\pi^g(B_0) = \mu^0_* = \arg\min_{\mu^{ij}\in\mathbb{M}(0)} \{C^g(B_0, \mu^{ij}) + \sum_{\gamma=1}^{m} \mathbb{P}^g(B^j_\gamma|B_0, \mu^{ij})J^g(B^j_\gamma) + \mathbb{P}^g(F|B_0, \mu^{ij})J^g(F)\} \tag{10}$$

Extending π^g to take B_0 into account, we now can use $\pi(B_0, b_0)$ to generate the control signal. It is worth noting that computing μ^0_* is the only part of computation that depends on the initial belief and has to be reproduced for every query with

a new initial belief. After computing μ_*^0 we always store the last visited FIRM node and use policy π (computed offline) in Eq. 9 to generate control signals in future time steps.

V. PLQG-based FIRM Construction

In this section, we construct a concrete instantiation of the graph described in the previous section. We utilize PLQG controllers to design graph edges and reachable FIRM nodes B_γ^j required in (8). Then we discuss how the transition probabilities $\mathbb{P}^g(\cdot|B_\alpha^i, \mu^{\alpha,ij})$, and costs $C^g(B_\alpha^i, \mu^{\alpha,ij})$ in (8) are computed. In this instantiation we restrict the belief to Gaussian distributions and we start by defining notation needed for dealing with Gaussian beliefs.

Gaussian belief space: Let us denote the estimation vector by x^+, whose distribution is $b_k = p(x_k^+) = p(x_k|z_{0:k})$. Denote the mean and covariance of x^+ by $\widehat{x}^+ = \mathbb{E}[x^+]$ and $P = \mathbb{E}[(x^+ - \widehat{x}^+)(x^+ - \widehat{x}^+)^T]$, respectively. Denoting the Gaussian belief space by $\mathbb{GB}$, every function $b(\cdot) \in \mathbb{GB}$, can be characterized by a mean-covariance pair, i.e., $b \equiv (\widehat{x}^+, P)$. Abusing notation, we also show this using "equality relation", i.e., $b = (\widehat{x}^+, P)$.

A. Designing PLQG-based Graph Nodes $\{B_\alpha^j\}$

LQG controllers: A Linear Quadratic Gaussian (LQG) controller is composed of a Kalman filter as the state estimator and a Linear Quadratic Regulator (LQR) as the separated controller [21]. Thus, the belief dynamics $b_{k+1} = \tau(b_k, u_k, z_{k+1})$ come from the Kalman filtering equations, and the controller $u_k = \mu(b_k)$ that acts on the belief, comes from the LQR equations. LQG is an optimal controller for linear systems with Gaussian noise [9]. However, it is most often used for stabilizing nonlinear systems to a given trajectory or to a given point.

Periodic LQG: Periodic LQG (PLQG) is a time-varying LQG that is designed to track a given periodic trajectory [11], [13]. In Appendix I we review the periodic LQG controller in detail. Here, we only state the belief reachability result under the PLQG.

System model and quadratic cost: Consider a T-periodic PNPRM orbit $O = (x_k^p, u_k^p)_{k\geq 1}$ and the set of nodes $\{\mathbf{v}_\alpha\}$ on it. Let us denote the time-varying linear (linearized) system along the orbit O by the tuple $\Upsilon_k = (\mathbf{A}_k, \mathbf{B}_k, \mathbf{G}_k, \mathbf{Q}_k, \mathbf{H}_k, \mathbf{M}_k, \mathbf{R}_k)$ that represents the following state space model, where $\Upsilon_k = \Upsilon_{k+T}$:

$$x_{k+1} = \mathbf{A}_k x_k + \mathbf{B}_k u_k + \mathbf{G}_k w_k, \quad w_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k) \tag{11a}$$

$$z_k = \mathbf{H}_k x_k + \mathbf{M}_k v_k, \qquad v_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k). \tag{11b}$$

Consider a PLQG controller that is designed for the system in (11) to track the orbit $(x_k^p, u_k^p)_{k\geq 1}$ through minimizing the following quadratic cost:

$$J = \mathbb{E}[\sum_{k\geq 0} \overline{x}_k^T \mathbf{W}_x \overline{x}_k + \overline{u}_k^T \mathbf{W}_u \overline{u}_k], \tag{12}$$

where $\overline{x}_k = x_k - x_k^p$ and $\overline{u}_k = u_k - u_k^p$. Matrices $\mathbf{W}_x$ and $\mathbf{W}_u$ are positive definite weight matrices for state and control cost, respectively. Let us also define matrices $\check{\mathbf{Q}}_k$ and $\check{\mathbf{W}}_x$ such that $\mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T = \check{\mathbf{Q}}_k\check{\mathbf{Q}}_k^T$, $\mathbf{W}_x = \check{\mathbf{W}}_x^T\check{\mathbf{W}}_x$, for all k. Now, consider the class of systems, and associated PLQG controllers that satisfy the following property.

Property 1: The pairs $(\mathbf{A}_k, \mathbf{B}_k)$ and $(\mathbf{A}_k, \check{\mathbf{Q}}_k)$ are controllable pairs [9], and the pairs $(\mathbf{A}_k, \mathbf{H}_k)$ and $(\mathbf{A}_k, \check{\mathbf{W}}_x)$ are observable pairs [9], for all $k = 1, \cdots, T$.

Belief node reachability under PLQG: In the following, we present three lemmas, through which we can construct pairs of periodic LQG controllers, and reachable nodes in belief space, for non-stoppable/nonholonomic dynamical systems.

Lemma 1: **(Cyclostationary behavior of belief under PLQG)** Consider the PLQG controller designed for the system in (11) to track the orbit $(x_k^p, u_k^p)_{k\geq 1}$. Given Property 1 is satisfied, the belief process b_k under PLQG converges to a Gaussian cyclostationary process [10], i.e., the distribution over belief converges to a T-periodic Gaussian distribution, where we denote the mean and covariance of this process by b_k^c and $\mathtt{C}_k$, respectively:

$$b_k \sim \mathcal{N}(b_k^c, \mathtt{C}_k) = \mathcal{N}(b_{k+T}^c, \mathtt{C}_{k+T}), \tag{13}$$

where $b_k \equiv (\widehat{x}_k^+, P_k)$ and $b_k^c \equiv (x_k^p, \check{P}_k)$. The covariance matrices $\check{P}_k$ is characterized in Lemma 2 and covariance $\mathtt{C}_k$ is characterized in Appendix I (Eq. 68).

Proof: See Appendix I. ∎

Lemma 2: **(Convergence of DPRE)** Given Property 1, the following Discrete Periodic Riccati Equation (DPRE) has a unique Symmetric T-Periodic Positive Semi-definite (SPPS) solution [11], denoted by $\check{P}_k^-$:

$$\check{P}_{k+1}^- = \mathbf{A}_k(\check{P}_k^- - \check{P}_k^-\mathbf{H}_k^T(\mathbf{H}_k\check{P}_k^-\mathbf{H}_k^T + \mathbf{M}_k\mathbf{R}_k\mathbf{M}_k^T)^{-1}\mathbf{H}_k\check{P}_k^-)\mathbf{A}_k^T + \mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T \tag{14}$$

Moreover, the covariance matrix $\check{P}_k$ introduced in Lemma 1 is computed as

$$\check{P}_k = \check{P}_k^- - \check{P}_k^-\mathbf{H}_k^T(\mathbf{H}_k\check{P}_k^-\mathbf{H}_k^T + \mathbf{M}_k\mathbf{R}_k\mathbf{M}_k^T)^{-1}\mathbf{H}_k\check{P}_k^- \tag{15}$$

Proof: See [11]. ∎

Now, we state the main result, through which we can construct the proper pairs of periodic LQG controller and reachable nodes in belief space.

Lemma 3: **(Belief node reachability under PLQG)** Consider the PLQG controller μ designed for the system in (11) to track the orbit $(x_k^p, u_k^p)_{k\geq 1}$. Suppose the matrix $\mathbf{H}_k$ is full rank, and Property 1 is satisfied. Also, consider the sets $B_1, B_2, \cdots, B_m$ in belief space, such that interior of B_α contains $b_{k_\alpha}^c$ for some $k_\alpha \in \{1, \cdots, T\}$. Then, under μ, the region $\cup_\alpha B_\alpha$ is reachable in finite time with probability one.

Proof: The intuitive idea behind the proof is: if we define a region centered at the mean value of a Gaussian distribution, and if we sample from this distribution, in a finite number of samples we will end up with a sample in the given region. The rigorous proof is detailed in Appendix II. ∎

FIRM nodes: As mentioned, to construct a graph in belief space we first construct its underlying PNPRM, characterized by the triple $\{\{O^j\}, \{\mathbf{v}_\alpha^j\}, \{\mathbf{e}_{ij}\}\}$. Linearizing the system along the j-th orbit $O^j = (x_k^{p^j}, u_k^{p^j})_{k\geq 0}$ results in a time-varying T-periodic system $\Upsilon_k^j = (\mathbf{A}_k^j, \mathbf{B}_k^j, \mathbf{G}_k^j, \mathbf{Q}_k^j, \mathbf{H}_k^j, \mathbf{M}_k^j, \mathbf{R}_k^j)$:

$$x_{k+1} = \mathbf{A}_k^j x_k + \mathbf{B}_k^j u_k + \mathbf{G}_k^j w_k, \quad w_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k^j) \tag{16a}$$

$$z_k = \mathbf{H}_k^j x_k + \mathbf{M}_k^j v_k, \qquad v_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k^j), \tag{16b}$$

where w_k and v_k are motion and measurement noises, respectively, drawn from zero-mean Gaussian distributions with covariances $\mathbf{Q}_k^j$ and $\mathbf{R}_k^j$. Since the system in (16) is T-periodic (i.e.,$\Upsilon_k^j = \Upsilon_{k+T}^j$), we can design a corresponding PLQG controller μ_k^j. The controller μ_k^j is referred to as the j-th *node-controller*. Since the orbits are designed such that Property 1 is satisfied on them, based on Lemma 1 the belief converges to a Gaussian cyclostationary process. The mean of this cyclostationary process is denoted by $b_k^{c^j}$ and is characterized in Lemma 2, where its existence and uniqueness are guaranteed.

Corresponding to the PNPRM node $\mathbf{v}_\alpha^j$ on orbit O^j we choose the belief nodes B_α^j as an ϵ-neighborhood of $\grave{b}_\alpha^j := b_{k_\alpha}^{c^j} \equiv (\mathbf{v}_\alpha^j, \check{P}_{k_\alpha}^j)$: (See Fig.2.)

$$B_\alpha^j = \{b \equiv (x, P) : \|x - \mathbf{v}_\alpha^j\| < \delta_1, \|P - \check{P}_{k_\alpha}^j\|_m < \delta_2\}, \tag{17}$$

where $\|\cdot\|$ and $\|\cdot\|_m$ denote suitable vector and matrix norms, respectively. The size of FIRM nodes are determined by δ_1 and δ_2. Based on Lemma 3, $\cup_\alpha B_\alpha^j$ is a reachable region under node-controller μ_k^j. Note that δ_1 and δ_2 need to be sufficiently small to satisfy the approximation in (4).

B. PLQG-based Graph Edges $\{\mu^{\alpha,ij}\}$

The role of the local controller $\mu^{\alpha,ij}$ is to drive the belief from the node B_α^i to $\cup_\gamma B_\gamma^j$, i.e., to a node B_γ^j on the j-th orbit. To construct the local controller $\mu^{\alpha,ij}$, we precede the node-controller μ_k^j, with a time-varying LQG controller $\overline{\mu}_k^{\alpha,ij}$, which is called the *edge-controller* here.

Edge-controller: Consider a finite trajectory that consists of three segments: *i)* the pre-edge $\mathbf{e}^{i_\alpha j}$ as defined in Section II, *ii)* the edge itself $\mathbf{e}^{ij}$, and *iii)* a part of O^j that connects the ending point of $\mathbf{e}^{ij}$ to $x_0^{p^j}$. Edge-controller $\overline{\mu}_k^{i_\alpha j}$ is a time-varying LQG controller that is designed to track this finite trajectory. The main role of the edge-controller is that it takes the belief at node B_i and drives it to the vicinity of a starting point of orbit O^j, where it hands over the system to the node-controller, and node-controller in turn takes the system to a FIRM node.

Local controllers: Thus, overall, the local controller (or graph edge in belief space) $\mu^{\alpha,ij}$ is the concatenation of the edge-controller $\overline{\mu}_k^{\alpha,ij}$ and the node-controller μ_k^j. Note that since reachability is guaranteed by the node-controller (PLQG), by this construction, the stopping region $\cup_\gamma B_\gamma^j$ is also reachable under the local controller $\mu^{\alpha,ij}$.

C. Transition Probabilities and Costs

In general, it can be a computationally expensive task to compute the transition probabilities $\mathbb{P}(\cdot|B_\alpha^i, \mu^{\alpha,ij})$ and costs $C(B_\alpha^i, \mu^{\alpha,ij})$ associated with invoking local controller $\mu^{\alpha,ij}$ at node B_α^i. However, owing to the offline construction of FIRM, it is not an issue in FIRM. We utilize sequential Monte-Carlo methods [15] to compute the collision and absorption probabilities. In other words, for each graph edge we simulate the execution of the corresponding local controller for M times and accordingly approximate the probability of reaching the nodes on the target orbit as well as probability of hitting the failure set along the way. This process is done offline.

Depending on the application, a suitable transition cost can be defined. In this paper, we consider a measure of estimation accuracy as the transition cost along the edges. This leads to a planner that favors paths, on which the estimator and consequently the controller can perform better. A measure of estimation error we use here is the trace of estimation covariance; i.e., $\Phi^{\alpha,ij} = \mathbb{E}[\sum_{k=1}^{\mathcal{T}} \mathrm{tr}(P_k^{\alpha,ij})]$, where $P_k^{\alpha,ij}$ is the estimation covariance at the k-th time step of the execution of local controller $\mu^{\alpha,ij}$. The outer expectation operator is useful in dealing with the Extended Kalman Filter (EKF), whose covariance is stochastic [14], [25]. Moreover, as we are also interested in faster paths, we take into account the corresponding mean stopping time, i.e., $\widehat{\mathcal{T}}^{\alpha,ij} = \mathbb{E}[\mathcal{T}^{\alpha,ij}]$, and the total cost of invoking $\mu^{\alpha,ij}$ at B_α^i is considered as a linear combination of the estimation accuracy and expected stopping time, with suitable scalar coefficients ξ_1 and ξ_2.

$$C(B_\alpha^i, \mu^{\alpha,ij}) = \xi_1 \Phi^{\alpha,ij} + \xi_2 \widehat{\mathcal{T}}^{\alpha,ij}. \tag{18}$$

D. Construction of PLQG-FIRM and Planning With it

Offline construction of FIRM: The crucial feature of FIRM is that it can be constructed offline and stored, independent of future queries. Note that based on Algorithms 1 and 2, we still need to know the goal location. However, to be fully independent of both start and goal location of the query, one can solve the DP in the online phase. Owing to the reduction from the original POMDP to a dynamic programming on the finite number of graph nodes, we can solve DP in (8) using standard DP techniques such as value/policy iteration to get the optimal graph policy π^g. Algorithm 1 details the offline construction of the FIRM graph.

Online planning with FIRM: Since the FIRM graph is computed offline, the online phase of planning (and replanning) on the roadmap becomes very efficient. If the given initial belief b_0 belongs to any FIRM node, online compuation reduces to evaluating the function π in Eq. 9. Otherwise, the only online compuation would be evaluation of the first local controller μ_*^0 based on Eq. 10. In the latter case, to form the new edges required in 10, we first create a singleton set $B_0 = \{b_0\}$. Then, to connect B_0 to FIRM, we compute the expected value of the robot state, i.e. $\mathbb{E}[x_0]$ using its distribution b_0 and add $\mathbb{E}[x_0]$ to the underlying PNPRM nodes. The set of newly added edges going from $\mathbb{E}[x_0]$ to the nodes on PNPRM are denoted as $\mathcal{E}(0)$. We design the local controllers associated with each edge in $\mathcal{E}(0)$ and call the set of them as $\mathbb{M}(0)$. Then, we choose μ_*^0 based on Eq. 10 and follow policy π in Eq. 9 afterwards. Algorithm 2 illustrates this procedure.

Computational complexity of offline graph construction: Consider an underlying PNPRM with N orbits, m nodes on each orbit, and degree k; i.e., each orbit in PNPRM is connected to k nearest neighboring orbits. Thus, overall it has mN nodes and Nk orbit edges. In the offline phase we need to leverage PNPRM orbits and edges to FIRM orbits and edges in belief space. *(i)* Extension of PNPRM orbits to belief space consists of a constant computation of solving two Riccati equations and designing corresponding PLQG controller. Denoting the computational complexity of this process by c_n, the computational complexity of extending PNPRM orbits to FIRM orbits is of the order $O(c_n N)$. *(ii)* Extension of each PNPRM edge to belief space consists of evaluating the performance of its corresponding local controller and computing transition probabilities and costs. Let us denote the cost of this process by c_e. In a PNPRM with degree k, we have Nk edges and corresponding to each PNPRM edge, we have m FIRM edges. Thus, the computational complexity of extending edges to belief space is $O(c_e mNk)$. So, overall the offline computational complexity is $O(c_n N + c_e mNk)$. The complexity of each iteration in value iteration algorithm is $O(|\mathbb{V}|^2|\mathbb{M}|)$, where $|\mathbb{V}| = mN$ nodes and $|\mathbb{M}| = mNk$. However, in practice the dominating factor is the extension of edges to belief space because the constant multiplier c_e in general is large. If the Monte Carlo simulation is chosen to evaluate the edge costs and transition probabilities, c_e will increase linearly in the number of particles utilized in the Monte Carlo simulation as well as the number of constraints. It will also depend on how the constraints are being evaluated.

Computational complexity of online planning with graph: As discussed in Section IV, the only part that needs to be done online is the computation of first local controller (See Eq. 10). To do so, we need to evaluate k edges only. Thus, the computational complexity of online planning with FIRM is $O(c_e k)$. This computation occurs once in the beginning of planning. The rest of planning is just plugging last visited FIRM node B and current belief b into the planner π (See Eq. 9) and generating the control signal u_k.

VI. Experimental Results

In this section we present simulation results for two different types of robots: a planar robot whose motion is described by a unicycle model and a 6 DoF small aerial vehicle subject to rigid body kinematics. The robots are equipped with exteroceptive sensors that provide range and bearing measurements from existing radio beacons (landmarks) in the environment.

A. 2D Unicycle Model

Here, we illustrate the results of FIRM construction on a simple PNPRM.

Motion model: As a motion model, we consider the nonholonomic unicycle model which has the following kinematics:

$$x_{k+1} = f(x_k, u_k, w_k) = \begin{pmatrix} \mathsf{x}_k + (V_k \delta t + n_v \sqrt{\delta t}) \cos\theta_k \\ \mathsf{y}_k + (V_k \delta t + n_v \sqrt{\delta t}) \sin\theta_k \\ \theta_k + \omega_k \delta t + n_\omega \sqrt{\delta t} \end{pmatrix}, \tag{19}$$

where $x_k = (\mathsf{x}_k, \mathsf{y}_k, \theta_k)^T$ describes the robot state (2D position and heading angle). The vector $u_k = (V_k, \omega_k)^T$ is the control vector consisting of linear velocity V_k and angular velocity ω_k. The motion noise vector is denoted by $w_k = (n_v, n_\omega)^T \sim \mathcal{N}(0, \mathbf{Q}_k)$.

Observation model: The i-th landmark is denoted by L^i and the vector from robot to the i-th landmark is denoted by ${}^i\mathbf{d} = [{}^i d_x, {}^i d_y]^T := L^i - \mathsf{p}$, where $\mathsf{p} = [\mathsf{x}, \mathsf{y}]^T$ is the position of the robot. Measuring L^i is modeled as follows:

$${}^i z = {}^i h(x, {}^i v) = [\|{}^i\mathbf{d}\|, \text{atan2}({}^i d_y, {}^i d_x) - \theta]^T + {}^i v, \tag{20}$$

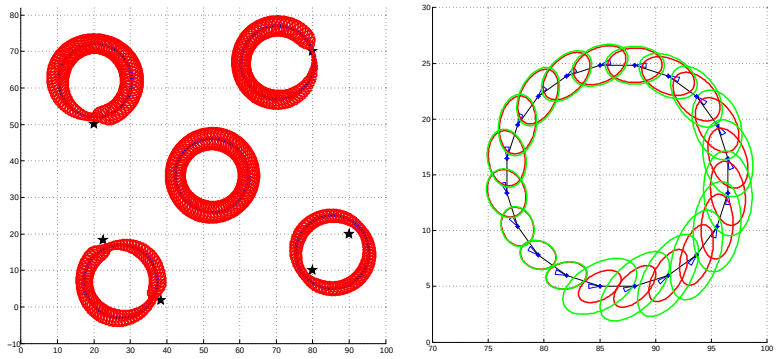
where, atan2$(\cdot,\cdot)$ is the four-quadrant inverse tangent function. Observation noise is drawn from a zero-mean Gaussian distribution ${}^i v \sim \mathcal{N}(\mathbf{0}, {}^i\mathbf{R})$ where ${}^i\mathbf{R} = \text{diag}((\eta_r \|{}^i\mathbf{d}\| + \sigma_b^r)^2, (\eta_\theta \|{}^i\mathbf{d}\| + \sigma_b^\theta)^2)$. Function "diag" returns a square block-diagonal matrix by placing its inputs on the main diagonal. The uncertainty (standard deviation) of sensor reading increases

Algorithm 1: Offline Construction of PLQG-FIRM

input : State space $\mathbb{X}$, constraints set F, belief space $\mathbb{B}$
output : FIRM graph G
Construct a PNPRM with T-periodic orbits $\mathcal{O} = \{O^j = (x_k^{p^j}, u_k^{p^j})_{k\geq 0}\}$, nodes $\mathcal{V} = \{\mathbf{v}_\alpha^j\}$, and edges $\mathcal{E} = \{\mathbf{e}^{ij}\}$, where $i, j = 1, \cdots, n$ and $\alpha = 1, \cdots, m$;
foreach *PNPRM orbit* $O^j \in \mathcal{O}$ **do**
    Design the node-controller (periodic LQG) μ_k^j along the periodic trajectory;
    Compute the periodic mean belief trajectory $b_k^{c^j} \equiv (x_k^{p^j}, \check{P}_k^j)$ using (15);
    Construct m FIRM nodes $\mathbb{V}^j = \{B_1^j, \cdots, B_m^j\}$ using (17), where B_α^j is centered at $b_{k_\alpha}^{c^j}$;
Collect all FIRM nodes $\mathbb{V} = \cup_{j=1}^n \mathbb{V}^j$;
foreach $(B_\alpha^i, \mathbf{e}^{ij})$ *pair* **do**
    Design the edge-controller $\overline{\mu}_k^{\alpha,ij}$, as discussed in Section V-B;
    Construct the local controller $\mu_k^{\alpha,ij}$ by concatenating edge-controller $\overline{\mu}_k^{\alpha,ij}$ and node-controller μ_k^j;
    Set the initial belief b_0 equal to the center of B_α^i, based on the approximation in (4);
    Generate (in simulation) sample belief paths $b_{0:\mathcal{T}}$ and state paths $x_{0:\mathcal{T}}$ induced by controller $\mu^{\alpha,ij}$ invoked at B_α^i;
    Compute the transition probabilities $\mathbb{P}^g(F|B_\alpha^i, \mu^{\alpha,ij})$ and $\mathbb{P}^g(B_\gamma^j|B_\alpha^i, \mu^{\alpha,ij})$ for all γ and transition cost $C^g(B_\alpha^i, \mu^{\alpha,ij})$ based on the simulated trajectories (see Section V-C);
Collect all local controllers $\mathbb{M} = \{\mu^{\alpha,ij}\}$;
Compute cost-to-go J^g and feedback π^g over the FIRM graph by solving the DP in (8);
$G = (\mathbb{V}, \mathbb{M}, J^g, \pi^g)$;
return G;

as the robot gets farther from the landmarks. The parameters $\eta_r = \eta_\theta = 0.3$ determine this dependency, and $\sigma_b^r = 0.01$ meter and $\sigma_b^\theta = 0.5$ degrees are the bias standard deviations. A similar model for range sensing is used in [24]. The robot observes all N_L landmarks at all times and their observation noises are independent. Thus, the total measurement vector is denoted by $z = [{}^1z^T, {}^2z^T, \cdots, {}^{N_L}z^T]^T$ and due to the independence of measurements of different landmarks, the observation model for all landmarks can be written as $z = h(x) + v$, where $v \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ and $\mathbf{R} = \text{diag}({}^1\mathbf{R}, \cdots, {}^{N_L}\mathbf{R})$.

We first show a typical SPPS solution of DPRE on the orbits. Fig. 3(a) shows a simple environment with six radio beacons (black stars). For illustration purposes, we choose five large circular orbits and every orbit is discretized to 100 steps. Thus the SPPS solution of the DPRE in (14) on each orbit leads to hundred covariance matrices that are superimposed on the graph in red. As is seen from Fig. 3(a), the localization uncertainty along the orbit is not homogeneous and varies periodically. Another important observation from the Fig. 3(a) is obtained by noticing the left top orbit in the Fig. 3(a). As it can be seen, the localization uncertainty (covariance ellipse) in the left and right hand sides of the landmark are not symmetric (the right hand side is larger than the left hand side). In other words, two points on an orbit with the same distance from landmarks (i.e., with the same observation noise) might have different localization uncertainty, which emphasizes the role of the dynamics model in filtering and its interaction with the observation model. In Fig. 3(b), we illustrate the covariance



(a) (b)

Fig. 3. (a) Five orbits ($T = 100$) and corresponding periodic estimation covariances as the SPPS solution of DPRE in (14). (b) Sample covariance convergence on an orbit ($T = 20$) under PLQG. Red ellipses are the solution of DPRE and green ellipses are the evolution of estimation covariance. The initial covariance is three times bigger than the SPPS solution of DPRE, i.e., $P_0 = 3\check{P}_0$.

Algorithm 2: Online Phase Algorithm (Planning with PLQG-based FIRM)

input : Initial belief b_0, FIRM graph G, Underlying PNPRM graph
if $\exists B^i_\alpha \in \mathbb{V}$ *such that* $b_0 \in B^i_\alpha$ **then**
  Choose the next local controller $\mu^{\alpha,ij} = \pi^g(B^i_\alpha)$;
else
  Compute $\mathbf{v}_0 = \mathbb{E}[x_0]$ based on b_0, and connect $\mathbf{v}_0$ to the PNPRM orbits. Call the set of newly added edges $\mathcal{E}(0) = \{\mathbf{e}^{0j}\}$;
  Design local planners associated with edges in $\mathcal{E}(0)$; Collect them in set $\mathbb{M}(0) = \{\mu^{0,0j}\}$;
  foreach $\mu \in \mathbb{M}(0)$ **do**
    Generate (simulate) sample belief and state paths $b_{0:\mathcal{T}}$, $x_{0:\mathcal{T}}$ induced by taking μ at b_0;
    Compute transition probabilities $\mathbb{P}(\cdot|b_0, \mu)$ and transition costs $C(b_0, \mu)$;
  Set $\alpha, i = 0$; Choose the best initial local planner $\mu^{0,0j}$ within the set $\mathbb{M}(0)$ using (10);
while $B^i_\alpha \neq B_{goal}$ **do**
  while $(\nexists B^j_\gamma, s.t.,\ b_k \in B^j_\gamma)$ *and "no collision"* **do**
    Apply the control $u_k = \mu^{\alpha,ij}_k(b_k)$ to the system;
    Get the measurement z_{k+1} from sensors;
    if *Collision happens* **then return** Collision;
    Update belief as $b_{k+1} = \tau(b_k, \mu^{\alpha,ij}_k(b_k), z_{k+1})$;
  Update the current FIRM node $B^i_\alpha = B^j_\gamma$;
  Choose the next local controller $\mu^{\alpha,ij} = \pi^g(B^i_\alpha)$;

convergence in the periodic belief process. As can be seen in Fig. 3(b), the initial covariance is three times larger than the limiting covariance, and in less than one period it converges to the SPPS solution of DPRE. The convergence time is a random quantity, whose mean and variance can be estimated through simulations. However, in practical cases it usually converges in less than one full period, because the initial covariance is closer to the actual solution (due to the use of edge-controllers) and also the orbit size is much smaller, when compared to Fig. 3(b).

Figure 4(a) shows a sample PNPRM with 23 orbits and 67 edges. To simplify the explanation of the results, we assume $m = 1$, i.e., we choose one node on each orbit. All elements in Fig. 4(a) are defined in $(\mathtt{x}, \mathtt{y}, \theta)$ space but only the $(\mathtt{x}, \mathtt{y})$ portion is shown here. To construct the FIRM nodes, we first solve the corresponding DPREs on each orbit and design its corresponding node-controller (PLQG). Then, we pick the node centers $\grave{b}^j_\alpha = (\mathbf{v}^j_\alpha, \check{P}^j_{k_\alpha})$ and construct the FIRM nodes based on the component-wise version of (17), to handle the error scale difference in position and orientation variables:

$$B^j_\alpha = \{b \equiv (x, P) |\ \ |x - \mathbf{v}^j_\alpha| \dot{<} \epsilon, |P - \check{P}^j_{k_\alpha}| \dot{<} \Delta\}, \tag{21}$$

where $|\cdot|$ and $\dot{<}$ stand for the absolute value and component-wise comparison operators, respectively. We set $\epsilon = [0.8, 0.8, 5^\circ]^T$ and $\Delta = \epsilon\epsilon^T$ to quantify B^j_α's.

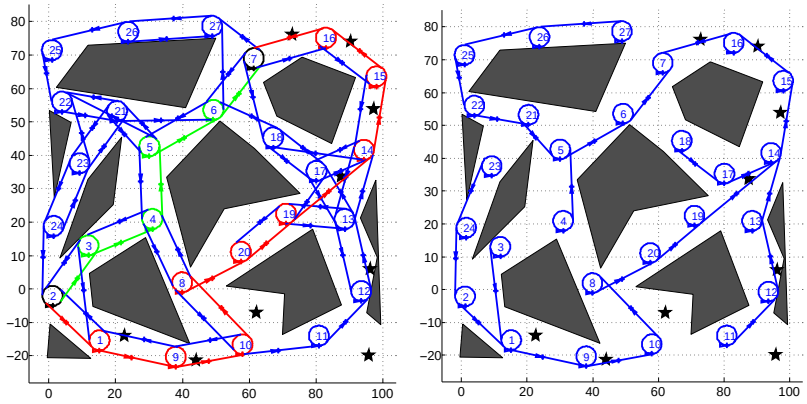
After designing FIRM nodes and local controllers, the transition costs and probabilities are computed in the offline construction phase. Here, we use sequential weighted Monte-Carlo based algorithms [15] to compute these quantities. In other words, for every $(B^i_\alpha, \mu^{\alpha,ij})$ pair, we perform M runs and accordingly approximate the transition probabilities $\mathbb{P}^g(B^j_\gamma|B^i_\alpha, \mu^{\alpha,ij})$, $\mathbb{P}^g(F|B^i_\alpha, \mu^{\alpha,ij})$, and costs $C^g(B^i_\alpha, \mu^{\alpha,ij})$. A similar approach is detailed in [5]. Table I shows these quantities for several $(B^i_\alpha, \mu^{\alpha,ij})$ pairs corresponding to Fig. 4(a), where $M = 101$ and the coefficients in (18) are $\xi_1 = 0.98$ and $\xi_2 = 0.02$.

TABLE I

COMPUTED COSTS FOR SEVERAL PAIRS OF NODE-AND-CONTROLLER USING 101 PARTICLES.

$(B^i_\alpha,\mu^{\alpha,ij})$ pair	$B^2_1,\mu^{1,(2,3)}$	$B^4_1,\mu^{1,(4,5)}$	$B^6_1,\mu^{1,(6,7)}$	$B^{11}_1,\mu^{1,(11,12)}$	$B^2_1,\mu^{1,(2,1)}$	$B^8_1,\mu^{1,(8,20)}$	$B^{16}_1,\mu^{1,(16,7)}$
$\mathbb{P}^g(F\|B^i_\alpha,\mu^{\alpha,ij})$	9.9010%	17.8218%	15.8416%	29.7030%	7.9208%	1.9802%	0.9901%
$\Phi^{\alpha,ij}$	2.1386	2.2834	1.9181	0.9152	2.1695	1.1857	0.4385
$\mathbb{E}[\mathcal{T}^{\alpha,ij}]$	63.6703	82.6747	62.5882	58.2000	51.7033	50.2755	35.4653

Plugging the computed transition costs and probabilities into (8), we can solve the DP problem and compute the policy π^g on the graph. This process is performed only once offline, independent of the starting point of the query. Fig. 4(b) shows the policy π^g on the constructed FIRM in this example. At every FIRM node B^i_α, the policy π^g decides which local controller needs to be invoked, which in turn aims to take the robot belief to the next FIRM node. It is worth noting that if we had more than one node on each orbit, the feedback π^g may return different controllers for each of them and for every orbit we may have more than one outgoing arrow in Fig. 4(b).



(a) (b)

Fig. 4. A sample PNPRM with circular orbits. Number of each orbit is written at its center. Nine landmarks (black stars) and obstacles (gray polygons) are also shown. The directions of motion on orbits and edges are shown by little triangles with a cross in their heading direction. (a) Orbits 2 and 7 (distinguished in black) are the start and goal orbits, respectively. Shortest path (green) and the most-likely path (red) under the FIRM policy are also shown. (b) Assuming on each orbit, there exists a single node, the feedback π^g is visualized for all FIRM nodes.

As discussed, the online part of planning is very efficient as it only requires executing the controller and generating the control signal. Moreover, if due to some unmodeled large disturbances, the system deviates significantly from the planned path, it suffices to bring the system back to the closest FIRM node and from thereon the optimal plan is already known, i.e., π^g drives the robot to the goal region as shown in Fig. 4(b).

We show the most likely path under the π^g in red in Fig. 4(a). The shortest path is also illustrated in Fig. 4(a) in green. It can be seen that the "most likely path under the best policy" detours from the shortest path to a path along which the filtering uncertainty is smaller, and it is easier for the controller to avoid collisions.

B. 6 DoF Aircraft Model

In this section, we consider a surveillance application for a small fixed wing aerial vehicle. Methods such as [7] have investigated stochastic optimal control of small aerial vehicles under stochastic wind. In this section, we extend such methods to belief space where the perfect state of vehicle is not available. We assume that targets to monitor are submitted from the control station frequently. Each time a new target is submitted, the aircraft has to replan in real-time and go toward the new goal, while minimizing the collision probability and the costs associated with the task objective.

System state: The system considered in this experiment is a robot with 6 Degrees of Freedom (DoF). The motion is the rigid body 6 DoF kinematics. The state of the robot x_k at time k is composed of its 3D position in Cartesian coordinates $\mathbf{p}_k$ described in the ground (inertial) frame and its orientation $\mathbf{q}_k$, which is encoded by quaternions.

$$x_k = [\mathbf{p}_k^T, \mathbf{q}_k^T]^T = [\mathtt{x_k}, \mathtt{y_k}, \mathtt{z_k}, q_{0_k}, q_{1_k}, q_{2_k}, q_{3_k}]^T, \tag{22}$$

where

$$\mathbf{p}_k = [\mathtt{x_k}, \mathtt{y_k}, \mathtt{z_k}]^T, \mathbf{q}_k = [q_{0_k}, q_{1_k}, q_{2_k}, q_{3_k}]^T, \tag{23}$$

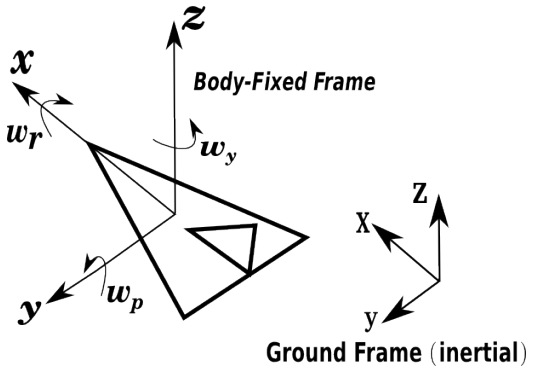


Fig. 5. The aircraft attached body-fixed frame and ground frame

Remark: Let us denote the control error (the difference between the desired state x_k^p and the mean of estimated state $\widehat{x}_k^+$) by $\widehat{e}_k^+$. In computing the control error, one can directly subtract the positional part of the state vector. However, the error in orientation (quaternions) $\mathbf{q}$ and $\mathbf{q}'$ is calculated as $\delta\mathbf{q} = \mathbf{q} \otimes \text{inv}(\mathbf{q}')$ where $\otimes$ and $\text{inv}(\cdot)$ denote the quaternion multiplication and inversion operators, respectively. We set $\delta q_0 = 0$ in calculating the control error. This is valid for small rotations since a change in the scalar part of the quaternion does not provide information about the direction of the rotation vector. Further, since we know that for a quaternion $\mathbf{q}$, $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$, by controlling q_1, q_2, q_3 we implicitly control q_0.

Motion Model: Let f be the state transition function such that,

$$x_{k+1} = f(x_k, u_k, w_k), \tag{24}$$

where, the control vector u_k is composed of the vehicle's linear velocity V_k along body x-axis and angular velocities about the body axes.

$$u_k = [V_k, \omega_k^r, \omega_k^p, \omega_k^y], \tag{25}$$

in which, ω^r, ω^p, and ω^y are the roll, pitch, and yaw rates, respectively. The motion noise is denoted by $w_k = (n_v, n_{\omega^r}, n_{\omega^p}, n_{\omega^y})^T \sim \mathcal{N}(0, \mathbf{Q}_k)$. In our simulations, $\mathbf{Q} = \text{diag}((\eta_V V + \sigma_b^V)^2, (\eta_{\omega^r}\omega^r + \sigma_b^{\omega^r})^2, (\eta_{\omega^p}\omega^p + \sigma_b^{\omega^p})^2), (\eta_{\omega^y}\omega^y + \sigma_b^{\omega^y})^2)$, where the parameters are $\eta_V = \eta_{\omega^r} = \eta_{\omega^p} = \eta_{\omega^y} = 0.005$, $\sigma_b^V = 0.02$ meters, $\sigma_b^{\omega^r} = \sigma_b^{\omega^p} = \sigma_b^{\omega^y} = 0.25$ degrees. To describe the kinematics model, we split the motion model into two parts: position and orientation (attitude).

To derive a model that governs the position of the robot (i.e., $\mathbf{p}_{k+1} = f_p(\mathbf{p}_k, \mathbf{q}_k, u_k, w_k)$), we first need to transform velocity V_k from body to the ground frame. We denote the velocity in the body-fixed frame as ${}^bV$ and in the inertial (ground) frame as ${}^gV$. Thus,

$${}^gV = R_{gb}\,{}^bV, \tag{26}$$

where, ${}^bV = [V, 0, 0]^T$ and R_{gb} is the rotation matrix that transforms the body frame to the ground frame. In terms of the quaternions, the R_{gb} matrix is as follows:

$$R_{gb}(\mathbf{q}) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}. \tag{27}$$

Similarly, we transform the motion noise in velocity to the ground frame,

$${}^gn_V = R_{gb}\,{}^bn_V, \tag{28}$$

where ${}^bn_V = [n_V, 0, 0]^T$. Therefore, f_p can be described as:

$$\mathbf{p}_{k+1} = f_p(\mathbf{p}_k, \mathbf{q}_k, u_k, w_k) = \mathbf{p}_k + {}^gV\delta t + {}^gn_V\sqrt{\delta t} = \mathbf{p}_k + R_{gb}(\mathbf{q})({}^bV\delta t + {}^bn_V\sqrt{\delta t}). \tag{29}$$

Now, we discuss the model we utilize to govern the orientation of the robot (i.e., $\mathbf{q}_{k+1} = f_q(\mathbf{q}_k, u_k, w_k)$). We start by the quaternion-based attitude kinematics in its continuous-time form that can be written as $\dot{\mathbf{q}} = A\omega$, where $\omega = [\omega^r, \omega^p, \omega^y]^T$ is the angular velocity vector of the robot with respect to the inertial frame expressed in the body frame, and A is given by:

$$A = \frac{1}{2}\begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix}. \tag{30}$$

Therefore, the discrete version of the quaternion evolution (before sign check) is as follows:

$$\mathbf{q}_{k+1}^s = f_q^s(\mathbf{q}_k, u_k, w_k) = \mathbf{q}_k + \dot{\mathbf{q}}\delta t + n_q, \tag{31}$$

where,

$$n_q = A(n_{\omega^r}, n_{\omega^p}, n_{\omega^y})^T\sqrt{\delta t}. \tag{32}$$

However, to avoid discontinuity in the control error $\widehat{e}_k^+$, we keep the scalar part of quaternion positive; i.e. the quaternion at the $(k+1)$-th time step is:

$$\mathbf{q}_{k+1} = \bar{f}(\mathbf{q}_{k+1}^s) = \mathbf{q}_{k+1}^s \text{sign}(\mathbf{q}_{0_{k+1}}^s), \tag{33}$$

where $\text{sign}(\mathbf{q}_{0_{k+1}}^s)$ is 1 if $\mathbf{q}_{0_{k+1}}^s \geq 0$, and is -1 otherwise. This procedure leads to the smaller angle since $q_0 = \cos(\phi/2)$ where ϕ is the magnitude of rotation, and thus, the smaller angular difference (i.e., $|\phi| < \pi$) always leads to a positive q_0. Note that we are allowed to do this because quaternions are invariant to sign; i.e., $\mathbf{q}_{k+1}$ and $-\mathbf{q}_{k+1}$ represent the same orientation. Thus overall we get $\mathbf{q}_{k+1} = f_q(\mathbf{q}_k, u_k, w_k) = \bar{f}(f_q^s(\mathbf{q}_k, u_k, w_k))$.

Finally, since the quaternions norm is constrained (i.e., $\|\mathbf{q}\| = 1$), if the result of an approximate calculation such as *linearized* Kalman filter is a quaternion q that does not satisfy this constraint, we apply the transformation $\mathbf{q}' = \|\mathbf{q}\|^{-1}\mathbf{q} = g(\mathbf{q})$. Note that function g is applied on the mean value and its first order approximation is applied on the covariance of the quaternion estimation.

Observation Model: The 3D location of the i-th Landmark is defined as $L^i = [L^i_x, L^i_y, L^i_z]$. We denote the relative vector from robot to landmark L^i by ${}^i\mathbf{d}^g = [{}^id^g_x, {}^id^g_y, {}^id^g_z]^T := L^i - \mathbf{p}$, where $\mathbf{p} = [\mathsf{x}, \mathsf{y}, \mathsf{z}]^T$ is the position of the robot in the ground frame. The relative vector ${}^i\mathbf{d}^g$ needs to be rotated from the ground frame to the body frame by the rotation matrix $R_{bg} = R^T_{gb}$. Thus, ${}^i\mathbf{d}^b = R_{bg}{}^i\mathbf{d}^g$.

The measurement L^i can be modeled as follows:

$$ {}^iz = h(x, {}^iv) = [{}^ir, {}^i\alpha, {}^i\beta]^T = [\|{}^i\mathbf{d}^b\|, \text{atan2}({}^id^b_y, {}^id^b_x), \text{atan2}({}^id^b_z, {}^id^b_x)]^T + {}^iv, \qquad {}^iv \sim \mathcal{N}(0, \mathbf{R}^i) \tag{34} $$

where $\mathbf{R}^i = \text{diag}((\eta_r\|{}^i\mathbf{d}\| + \sigma^r_b)^2, (\eta_\alpha\|{}^i\mathbf{d}\| + \sigma^\alpha_b)^2, (\eta_\beta\|{}^i\mathbf{d}\| + \sigma^\beta_b)^2)$. The parameters are $\eta_r = 0.01$, $\eta_\alpha = \eta_\beta = 0.3$ and $\sigma^r_b = 0.01$ meter and $\sigma^\alpha_b = \sigma^\beta_b = 0.5$ degrees are the bias standard deviations.

PNPRM generation: To generate the underlying PNPRM, we need to sample orbits and connect them to each other. In this experiment, we consider circular (counter-clockwise) orbits that are parallel to the ground. To sample an orbit, we sample a random point p^c in 3D space as the orbit center, and generate a circular trajectory with a given maximum yaw rate centered at p^c. More details on this construction can be found in [1]. Finally, we choose three nodes on each orbit uniformly distributed along the orbit.

The edge connecting node $\mathbf{v}^i_\alpha$ to orbit O^j is composed of two segments: pre-edge $\mathbf{e}^{i_\alpha j}$ and orbit-edge $\mathbf{e}^{ij}$. The edge $\mathbf{e}^{ij}$, connects the leaving point on orbit O^i to the entry point on orbit O^j. To construct $\mathbf{e}^{ij}$, we use the RRT (Rapidly exploring Random Tree) approach [22]. However, we inject user information and guide the sampling procedure in RRT to obtain better and faster results. The details of this implementation can be found in [1]. It is worth noting that in our PNPRM construction for both 2D and 3D systems, we assume that orbits are counter-clockwise in direction. An alternate approach with both clockwise and counter-clockwise orbits could also be adopted since our method is not restrictive in that sense. In this simulations, we limit ourselves to a single orbit direction for reasons of simplicity and clarity.

Planning for 6D aircraft with FIRM: After generating a PNPRM, we leverage the orbits and edges to belief space as discussed in Section V. Accordingly, we compute the edge costs and solve the DP on the FIRM graph to get a feedback from graph nodes to graph edges. Fig. 6 depicts a 3-D environment with the constructed PNPRM. The robot is given a task to visit nodes 2, 3 and 7 in that order starting from node 1. These nodes represent locations where the robot is to perform intelligence gathering. Fig. 7 shows the feedback π^g on the FIRM graph; i.e., it shows the best edge that π^g selects at each node. Shortest path is shown in green whereas the most likely path under the policy is depicted in red. It can be seen that the path selected through FIRM takes routes which are more informative and thus have less filtering uncertainty. It is worth noting that the green edges are not a part of feedback; they are just drawn to illustrate the shortest path. Fig. 8 shows the feedback to go to node 3, resulting from online replanning after the query to node 3 is submitted. Finally, Fig. 9 shows the feedback to node 7 after the next online replanning. To perform replanning (recomputing the feedback), we do not need to re-construct the graph or recompute the edge cost. Multiple queries can be executed by simply re-solving the DP on the FIRM graph with a new goal.

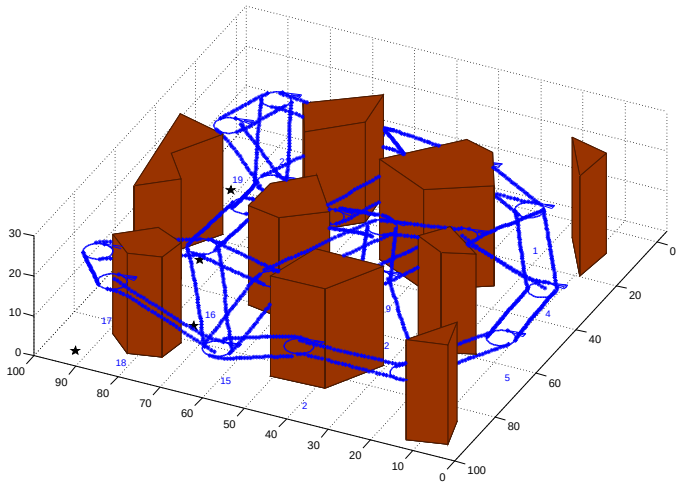
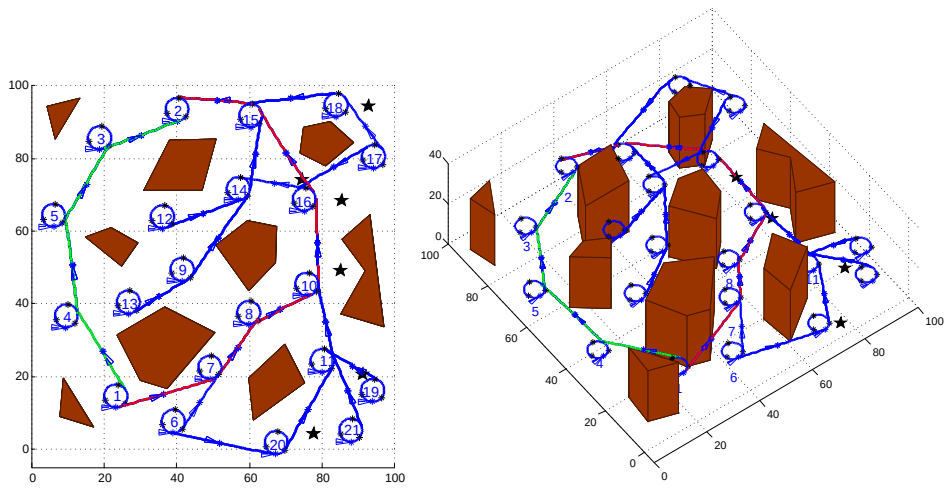
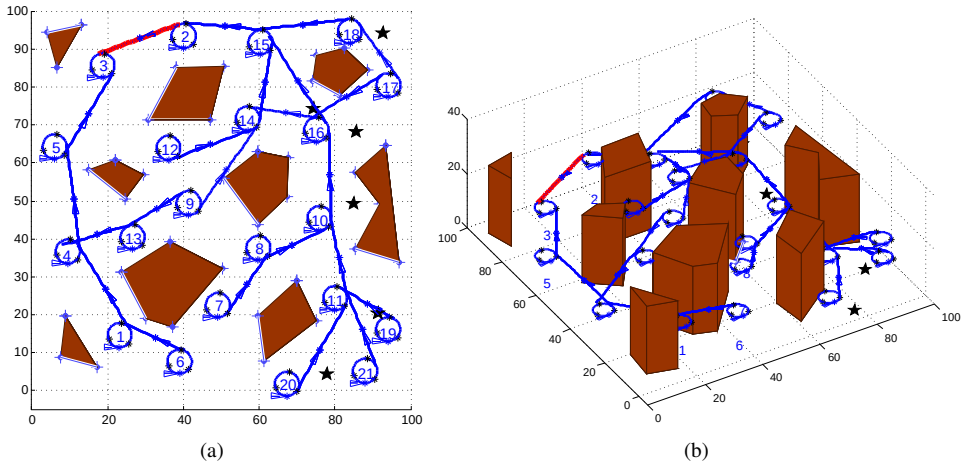


Fig. 6. The PNPRM in 3D showing the orbits and edges.



(a) (b)

Fig. 7. Feedback π^g is shown with orbit 2 as the goal orbit. (a) Starting from orbit 1, the shortest path (green) and the most-likely path (red) are shown from the top view (b) The shortest path (green) and the most-likely path (red) are shown in the 3D environment.

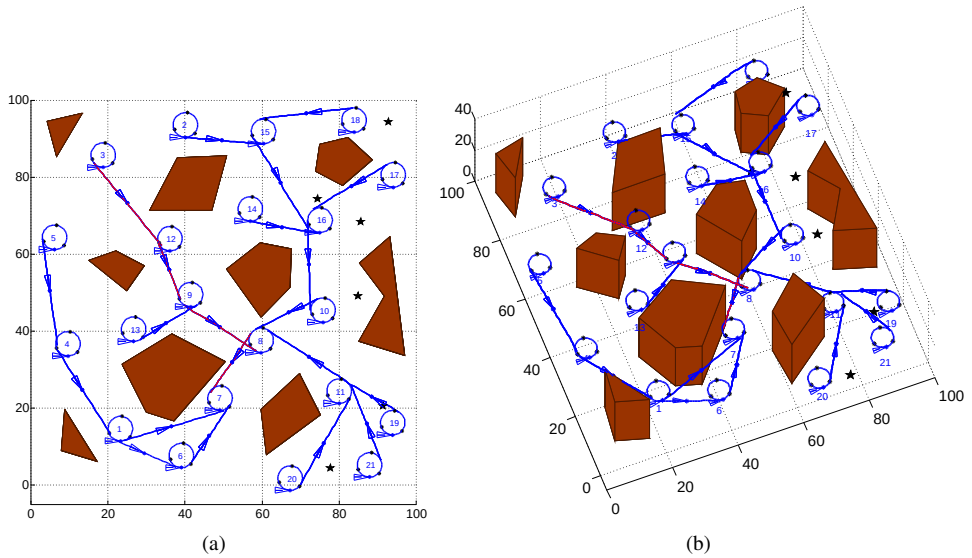


(a) (b)

Fig. 8. Feedback π^g is shown with orbit 3 as the goal orbit. Starting from orbit 2, the most-likely path (red) is shown (a) from the top view and (b) in the 3D environment.

VII. Conclusion

This paper proposes a solution to the problem of stochastic planning for non-stoppable (and possibly nonholonomic) systems, such as small fixed-wing aerial vehicles. The Periodic-Node PRM (PNPRM) is introduced as a graph in the state space, whose nodes lie on periodic trajectories, called orbits. Exploiting the properties of periodic LQG controllers on the orbits, we designed appropriate local controllers to accomplish the task of belief reachability for non-stoppable systems. Accordingly, by suitably choosing belief nodes along the orbits we constructed a graph in belief space. Planning constraints can be seamlessly embedded along the edges of this graph. Finally, the framework characterizes the success probability of reaching the goal point from any given graph node. With estimation uncertainty chosen as the planning cost, simulation



(a) (b)

Fig. 9. Feedback π^g is shown with orbit 7 as the goal orbit. Starting from orbit 3, the most-likely path (red) is shown (a) from the top view and (b) in the 3D environment.

results for two different types of robots were presented. It was demonstrated that the proposed graph-based scheme for planning under uncertainty tends to find feedback laws that guide the robot toward goal through information-rich regions (leading to less estimation uncertainty) and regions with less collision probability.

REFERENCES

[1] Aliakbar Agha-mohammadi, Saurav Agarwal, and Suman Chakravorty. Periodic-node graph-based framework for stochastic control of small aerial vehicles. Technical report, Department of Aerospace Engineering, Texas A&M University, 2014.

[2] Aliakbar Agha-mohammadi, Saurav Agarwal, Aditya Mahadevan, Suman Chakravorty, Daniel Tomkins, Jory Denny, and Nancy Amato. Robust online belief space planning in changing environments: Application to physical mobile robots. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, Hong Kong, China, 2014.

[3] Aliakbar Agha-mohammadi, Suman Chakravorty, and Nancy Amato. FIRM: Feedback controller-based Information-state RoadMap -a framework for motion planning under uncertainty-. In *International Conference on Intelligent Robots and Systems (IROS)*, SanFrancisco, 2011.

[4] Aliakbar Agha-mohammadi, Suman Chakravorty, and Nancy Amato. On the probabilistic completeness of the sampling-based feedback motion planners in belief space. In *IEEE International Conference on Robotics and Automation (ICRA)*, Minneapolis, 2012.

[5] Aliakbar Agha-mohammadi, Suman Chakravorty, and Nancy Amato. FIRM: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements. *International Journal of Robotics Research (IJRR)*, 33(2):268–304, 2014.

[6] N. Amato, B. Bayazit, L. Dale, C. Jones, and D. Vallejo. OBPRM: An Obstacle-Basaed PRM for 3D workspaces. In *International Workshop on the Algorithmic Foundations of Robotics*, pages 155–168, 1998.

[7] Ross P. Anderson, Efstathios Bakolas, Dejan Milutinović, , and Panagiotis Tsiotras. Optimal feedback guidance of a small aerial vehicle in a stochastic wind. *Journal of Guidance, Control, and Dynamics*, 36:975–985, 2013.

[8] K Astrom. Optimal control of Markov decision processes with incomplete state estimation. *Journal of Mathematical Analysis and Applications*, 10:174–205, 1965.

[9] Dimitri Bertsekas. *Dynamic Programming and Optimal Control: 3rd Ed.* Athena Scientific, 2007.

[10] S. Bittanti, P. Bolzern, G. De Nicolao, and L. Piroddi. Representation, prediction and identification of cyclostationary processes. Cyclostationarity in Communications and Signal Processing*, Edited by W.A. Gardner, IEEE Press*, 1994.

[11] S. Bittanti and P. Colaneri. *Periodic systems filtering and control*. Springer-Verlag, 2009.

[12] Roger W. Brockett. Asymptotic stability and feedback stabilization. In R. S. Millman R. W. Brockett and H. J. Sussmann, editors, *Differential Geometric Control Theory*, pages 181–191. Birkhauser, Boston, 1983.

[13] P. Colaneri, R. Scattolini, and N. Schiavoni. LQG optimal control for multirate samled-data systems. *IEEE Transactions on Automatic Control*, 37(5):675–682, 1992.

[14] John Crassidis and John Junkins. *Optimal Estimation of Dynamic Systems*. Chapman & Hall/CRC, 2004.

[15] A. Doucet, J.F.G. de Freitas, and N.J. Gordon. *Sequential Monte Carlo methods in practice*. New York: Springer, 2001.

[16] D. Hsu. *Randomized single-query motion planning in expansive spaces*. PhD thesis, Department of Computer Science, Stanford University, Stanford, CA, 2000.

[17] V. Huynh and N. Roy. icLQG: combining local and global optimization for control in information space. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

[18] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

[19] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, June 2011.

[20] L.E. Kavraki, P. Švestka, J.C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[21] P. R. Kumar and P. P. Varaiya. *Stochastic Systems: Estimation, Identification, and Adaptive Control*. Prentice-Hall, Englewood Cliffs, NJ, 1986.

[22] Steve Lavalle and J. Kuffner. Randomized kinodynamic planning. *International Journal of Robotics Research*, 20(378-400), 2001.

[23] Sean Meyn and Richard L. Tweedie. *Markov Chains and Stochastic Stability: 2nd Edition*. Cambridge University Press, 2009.

[24] Sam Prentice and Nicholas Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *International Journal of Robotics Research*, 28(11-12), October 2009.
[25] Dan Simon. *Optimal State Estimation: Kalman, H-infinity, and Nonlinear Approaches*. John Wiley and Sons, Inc, 2006.
[26] Richard Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112:181–211, 1999.
[27] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, 2005.
[28] Jur van den Berg, Pieter Abbeel, and Ken Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *IJRR*, 30(7):895–913, 2011.

APPENDIX I
PERIODIC LQG CONTROLLER

Periodic Linear Quadratic Gaussian (PLQG) controller is a time-varying LQG controller that is designed to track a periodic nominal trajectory in the presence of process and observation noise.

In this section, we first discuss the system linearization and nominal trajectory, and then discuss the KF, LQR and LQG designed along this trajectory. Consider the nonlinear partially-observable state-space equations of the system as follows:

$$x_{k+1} = f(x_k, u_k, w_k), \quad w_k \sim \mathcal{N}(0, Q_k) \tag{35a}$$
$$z_k = h(x_k, v_k), \quad v_k \sim \mathcal{N}(0, R_k). \tag{35b}$$

A T-periodic nominal trajectory for the robot is a sequence of planned states $(x_k^p)_{k\geq 0}$ and planned controls $(u_k^p)_{k\geq 0}$, such that it is consistent with the noiseless dynamics model, i.e., we have:

$$x_{k+1}^p = f(x_k^p, u_k^p, 0), \quad x_{k+T}^p = x_k^p, \quad u_{k+T}^p = u_k^p. \tag{36}$$

The role of a closed-loop stochastic controller in tracking a nominal trajectory is to compensate for robot's deviations (due to the noise) from the nominal trajectory and to keep the robot close to the nominal trajectory in the sense of minimizing the following quadratic cost:

$$J = \mathbb{E}\left[\sum_{k\geq 0}(x_k - x_k^p)^T W_x (x_k - x_k^p) + (u_k - u_k^p)^T W_u (u_k - u_k^p)\right] \tag{37}$$

where W_x and W_u are positive definite weight matrices for the state and control cost, respectively.

Since the system's state is only partially observable, at every step of LQG execution, a Kalman filter estimates the system's state and an LQR controller generates the optimal control based on this estimation. We first linearize the system along the nominal trajectory and then describe the KF and LQR designed along this path.

Model linearization: Given a periodic nominal trajectory $(x_k^p, u_k^p)_{k\geq 0}$, we linearize the dynamics and observation model in (35), as follows:

$$x_{k+1} = f(x_k^p, u_k^p, 0) + A_k(x_k - x_k^p) + B_k(u_k - u_k^p) + G_k w_k, \quad w_k \sim \mathcal{N}(0, Q_k) \tag{38a}$$
$$z_k = h(x_k^p, 0) + H_k(x_k - x_k^p) + M_k v_k, \quad v_k \sim \mathcal{N}(0, R_k) \tag{38b}$$

where

$$\begin{cases} A_k &= \frac{\partial f}{\partial x}(x_k^p, u_k^p, 0), \quad B_k = \frac{\partial f}{\partial u}(x_k^p, u_k^p, 0), \quad G_k = \frac{\partial f}{\partial w}(x_k^p, u_k^p, 0), \\ H_k &= \frac{\partial h}{\partial x}(x_k^p, 0), \quad M_k = \frac{\partial h}{\partial v}(x_k^p, 0) \end{cases} \tag{39}$$

It is worth noting that the linearized system is a periodic one, i.e.,

$$A_{k+T} = A_k, \quad B_{k+T} = B_k, \quad G_{k+T} = G_k, \quad H_{k+T} = H_k, \quad M_{k+T} = M_k, \quad Q_{k+T} = Q_k, \quad R_{k+T} = R_k. \tag{40}$$

Error system: Now, let us define the following errors:

- LQG error (main error): $e_k = x_k - x_k^p$
- KF error (estimation error): $\widetilde{e}_k = x_k - \widehat{x}_k^+$
- LQR error (mean of estimation of LQG error): $\widehat{e}_k^+ = \widehat{x}_k^+ - x_k^p$

Note that these errors are linearly dependent: $e_k = \widehat{e}_k^+ + \widetilde{e}_k$. Also, defining $\delta u_k = u_k - u_k^p$ and $\delta z_k = z_k - z_k^p := z_k - h(x_k^p, 0)$, we can rewrite above linearized models as follows:

$$e_{k+1} = A_k e_k + B_k \delta u_k + G_k w_k, \quad w_k \sim \mathcal{N}(0, Q_k) \tag{41a}$$
$$\delta z_k = H_k e_k + M_k v_k, \quad v_k \sim \mathcal{N}(0, R_k) \tag{41b}$$

which is a periodic linear system due to (40).

Periodic Kalman filter: Periodic Kalman Filter (PKF) is a time-varying Kalman filter, whose underlying linear system is periodic. In Kalman filtering, we aim to provide an estimate of the system's state based on the observations we have obtained and the control signals we have applied up to time k, i.e., $z_{0:k}$ and $u_{0,k-1}$. The estimated state is a random vector denoted by x_k^+, whose distribution is the conditional distribution of the state on the obtained data so far, which is referred to as belief and is denoted by b_k:

$$b_k = p(x_k^+) = p(x_k | z_{0:k}, u_{0:k-1}) \tag{42}$$
$$\widehat{x}_k^+ = \mathbb{E}[x_k | z_{0:k}, u_{0:k-1}] \tag{43}$$
$$P_k = \mathbb{C}[x_k | z_{0:k}, u_{0:k-1}] \tag{44}$$

where $\mathbb{E}[\cdot|\cdot]$ and $\mathbb{C}[\cdot|\cdot]$ are the conditional expectation and conditional covariance operators, respectively. In the Gaussian case, we have $b_k = \mathcal{N}(\widehat{x}_k^+, P_k)$, i.e., the belief can only be characterized by its mean and covariance. Hence, we can show b_k as the mean-covariance pair $b_k \equiv (\widehat{x}_k^+, P_k)$. Similar to the conventional Kalman filtering, PKF consists of two steps at every time stage: the prediction step and the update step. In the prediction step, the mean and covariance of prior x_k^- is computed. For the system in (41) the prediction step is:

$$\widehat{e}_{k+1}^- = A_k \widehat{e}_k^+ + B_k \delta u_k \tag{45}$$

$$P_{k+1}^- = A_k P_k^+ A_k^T + G_k Q_k G_k^T \tag{46}$$

In the update step, the mean and covariance of posterior x_k^+ is computed. For the system in (41), the update step is:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + M_k R_k M_k^T)^{-1} \tag{47}$$

$$\widehat{e}_{k+1}^+ = \widehat{e}_{k+1}^- + K_{k+1}(\delta z_{k+1} - H_{k+1}\widehat{e}_{k+1}^-) \tag{48}$$

$$P_{k+1}^+ = (I - K_{k+1}H_{k+1})P_{k+1}^- \tag{49}$$

Note that

$$\widehat{x}_k^+ = \mathbb{E}[x_k | z_{0:k}, u_{0:k-1}] = x_k^p + \mathbb{E}[e_k | z_{0:k}, u_{0:k-1}] = x_k^p + \widehat{e}_k^+ \tag{50}$$

$$P_k = \mathbb{C}[x_k | z_{0:k}, u_{0:k-1}] = \mathbb{C}[e_k | z_{0:k}, u_{0:k-1}] = P_k^+ \tag{51}$$

Lemma 4: **(Covariance convergence under PLQG):** In Periodic Kalman filtering, if for all k, the pair (A_k, H_k) is detectable and the pair $(A_k, \check{Q}_k)$ is stabilizable, where $G_k Q_k G_k^T = \check{Q}_k \check{Q}_k^T$, then the prior covariance P_k^-, the posterior covariance P_k, and the filter gain K_k all converge to their T-periodic stationary values, denoted by $\check{P}_t^-$, $\check{P}_t$, and $\check{K}_t$, respectively [11]. Matrix $\check{P}_t^-$ is the unique Symmetric T-Periodic Positive Semi-definite (SPPS) solution [11] of the following Discrete Periodic Riccati Equation (DPRE):

$$\check{P}_{k+1}^- = G_k Q_k G_k^T + A_k(\check{P}_k^- - \check{P}_k^- H_k^T (H_k \check{P}_k^- H_k^T + M_k R_k M_k^T)^{-1} H_k \check{P}_k^-) A_k^T \tag{52}$$

Having $\check{P}_k^-$, the periodic gain $\check{K}_k$ and estimation covariance $\check{P}_k$ are computed as follows:

$$\check{K}_k = \check{P}_k^- H_k^T (H_k \check{P}_k^- H_k^T + M_k R_k M_k^T)^{-1}, \tag{53}$$

$$\check{P}_k = (I - \check{K}_k H_k)\check{P}_k^- \tag{54}$$

where

$$\check{P}_{k+T}^- = \check{P}_k^-, \quad \check{K}_{k+T} = \check{K}_k, \quad \check{P}_{k+T} = \check{P}_k \tag{55}$$

Proof: See [11]. ∎

Note that if the pair (A_k, H_k) is detectable and the pair $(A_k, \check{Q}_k)$ is stabilizable, then the pair (A_k, H_k) is observable and the pair $(A_k, \check{Q}_k)$ is controllable, and hence Lemma 2 follows.

Periodic LQR controller: An LQR controller is utilized as the separated controller [21] within the structure of the LQG controller. Once Kalman filter produces the estimation (belief), the LQR controller generates the optimal control signal accordingly. In other words, we have a time-varying mapping μ_k from belief space into the control space that generates an optimal control based on the given belief $u_k = \mu_k(b_k)$ at every time step k. In LQG, the mapping μ_k is the control law of the LQR controller, which is optimal in the sense of minimizing the following cost:

$$\begin{aligned} J_{PLQR} &= \mathbb{E}\left[\sum_{k\geq 0} (\widehat{x}_k^+ - x_k^p)^T W_x (\widehat{x}_k^+ - x_k^p) + (u_k - u_k^p)^T W_u (u_k - u_k^p)\right] \\ &= \mathbb{E}\left[\sum_{k\geq 0} (\widehat{e}_k^+)^T W_x (\widehat{e}_k^+) + (\delta u_k)^T W_u (\delta u_k)\right]. \end{aligned} \tag{56}$$

The linear control law that minimizes this cost function for a linear system is:

$$\delta u_k = -L_k \widehat{e}_k^+, \quad L_{k+T} = L_k \tag{57}$$

Lemma 5: In Periodic LQR (PLQR), if for all k, the pair (A_k, B_k) is stabilizable and the pair $(A_k, \check{W}_x)$ is detectable, where $W_x = \check{W}_x^T \check{W}_x$, then the time-varying feedback gains L_k are T-periodic gains, i.e., $L_{k+T} = L_k$ and are computed as follows:

$$L_k = (B_k^T S_{k+1} B_k + W_u)^{-1} B_k^T S_{k+1} A_k, \tag{58}$$

where S_k is the SPPS solution of the following DPRE:

$$S_k = W_x + A_k^T S_{k+1} A_k - A_k^T S_{k+1} B_k (B_k^T S_{k+1} B_k + W_u)^{-1} B_k^T S_{k+1} A_k. \tag{59}$$

Note that the whole control is $u_k = u_k^p + \delta u_k$.

Periodic LQG controller: Plugging the obtained control law of PLQR into the PKF equations, we can get the following error dynamics:

$$\begin{aligned}\begin{pmatrix} e_{k+1} \\ \widetilde{e}_{k+1} \end{pmatrix} &= \begin{pmatrix} A_k - B_k L_k & B_k L_k \\ 0 & A_k - \check{K}_{k+1} H_{k+1} A_k \end{pmatrix} \begin{pmatrix} e_k \\ \widetilde{e}_k \end{pmatrix} \\ &+ \begin{pmatrix} G_k & 0 \\ G_k - \check{K}_{k+1} H_{k+1} G_k & -\check{K}_{k+1} M_{k+1} \end{pmatrix} \begin{pmatrix} w_k \\ v_{k+1} \end{pmatrix},\end{aligned} \tag{60}$$

or equivalently,

$$\begin{aligned}\begin{pmatrix} e_{k+1} \\ \widehat{e}_{k+1}^+ \end{pmatrix} &= \begin{pmatrix} A_k & -B_k L_k \\ \check{K}_{k+1} H_{k+1} A_k & A_k - B_k L_k - \check{K}_{k+1} H_{k+1} A_k \end{pmatrix} \begin{pmatrix} e_k \\ \widehat{e}_k^+ \end{pmatrix} \\ &+ \begin{pmatrix} G_k & 0 \\ \check{K}_{k+1} H_{k+1} G_k & \check{K}_{k+1} M_{k+1} \end{pmatrix} \begin{pmatrix} w_k \\ v_{k+1} \end{pmatrix}.\end{aligned} \tag{61}$$

Defining $\zeta_k := (e_k, \widehat{e}_k^+)^T$ and $q_k := (w_k, v_{k+1})^T$, we can rewrite (61) in a more compact form as

$$\zeta_{k+1} = \overline{F}_k \zeta_k - \overline{G}_k q_k, \quad q_k \sim \mathcal{N}(0, \overline{Q}_k), \quad \overline{Q}_k = \begin{pmatrix} Q_k & 0 \\ 0 & R_{k+1} \end{pmatrix} \tag{62}$$

with appropriate definitions for $\overline{F}_k$ and $\overline{G}_k$. Thus, ζ_k is a random variable with a Gaussian distribution, i.e.,

$$\zeta_k \sim \mathcal{N}(0, \mathcal{P}_k), \tag{63}$$

or

$$\begin{pmatrix} x_k \\ \widehat{x}_k^+ \end{pmatrix} \sim \mathcal{N}(\begin{pmatrix} x_k^p \\ x_k^p \end{pmatrix}, \mathcal{P}_k), \tag{64}$$

where $\mathcal{P}_k$ is the solution of the following Discrete Periodic Lyapunov Equation (DPLE):

$$\mathcal{P}_{k+1} = \overline{F}_k \mathcal{P}_k \overline{F}_k^T - \overline{G}_k \overline{Q}_k \overline{G}_k^T, \tag{65}$$

which can be decomposed into four blocks

$$\mathcal{P}_k = \begin{pmatrix} \mathcal{P}_{k,11} & \mathcal{P}_{k,12} \\ \mathcal{P}_{k,21} & \mathcal{P}_{k,22} \end{pmatrix}. \tag{66}$$

Lemma 6: Under the preceding assumptions in Lemmas 4 and 5, the solution of DPLE in (65) converges to a unique SPPS solution $\check{\mathcal{P}}_k$ independent of the initial covariance $\mathcal{P}_0$, i.e., $\check{\mathcal{P}}_{k+T} = \check{\mathcal{P}}_k$.

Proof: See [11]. ■

Therefore, the process in (62) converges to a cyclostationary process [10], i.e., the distribution over ζ_k is periodic. Thus, since $\widehat{x}_k^+ \sim \mathcal{N}(x_k^p, \mathcal{P}_{k,22})$, the distribution over the estimation mean is also converges to a periodic distribution, i.e., $\widehat{x}_k^+ \sim \mathcal{N}(x_k^p, \check{\mathcal{P}}_{k,22}) = \mathcal{N}(x_{k+T}^p, \check{\mathcal{P}}_{k+T,22})$. Hence, this analysis leads to the following lemma:

Lemma 7: Under Periodic LQG, belief falls into a Gaussian cyclostationary process, i.e., the distribution over belief $b_k \equiv (\widehat{x}_k^+, P_k)$ converges to the following periodic Gaussian distribution:

$$b_k \equiv (\widehat{x}_k^+, P_k) \sim \mathcal{N}\left(\begin{pmatrix} x_k^p \\ \check{P}_k \end{pmatrix}, \begin{pmatrix} \check{\mathcal{P}}_{k,22} & 0 \\ 0 & 0 \end{pmatrix}\right) \tag{67}$$

The degeneracy of the Gaussian distribution over belief in (67) is due to the fact that $\check{P}_k$ is a deterministic process. It is worth noting that the belief mean converges to the T-periodic belief $\mathbb{E}[b_{k+T}] = \mathbb{E}[b_k] = (x_k^p, \check{P}_k)$. Hence, the Lemma 1 follows, as it is the same as Lemma 7, where we have:

$$b_k^c = \begin{pmatrix} x_k^p \\ \check{P}_k \end{pmatrix}, \quad \mathtt{C}_k = \begin{pmatrix} \check{\mathcal{P}}_{k,22} & 0 \\ 0 & 0 \end{pmatrix} \tag{68}$$

APPENDIX II
PROOF OF LEMMA 3

Proof: Let us consider the state space model of a T-periodic linear system of interest as follows:

$$x_{k+1} = \mathbf{A}_k x_k + \mathbf{B}_k u_k + \mathbf{G}_k w_k, \quad w_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k) \tag{69a}$$

$$z_k = \mathbf{H}_k x_k + v_k, \qquad v_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k). \tag{69b}$$

Based on Lemma 1 and Lemma 2, if $(\mathbf{A}, \mathbf{B})$ and $(\mathbf{A}, \check{\mathbf{Q}})$ are controllable pairs, where $\mathbf{G}\mathbf{Q}\mathbf{G}^T = \check{\mathbf{Q}}\check{\mathbf{Q}}^T$, and if $(\mathbf{A}, \mathbf{H})$ and $(\mathbf{A}, \check{\mathbf{W}}_x)$ are observable pairs, where $\mathbf{W}_x = \check{\mathbf{W}}_x^T \check{\mathbf{W}}_x$, then the estimation covariance deterministically tends to a T-periodic stationary covariance $\check{P}_k$. Therefore, for any $\epsilon > 0$, after a deterministic finite time, P_k enters the ϵ-neighborhood of the periodic stationary covariance, i.e., $\|P_k - \check{P}_k\|_m < \epsilon$ for all k large enough, where $\|\cdot\|$ stands for an appropriate matrix norm.

The estimation mean dynamics, however, is stochastic and is as follows for the system in Eq. (69):

$$\begin{aligned}
\widehat{x}_{k+1}^+ &= x_{k+1}^p + (\mathbf{A}_k - \mathbf{B}_k\mathbf{L}_k - \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{A}_k)(\widehat{x}_k^+ - x_k^p) \\
&\quad + \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{A}_k(x_k - x_k^p) + \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{G}_k w_k + \mathbf{K}_{k+1}v_{k+1} \\
&= x_{k+1}^p - (\mathbf{A}_k - \mathbf{B}_k\mathbf{L}_k)x_k^p + (\mathbf{A}_k - \mathbf{B}_k\mathbf{L}_k - \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{A}_k)\widehat{x}_k^+ \\
&\quad + \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{A}_k x_k + \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{G}_k w_k + \mathbf{K}_{k+1}v_{k+1},
\end{aligned} \tag{70}$$

where the Kalman gain $\mathbf{K}_k$ is:

$$\mathbf{K}_k = P_k^- \mathbf{H}_k^T (\mathbf{H}_k P_k^- \mathbf{H}_k^T + \mathbf{R}_k)^{-1}. \tag{71}$$

Since $\mathbf{K}_k$ is full rank (due to the condition on the rank of $\mathbf{H}_k$) for all k and since v and w are Gaussian noises, (70) induces an irreducible Markov process over the state space [23]. Thus, if we have a stopping region for the estimation mean with size $\epsilon > 0$, the estimation mean process will hit this stopping region in finite time [23], with probability one, i.e., for a finite $\mathbf{v} \in \mathbb{X}$, the condition $\|\widehat{x}_k^+ - \mathbf{v}\| < \epsilon$ is satisfied in finite time. However, $\mathbf{v}$ can be chosen in a way that maximizes the absorption probability and minimizes the hitting time.

Based on the estimation mean dynamics in (70) and the state dynamics in Appendix I, if the estimation mean process and state process start from $\widehat{x}_0^+$ and x_0, respectively, such that $\mathbb{E}[\widehat{x}_0^+] = x_k^p$ and $\mathbb{E}[x_0] = x_k^p$ (which indeed is the case in FIRM due to the usage of edge-controllers), "the mean of estimation mean" remains on x_k^p, i.e., $\mathbb{E}[\widehat{x}_k^+] = x_k^p$, for all k. As a result, x_k^p is the optimal choice for the center of stopping region and thus, the condition $\|\widehat{x}_k^+ - x_k^p\| < \epsilon$ is satisfied in minimum time in the sense of "expected value".

Combining the results for estimation covariance and estimation mean, if we define the region $\check{B}_k$ as a set in the Gaussian belief space with a non-empty interior centered at $(x_{k_\alpha}^p, \check{P}_{k_\alpha})$, then the belief $b_k \equiv (\widehat{x}_k^+, P_k)$ enters region $\cup_k \check{B}_k$ with minimum finite expected time with probability one. To decrease the number of nodes, one can only look at the subsequence $\grave{b}_\alpha := b_{k_\alpha} \equiv (\widehat{x}_{k_\alpha}^+, P_{k_\alpha})$ and $B_\alpha := \check{B}_{k_\alpha}$ for $\{k_1, k_2, \cdots, k_m\} \subset \{1, 2, \cdots, T\}$, then similarly the belief $\grave{b}_\alpha$ enters region $\cup_\alpha B_\alpha$ in finite time with probability one. ■

Robust Online Belief Space Planning in Changing Environments: Application to Physical Mobile Robots

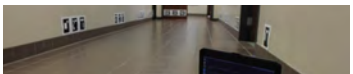
Ali-akbar Agha-mohammadi, Saurav Agarwal, Aditya Mahadevan,
Suman Chakravorty, Daniel Tomkins, Jory Denny, Nancy M. Amato

***Abstract*— Motion planning in belief space (under motion and sensing uncertainty) is a challenging problem due to the computational intractability of its exact solution. The Feedback-based Information RoadMap (FIRM) framework made an important theoretical step toward enabling roadmap-based planning in belief space and provided a computationally tractable version of belief space planning. However, there are still challenges in applying belief space planners to physical systems, such as the discrepancy between computational models and real physical models. In this paper, we propose a dynamic replanning scheme in belief space to address such challenges. Moreover, we present techniques to cope with changes in the environment (e.g., changes in the obstacle map), as well as unforeseen large deviations in the robot's location (e.g., the kidnapped robot problem). We then utilize these techniques to implement the first online replanning scheme in belief space on a physical mobile robot that is robust to changes in the environment and large disturbances. This method demonstrates that belief space planning is a practical tool for robot motion planning.**

I. Introduction

Sequential decision making under uncertainty is a key prerequisite for many robotics applications. Consider an autonomous, low-cost mobile robot that is subject to motion noise and lacks exact measurements due to sensor noise. Controlling this robot and planning motions for it is an instance of the Partially-Observable Markov Decision Process (POMDP) [13], [23] problem, which is a formal framework for sequential decision making under uncertainty. However, the POMDP problem is also notorious for its computational intractability. Methods such as [11], [15], [18], [24], [25] reduce the computation burden of POMDPs and aim to solve more challenging and realistic problems. Recently, the Feedback-based Information RoadMap (FIRM) framework [3] takes an important theoretical step toward realistic scenarios by significantly reducing the computational complexity of planning under uncertainty.

Additionally, handling changes in the environment (e.g., obstacles), changes in the goal location, and large deviations in the robot's location calls for online planning in uncertain, partially observable environments. However, when dealing with real-world physical systems, POMDP-based methods, including FIRM, encounter another important challenge: *discrepancy between real models with the models used for computation*. Such discrepancies can lead to deviations from the desired plan. Moreover, changes in the environment and large disturbances are other important challenges that needs to be handled. One strategy to address this problem is an ability to dynamically replan in belief space. In this paper, we propose a principled rollout-based extension of FIRM planning to facilitate its application to real-time stochastic (re)planning problems, and deal with changes in the environment and large disturbances in the robot's state.



(a)

(b)

Fig. 1. (a) A picture of robot (iRobot Create) in the operating environment. Landmarks can be seen on the walls. (b) Floorplan of the environment, in which experiments are conducted.

In the main body of POMDP literature, in particular sampling-based methods, the computed solution depends on the initial belief [4], [7], [9], [14], [21], [28] (sometimes referred to as single-query solvers). Therefore, in replanning (planning from a new initial belief) almost all the computations need to be done again, which prohibits their usage in cases where real-time dynamic replanning schemes, such as Receding Horizon Control (RHC), are needed. However, multi-query methods such as FIRM [2], [3] provide a construction mechanism independent of the initial belief of the system. As a result, they are suitable methods to be used for dynamic replanning purposes.

Trajectory optimization-based methods can also be used for replanning in an RHC scheme. The RHC framework was

Agha-mohammadi is with the Laboratory for Information and Decision Systems, MIT, Cambridge, MA, 02139. Agarwal and Chakravorty are with the Dept. of Aerospace Engineering, and Mahadevan, Tomkins, Denny, and Amato are with the Dept. of Computer Science and Engineering, Texas A&M University, TX, 77843, USA. Emails: aliagha@mit.edu {sauravag,mahadeven,schakrav,kittsil,jorydenny amato}@tamu.edu. This research supported in part by AFOSR Grant FA9550-08-1-0038 and by NSF awards CNS-0551685, CCF-0833199, CCF-0830753, IIS-0916053, IIS-0917266, EFRI-1240483, RI-1217991, by NIH NCI R25 CA090301-11, by Chevron, IBM, Intel, Oracle/Sun and by Award KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST). J. Denny supported in part by an NSF Graduate Research Fellowship.

originally designed for deterministic systems. The most common approach is to approximate the stochastic system with a deterministic one by replacing the uncertain quantities with their mean (or maximum likelihood) values [5]. Methods such as [8], [10], [12], [20], [27] fall into this category and can be used in the RHC setting; they replace future random observations with their deterministic maximum likelihood value. However, in this form of RHC, the optimization is carried out only within a limited horizon. Also, removing the system's stochasticity may lead to unreliable plans.

The main contributions of this paper are threefold.

- We propose a principled method for real-time replanning in belief space by extending the idea of the rollout policy [5] to belief space using FIRM. This method considers all possible future observations.
- We propose techniques such as a "lazy feedback evaluation" algorithm to react to changes in the environment as well as large disturbances.
- We implement the proposed belief space planning scheme on a physical robotic system as an application of the FIRM framework. We demonstrate the robustness of the method to changes in the environment, failures in the sensory system, and large deviations.

These results lay the groundwork for further application of the theoretical POMDP framework to practical applications, thus moving toward long-term autonomy in robotic systems.

II. Problem Statement and Target Application

We aim to design a belief space planner that can handle uncertainties associated with a typical low-cost robot. Moreover, the planner needs to be able to replan in real-time so that it can cope with changes in the environment as well as deviations resulting from model discrepancies, large disturbances, and sensor failures.

To formally define the problem, we start by defining the concept of belief and policy. Consider a system whose state, control, and motion noise are denoted by x_k, u_k, and w_k, respectively, at the k-th time step. Let us denote the state evolution model by $x_{k+1} = f(x_k, u_k, w_k)$. In a partially observable environment, the exact value of system state x_k is not known. However, we can get the measurement (or observation) vector z_k at the k-th time step through sensors. Let us denote the measurement model by $z_k = h(x_k, v_k)$, where v_k denotes sensing noise. Therefore, the only available data for decision making at the k-th time step are the observations we have received and the controls we have applied up to that time step, i.e., $\mathcal{H}_k = \{z_{0:k}, u_{0:k-1}\} = \{z_0, z_1, \cdots, z_k, u_0, \cdots, u_{k-1}\}$. A filtering module can encode this data into a probability distribution over all possible system states $b_k = p(x_k|\mathcal{H}_k)$, which is referred to as the *belief* or *information-state*. Therefore, the action u_k can be taken based on the belief b_k using a policy (planner) π_k, i.e., $u_k = \pi_k(b_k)$. In Bayesian filtering, belief can be computed recursively based on the last action and current observation, $b_{k+1} = \tau(b_k, u_k, z_{k+1})$ [5], [26].

To find the policy π_k, we need to define the objective of planning. Although the objective function can be general, the cost function we will use in our experiments includes the localization uncertainty, control effort, and elapsed time.

$$c(b_k, u_k) = \zeta_p \text{tr}(P_k) + \zeta_u \|u_k\| + \zeta_T, \tag{1}$$

where $\text{tr}(P_k)$ is the trace of estimation covariance. The norm of the control signal $\|u_k\|$ denotes the control effort, and ζ_T is present in the cost to penalize each time lapse. Coefficients ζ_p, ζ_u, and ζ_T are user-defined task-dependent scalars that combine these costs to achieve a desirable behaviour. In the presence of a constraint set F (e.g., obstacles), we assume that the task fails if the robot violates these constraints (e.g., collides with obstacles). Therefore, in case of failure, the running-sum of costs (cost-to-go), i.e., $J(F) = \sum_{t'}^{\infty} c(b, u)$ is set to a suitably high cost-to-go.

Planning under uncertainty is defined as finding a sequence of policies $\pi_{0:\infty}(\cdot) = \{\pi_1(\cdot), \pi_2(\cdot), \pi_3(\cdot), \cdots\}$. Therefore, the original problem of stochastic control with imperfect state information is defined as follows:

Problem 1. ***(POMDP)*** *The problem of stochastic control with imperfect state information, or the Partially-Observable Markov Decision Process (POMDP) problem, is defined as the following optimization over the policy space:*

$$\pi_{0:\infty}(\cdot) = \arg\min_{\Pi_{0:\infty}} \sum_{k=0}^{\infty} \mathbb{E}\left[c(b_k, \pi_k(b_k))\right] \tag{2}$$
$$s.t. \quad b_{k+1} = \tau(b_k, \pi_k(b_k), z_k), \quad z_k \sim p(z_k|x_k)$$
$$x_{k+1} = f(x_k, \pi_k(b_k), w_k), \quad w_k \sim p(w_k|x_k, \pi_k(b_k))$$

where, Π_k is the space of all possible policies at time step k, i.e., $\pi_k \in \Pi_k$.

In the infinite horizon case, the solution is a stationary policy π_s, i.e., $\pi_1 = \pi_2 = \cdots = \pi_s$. However, Problem 1 is written in a more general setting to emphasize the connection with rollout policy, discussed further below. Solving the POMDP problem is computationally intractable over continuous state, action, and observation spaces. However, the main problem that this paper aims to solve is the following:

Problem 2. ***(Re-Solving POMDPs in real-time)*** *In case of a change in the failure set F (e.g. obstacles) or large deviation in the system's belief, re-solve Problem 1 in real-time.*

This paper aims at solving Problem 2 by exploiting FIRM, which re-use the computations performed for solving the POMDP problem a priori and hence can deal with such changes online.

A. Sample Application Scenario

We exercise the proposed planner in an office-like environment, where we use a low-cost iRobot Create platform (Figure 1(a)), on which a Dell Latitude laptop with an on-board camera is mounted. The robot obtains noisy measurements (relative range and bearing) from unique landmarks that are installed in the environment. The desired behaviour for the planner is to guide the robot to a goal through those regions of the environment where the robot can better localize itself and hence better avoid collisions. Most importantly, the

planner needs to be able to replan online so that it can handle changes in the environment and deviations resulting from model discrepancies, large disturbances, and sensor failures. We briefly discuss the environment, robot model, and sensory system. More detailed descriptions can be found in [1].

Environment: The specific environment for conducting experiments is the fourth floor of the Bright building at Texas A&M University. A floorplan is shown in Fig. 1(b). The hallway (yellow) and the experiment region (blue) are highlighted. The blue region contains a large cluttered office (room 407) with several doors.

System model (robot and sensors): We use an iRobot Create (Fig. 1(a)), whose state $x_k = (\mathrm{x}_k, \mathrm{y}_k, \theta_k)^T$ encodes its 2D position and heading angle at the k-th time step. The state evolution model $x_{k+1} = f(x_k, u_k, w_k)$ is the unicycle model, where the control command u_k consists of the linear and angular velocities $u_k = (V_k, \omega_k)^T$. Motion noise $w_k \sim \mathcal{N}(0, \mathbf{Q}_k)$ gets added to the control signal (see [1] for details). For sensing purposes, we use the laptop's on-board camera to detect landmarks (with unique black and white patterns) that are placed at known locations on walls (Fig. 1(a)). Denoting the j-th landmark position as ${}^jL$, the obtained measurement is the relative range and bearing to the landmark:

$$ {}^j z_k = [\|{}^j\mathbf{d}_k\|, \text{atan2}({}^jd_{2_k}, {}^jd_{1_k}) - \theta]^T + {}^j v,\ {}^j v \sim \mathcal{N}(\mathbf{0}, {}^j\mathbf{R}), $$

where ${}^j\mathbf{d}_k = [{}^jd_{1_k}, {}^jd_{2_k}]^T := [\mathrm{x}_k, \mathrm{y}_k]^T - L_j$. Experimentally, we have found that the intensity of measurement noise ${}^j v$ increases with the distance from the j-th landmark and the incidence angle. The incidence angle refers to the angle between the line connecting the camera to a landmark and a surface normal to the wall on which the landmark is mounted. Denoting the incident angle by $\phi \in [-\pi/2, \pi/2]$, we model the sensing noise associated with the j-th landmark as a zero mean Gaussian whose covariance is

$$ {}^j\mathbf{R}_k = \mathbf{diag}\left((\eta_{r_d}\|{}^j\mathbf{d}_k\| + \eta_{r_\phi}|\phi_k| + \sigma_b^r)^2, (\eta_{\theta_d}\|{}^j\mathbf{d}_k\| + \eta_{\theta_\phi}|\phi_k| + \sigma_b^\theta)^2\right) \quad (3) $$

In our implementation, we use $\eta_{r_d} = 0.1$, $\eta_{r_\phi} = 0.01$, $\sigma_b^r = 0.05$m, $\eta_{\theta_d} = 0.001$, $\eta_{\theta_\phi} = 0.01$, and $\sigma_b^\theta = 2.0$deg. The full vector of measurements z is the concatenation of measurements from visible landmarks.

III. Overview of FIRM

In this section, we briefly review the Feedback-based Information RoadMap (FIRM) framework [2], [3]. However, the concrete realization of FIRM constructed for conducting the experiments is detailed in [1]. An Information RoadMap (IRM) is a "multi-query" graph in belief space constructed independent of the initial belief space. Therefore, the intractable belief MDP problem can be reduced to a tractable MDP problem on this graph. Each node in an IRM is a small region $B = \{b : \|b - \grave{b}\| \leq \epsilon\}$ around a sampled belief $\grave{b}$. We denote the i-th node by B^i and the set of nodes by $\mathbb{V} = \{B^i\}$. Each edge in an IRM is a local controller. In FIRM, each edge (local controller) is a feedback controller whose goal is to drive the belief into the target node of the edge. We denote the edge (controller) between nodes i and j by μ^{ij} and the set of edges by $\mathbb{M} = \{\mu^{ij}\}$. A policy π^g on the graph is a mapping from graph nodes to edges; i.e., $\pi^g : \mathbb{V} \to \mathbb{M}$. Denote the set of all possible policies as Π^g.

Having such a graph in belief space, we can form the POMDP on the FIRM graph (so-called FIRM MDP):

$$ \pi^g = \arg\min_{\Pi^g} \mathbb{E} \sum_{n=0}^{\infty} C^g(B_n, \pi^g(B_n)) \quad (4) $$

where, B_n is the n-th visited node, and μ_n is the edge taken at B_n. $C^g(B, \mu) := \sum_{k=0}^{\mathcal{T}} c(b_k, \mu(b_k))$ is the generalized cost of taking local controller μ at node B centered at b_0.

We incorporate the failure set in planning by adding a hypothetical FIRM node $B^0 = F$ to the list of FIRM nodes. As the FIRM MDP in Eq.(4) is defined over the finite set of nodes, we can solve it by computing the graph cost-to-go through solving the following dynamic programming:

$$ J^g(B^i) = \min_{\mu}\{C^g(B^i, \mu) + \sum_{\gamma=0}^{N} \mathbb{P}^g(B^\gamma | B^i, \mu) J^g(B^\gamma)\} \quad (5) $$

where $\mathbb{P}^g(B^\gamma | B^i, \mu)$ is the probability of reaching B^γ from B^i under μ. The failure and goal cost-to-go's (i.e., $J^g(B^0)$ and $J^g(B^{goal})$) are set to a suitably high positive value and zero, respectively. Accordingly, the replanning algorithms, when start or goal changes, are presented in Algorithms 1 and 2. For a more detailed description of FIRM, see [1].

Algorithm 1: (Re)plan_from

```
input : Start belief b_0, cost-to-go J^g(·), nodes V = {B^i}
output : Next Local Controller μ*
Find r neighboring nodes N = {B^i}_{i=1}^r to b_0;
Set J*(B) = ∞;
for B ∈ N do
    Construct local planner μ from b_0 to B;
    Compute transition cost C(b_0, μ) and probability P(B|b_0, μ);
    if C(b_0, μ) + Σ_{γ=0}^{N} P(B^γ|b_0, μ) J^g(B^γ) < J*(B) then
        J*(B) = C(b_0, μ) + Σ_{γ=0}^{N} P(B^γ|b_0, μ) J^g(B^γ);
        μ* = μ;
return μ*;
```

Algorithm 2: (Re)plan_to

```
input : Goal node B^goal, FIRM Graph G = {V, M}
output : FIRM feedback π^g
Add B^goal to the graph; update V and M accordingly;
Compute the cost-to-go J^g and feedback π^g over the FIRM nodes by solving the MDP in Eq. (5);
return π^g;
```

IV. Dynamic Replanning in Belief Space

In this section, we first discuss the extension of the Receding Horizon Control (RHC) and Rollout Policy (ROP) [5] to belief space. Then we propose an ROP based on FIRM

that can cope with changes in the environment as well as large deviations.

RHC in belief space: Receding horizon control (often referred to as rolling-horizon or model-predictive control) was originally designed for deterministic systems (to cope with model discrepancy). For stochastic systems, where the closed-loop (feedback) control law is needed, the best formulation of the RHC scheme is a subject of current research [8], [16], [22]. In the most common form of RHC [5], the stochastic system is approximated with a deterministic system by replacing the uncertain quantities with their typical values (e.g., maximum likelihood value). In belief space planning, the quantities that inject randomness into belief dynamics are unknown future observations. Thus, one can replace random observations z_k with their deterministic maximum likelihood value z_k^{ml}, where $z_k^{ml} := \arg\max_z p(z_k|x_k^d)$ in which x^d is the nominal deterministic value for the state that results from replacing the motion noise w by zero; i.e., $x_{k+1}^d = f(x_k^d, \pi_k(b_k^d), 0)$. The deterministic belief b^d is then used for planning in the receding horizon window. At every time step, the RHC scheme performs a two-stage computation. To describe these stages, let us assume we are at step n and the belief is b_n. At the first stage, the RHC scheme for deterministic systems solves an open-loop control problem (i.e., returns a sequence of actions $u_{0:T}$) over a fixed finite horizon T by solving the following optimization problem:

$$\begin{aligned} u_{0:T} = \arg\min_{\mathbb{U}_{0:T}} & \sum_{k=0}^{T} c(b_k^d, u_k) \\ s.t. \quad & b_{k+1}^d = \tau(b_k^d, u_k, z_{k+1}^{ml}), \quad b_0^d = b_n \\ & z_{k+1}^{ml} = \arg\max_z p(z|x_{k+1}^d) \\ & x_{k+1}^d = f(x_k^d, u_k, 0), \end{aligned} \tag{6}$$

In the second stage, it executes only the first action u_0 and discards the remaining actions in the sequence $u_{0:T}$. However, since the actual observation is noisy and is not equal to the z^{ml}, belief b_{n+1} will be different than b_1^d. Subsequently, RHC performs these two computations from the new belief b_{n+1}. In other words, RHC computes an open loop sequence $u_{0:T}$ from this new belief. This process continues until the belief reaches a desired belief location. Algorithm 3 recaps this procedure.

Algorithm 3: RHC for Partially-observable stochastic systems

input : Initial belief $b_{current} \in \mathbb{X}$, $B_{goal} \subset \mathbb{B}$
while $b_{current} \notin B_{goal}$ **do**
  $u_{0:T}$ = Solve the optimization in Eq.(6) starting from $b_0^d = b_{current}$;
  Apply the action u_0 to the system;
  Observe the actual z;
  Compute the belief $b_{current} \leftarrow \tau(b_{current}, u_0, z)$;

State-of-the-art methods such as [27] and [19] utilize this form of RHC in belief space. This framework is also called Partially-Closed Loop RHC (PCLRHC) [27] since it partially exploits some information about the future observations (i.e., z^{ml}) and does not fully ignore them.

Issues with RHC: There are some issues regarding the presented form of the RHC framework: First, due to the limited horizon and ignoring the cost-to-go beyond the horizon, the method may get stuck into pitfalls by choosing actions that guide the robot toward "favorable" states (with low cost) in the near future followed by a set of "unfavorable" states (with a high cost) in the long run. Second, the presented form of RHC ignores the stochasticity of the system within the horizon, which may lead to inaccurate approximations of the cost and unreliable control actions. To overcome these issues, researchers have proposed variants of RHC and different frameworks based on the idea of repeated planning [5]. Here, we discuss such a framework called "rollout policy" [5] and aim to realize it in belief space using the FIRM framework.

Rollout policy in belief space: A class of methods that aims to reduce the complexity of the stochastic planning problem in Eq.2 is the class of Rollout Policies (ROP) [5], which are more powerful than the described version of RHC in the following sense: First, they search for a sequence of policies (instead of open-loop controls) within the horizon, and do not approximate the system with a deterministic one. Second, they use a suboptimal policy, called the "base policy," to compute a cost-to-go function $\widetilde{J}$ that approximates the true cost-to-go beyond the horizon. In other words, at each step of the rollout policy scheme, the following closed-loop optimization is solved:

$$\begin{aligned} \pi_{0:T}(\cdot) = \arg\min_{\Pi_{0:T}} & \mathbb{E}\left[\sum_{k=0}^{T} c(b_k, \pi_k(b_k)) + \widetilde{J}(b_{T+1})\right] \\ s.t. \quad & b_{k+1} = \tau(b_k, \pi_k(b_k), z_k), \quad z_k \sim p(z_k|x_k) \\ & x_{k+1} = f(x_k, \pi_k(b_k), w_k), \quad w_k \sim p(w_k|x_k, \pi_k(b_k)) \end{aligned} \tag{7}$$

Then, only the first control law π_0 is used to generate the control signal u_0 and the rest of the policies are discarded. Similar to RHC, after applying the first control, a new sequence of policies is computed from the new point. The rollout algorithm is shown in Algorithm 4.

Algorithm 4: Rollout algorithm in Belief Space:

input : Initial belief $b_{current} \in \mathbb{B}$, $B_{goal} \subset \mathbb{B}$
while $b_{current} \notin B_{goal}$ **do**
  $\pi_{0:T}$ = Solve optimization in Eq.(7) starting from $b_0 = b_{current}$;
  Apply the action $u_0 = \pi(b_0)$ to the system;
  Observe the actual z;
  Compute the belief $b_{current} \leftarrow \tau(b_{current}, u_0, z)$;

Although the rollout policy in the belief space efficiently reduces the computational cost compared to the original POMDP problem, it is still formidable to solve, since the optimization is carried out over the policy space. Moreover, there should be a base policy that provides a reasonable cost-to-go $\widetilde{J}$. We propose a rollout policy in the belief space based on the FIRM-based cost-to-go.

FIRM-based Rollout Policy: In the FIRM-based rollout policy, we adopt the FIRM policy as the base policy of the rollout algorithm. Accordingly, the cost-to-go of the FIRM policy will be used as the cost-to-go beyond the horizon. Now, if we have a dense FIRM graph such that FIRM nodes partition the belief space (i.e., $\cup_i B^i = \mathbb{B}$), then at the end of the horizon, the belief b_{T+1} belongs to a FIRM node B from which the FIRM cost-to-go is available. However, in practice, when the FIRM nodes cannot cover the entire belief space, we need to make sure that a truncated policy can drive the belief into a FIRM node at the end of horizon. Nevertheless, since the belief evolution is random, we may not be able to guarantee that the belief reaches a FIRM node at the end of a deterministic horizon T. Therefore, instead of truncating the policy over a fixed time, we truncate the policy once the belief reaches a pre-specified stopping region (which happens in a random time denoted by $\mathcal{T}$) as follows:

$$\begin{aligned}
&\pi_{0:\infty}(\cdot) = \arg\min_{\Pi_{0:\infty}} \mathbb{E}\left[\sum_{k=0}^{\mathcal{T}} c(b_k, \pi_k(b_k)) + \widetilde{J}(b_{\mathcal{T}+1})\right] \\
&s.t. \quad b_{k+1} = \tau(b_k, \pi_k(b_k), z_k), \quad z_k \sim p(z_k|x_k) \\
&\qquad x_{k+1} = f(x_k, \pi_k(b_k), w_k), \quad w_k \sim p(w_k|x_k, \pi_k(b_k)) \\
&\qquad b_{\mathcal{T}+1} \in \cup_j B^j, \qquad (8)
\end{aligned}$$

where for $b_{\mathcal{T}+1} \in B^j$ we have

$$\widetilde{J}(b_{\mathcal{T}+1}) = J^g(B^j) \qquad (9)$$

The last condition in Eq.8 can be written more rigorously as $\mathbb{P}(b_{\mathcal{T}+1} \in \cup_j B^j | \pi) = 1$ for a finite $\mathcal{T}$. Also, as noted in Eq.(9), it is worth noting that the FIRM-based cost-to-go $J^g(\cdot)$ plays the role of the cost-to-go beyond the horizon $\widetilde{J}(\cdot)$.

Therefore, in solving the FIRM-based rollout policy problem, we aim to find a sequence of policies that ends up in a FIRM node and minimizes the cost in Eq.8. To find this optimal policy, we parametrize the policy space and perform minimization over the parameter space.

In our implementation, we adopt a variant of the Open-Loop Feedback Control (OLFC) scheme [5] along with a Kalman Filter as the belief controller. In this variant of OLFC, for a given $\mathbf{v}$, we compute an open-loop control sequence starting from the current estimation mean and ending at $\mathbf{v}$. Then, we apply a truncated sequence of the first l controls ($l = 5$ in our experiments). This process repeats every l steps until we reach the graph node. More details can be found in [1]. Therefore, the policy can be characterized by the next node; i.e., $\pi(\cdot; \mathbf{v})$. Thus, to solve the optimization in Eq.7 we search for the FIRM node $\grave{b}^j = (\mathbf{v}^j, P^j)$ whose mean, i.e., $\mathbf{v}^j$, leads to the best local policy $\pi(\cdot; \mathbf{v}^j)$. Accordingly, we implement the rollout technique in Algorithm 4.

V. Replanning in Changing Environments and Presence of Large Deviations

In this section, we discuss how we handle changes in the obstacle map and large deviations in the robot's belief. In general, handling these cases in belief space is a big challenge as they require online updating of the planning structure in belief space. It is important to note that it is the graph structure of FIRM that makes such an update and replanning feasible in real-time. The graph structure of FIRM allows us to *locally* change collision probabilities without affecting the rest of the graph (i.e., properties of different edges on the graph are independent of each other). It is important to note that such a property is not present in other state-of-the-art belief space planners, including SARSOP [15], BRM (Belief Roadmap Method) [21], or LQG-MP [28]. In those methods, collision probabilities and costs on *all* edges (number of possible edges is exponential in the size of underlying PRM) need to be re-computed.

A. Lazy Feedback Evaluation in Changing Environments

To adapt the proposed framework to handle changing environments, we rely on lazy evaluation methods. Inspired by the lazy evaluation methods for PRM frameworks [6], we propose a variant of the lazy evaluation methods for evaluating the generated feedback law. The basic idea is that at every node the robot re-evaluates *only* the next edge that it needs to take or a limited set of edges in the vicinity of the robot. By re-evaluation, we mean it re-computes collision probabilities along those edges. If there is a significant change in the local collision probabilities, then the dynamic programming problem is re-solved and a new feedback tree is computed. Otherwise, the feedback tree remains unchanged and the robot keeps following it. This lazy evaluation scheme can be performed in real-time. The method is outlined in Algorithm 5.

Algorithm 5: Lazy Feedback Re-Evaluation

```
input : Feedback π^g, current belief b_current
output : Updated feedback π^g
Update the obstacles map;
if there is a change in map then
    W ← Retrieve the sequence of nominal edges
    returned by feedback up to horizon l;
    forall the edges μ ∈ W do
        Re-compute the collision probabilities
        P_new(B, μ) from the start node B of edge;
    if exists μ ∈ W such that
    |P_new(B, μ) − P(B, μ)| > α then
        P(B, μ) ← P_new(B, μ);
        π^g ← Replan(b_current);
return π^g;
```

B. Handling Large Disturbances (kidnapped robot problem)

In robotics, the kidnapped robot problem commonly refers to a situation where an autonomous robot in operation is carried to an arbitrary location. This problem introduces different challenges such as (i) how to detect kidnapping, (ii) how to localize the robot, and (iii) how to control the robot to recover from this situation and accomplish its goal. The third part of this problem calls for online replanning in belief space.

Detecting a kidnapped situation: To detect the kidnapped situation, we constantly monitor the innovation signal $\widetilde{z}_k = z_k - z_k^-$ (the difference between actual and predicted observations). Recall that in our setting the observation at time step k from the j-th landmark is the relative range and bearing of the robot to the j-th landmark, i.e., ${}^j z_k = ({}^j r_k, {}^j \theta_k)$. The predicted version of this measurement is shown by ${}^j z_k^- = ({}^j r_k^-, {}^j \theta_k^-)$. We monitor the following measures of the innovation signal:

$$\widetilde{r}_k = \max_j(|{}^j r_k - {}^j r_k^-|), \quad \widetilde{\theta}_k = \max_j(d^\theta({}^j \theta_k, {}^j \theta_k^-)), \quad (10)$$

where $d^\theta(\theta, \theta')$ returns the absolute value of the smallest angle that maps θ onto θ'. Passing these signals through a low-pass filter, we filter out the outliers (temporary failures in the sensory reading). Denoting the filtered signals by $\overline{r}_k$ and $\overline{\theta}_k$, we monitor the conditions $\overline{r}_k < r_{max}$ and $\overline{\theta}_k < \theta_{max}$. If both of them are satisfied, we follow the FIRM feedback (i.e., we are in the *Feedback Following Mode* (FFM)). However, violation of either of these conditions means that the robot is constantly observing high innovations, and thus it is not in the location that it was supposed to be (i.e., it is kidnapped). In Section VI, we show the innovation signal for a sample run on a physical robot. In our implementation, we consider $r_{max} = 1$ (meters) and $\theta_{max} = 50$ (degrees).

Information Gathering Mode (IGM): Once the robot detects it has been kidnapped, the estimation covariance is replaced with a large covariance to get an approximately uniform distribution over the state space. Then, we enter the Information Gathering Mode (IGM), where we take small and conservative steps (e.g., turning in place or taking random actions with small velocities) to obtain more measurements. Once the robot gets these measurements, the localization module corrects the estimation value and the innovation signal reduces. When conditions $\overline{r}_k < r_{max}$ and $\overline{\theta}_k < \theta_{max}$ are satisfied again, we exit the information gathering mode.

Post-IGM replanning: After recovering from being kidnapped, controlling the robot in belief space remains a significant challenge because the system can be far from where it was expected to be. However, using the proposed method and assuming the FIRM graph has enough nodes distributed well in the space, the robot needs to go only to a neighboring node from this new point. Therefore, there is no need for a costly replanning procedure. Indeed, the only required computation is to evaluate the cost of edges that connect the new start point to the neighboring FIRM nodes based on Algorithm 1.

VI. Experimental Results

In this section, we first discuss the results of PRM and FIRM-based motion planning and show how belief space planning can improve the performance. Then, we distinguish our method from the state-of-the-art by examining and discussing the robustness properties of the proposed method to changes in the obstacle map, and to large deviations in the robot's location and the goal location. The experiments are conducted on a low-cost iRobot Create equipped with a laptop and an integrated monocular web-camera (Fig. 1(a)).

A. Planning with PRM and FIRM

The goal of this section is to compare the performance of FIRM with deterministic planners such as Medial Axis PRM (MAPRM) [29]. The solution of the dynamic programming problem, i.e., π^g, is visualized with a *feedback tree* (FT). For each node, FT contains only one outgoing edge ($\mu = \pi^g(B^i)$). FT is rooted at the goal node.

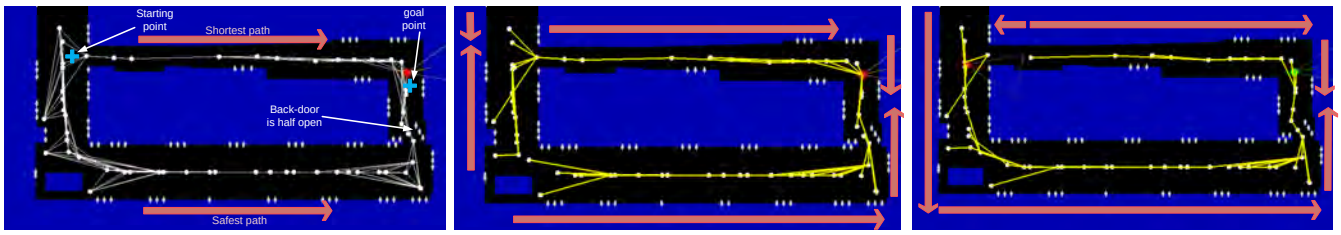
MAPRM-based planning: As one of the best variants of PRM when it comes to collision avoidance, we construct an MAPRM [29] in the environment (Fig. 2(a)). As is seen in Fig. 2(a), the path with maximum obstacle clearance (and the shortest path) is the one through the front door of room 407 (see Fig. 1(b)). Therefore, based on the obstacle clearance, MAPRM leads to the feedback tree shown in Fig. 2(b) that guides the robot through the front door. To execute the MAPRM plan we design LQG controllers to track the computed path. However, due to the lack of enough information along the solution path, the success rate of this plan is 27% (27 runs out of 100 Monte Carlo runs were successful) and the robot frequently collides with obstacles.

FIRM-based planning: In planning with FIRM, the information distribution in the environment is encoded in the planning via a framework which leads to a better judgement of the narrowness of passages in the belief space. Although in this environment the path through the front door is shorter, the success probability of traversing through the back door is more due to the presence of more information sources. Such knowledge about the environment is reflected in the FIRM cost-to-go and success probability. As a result, it generates a policy that suits the application, taking into account the uncertainty, and available information in the environment. Solving DP on the FIRM graph gives the feedback shown in Fig. 2(c), which results in an 88% success probability.

B. Robustness to Changes in the obstacle map

In this section, we investigate the robustness of the proposed algorithm to changes in obstacles for a physical system. In our experiments, we consider two types of obstacles. The first set of obstacles (most of the map) are static obstacles such as walls. The second class of obstacles include those that discretely change their state such as doors (state changes between "open" or "closed") in the environment. As discussed earlier, handling such changes is a challenge in state-of-the-art belief space planners since the planner cannot be updated locally and all computation for constructing the planner needs to be reproduced, which is not a feasible operation in real-time. The main focus of the following experiments is to demonstrate how our method can replan in real-time when faced with changes in the obstacle map.

We consider the environment shown in Fig. 1(b). The start and goal locations are shown in Fig. 3(a). We construct a PRM in the environment ignoring the changing obstacles (e.g., assuming all doors are open). Leveraging PRM to construct a FIRM and solving the dynamic programming problem on it, we get the feedback tree shown in Fig. 3(a)



(a) (b) (c)

Fig. 2. (a) The environment including obstacles (blue), free space (black), and landmarks (white diamonds) on the walls are shown. An MAPRM graph approximating the connectivity of free space, starting point, and goal point are shown. (b) The feedback tree generated by solving DP on MAPRM is shown in yellow. From each node there is only one outgoing edge (in yellow), computed by DP, guiding the robot toward the goal. Arrows in pink coarsely represent the direction on which the feedback guides the robot. (c) The feedback tree generated by solving DP on FIRM; As is seen, the computed feedback guides robots through more informative regions that leads to more accurate localization and less collision probabilities.

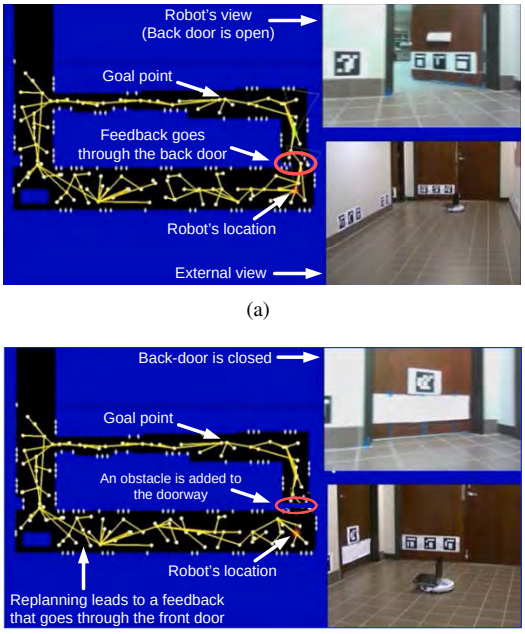
that guides the robot toward the goal through the back-door of room 407. However, the challenge is that the door may be closed when the robot reaches it, and there may be new obstacles in the environment. The robot needs to replan in real-time once it encounters such changes in the environment. For details on the obstacle detection mechanism see [1].

Figure 3(b) shows a snapshot of our run when the robot detects the door is in a different situation than expected. As a result, the robot updates the obstacle map as can be seen in Fig. 3(b), in which the door is closed. Accordingly, the robot replans in belief space based on Algorithm 5. Figure 3(b) shows the feedback tree resulting from replanning. As seen, the new feedback guides the robot through the front door, since it detects the back door is closed. The video of a long run (see Section VI-D) provides more detail on this procedure. Moreover, this video shows the robustness of the method to temporary failures in the perception system (e.g., missing landmarks due to blockages, blur, etc.), which is discussed more in [1].

C. Robustness to large deviations

In this section, we investigate the robustness of the proposed framework in dealing with large deviations in the robot's position. As a more general form of this problem, we consider the *kidnapped robot problem* as discussed in the previous section. The need for online replanning in belief space makes this problem challenging.

Figure 4(a) shows a snapshot of a run that involves two kidnappings and illustrates the robustness of the planning algorithm to the kidnapping situation. The start and goal positions are shown in Fig. 4(a). The feedback tree (shown in yellow) guides the robot toward the goal through the front door. However, before reaching the goal point the robot gets kidnapped in the hallway (cf. Fig. 4(a)) and placed in an unknown location within room 407 (cf. Fig. 4(a)). The first jump in 4(b) shows this deviation. Once the robot recovers from being kidnapped (i.e., when both innovation signals in Fig. 4(b) fall below their corresponding thresholds), replanning from the new point is performed. Feedback guides the robot toward the goal point from within room 407. However, again, before robot reaches the goal point, it is kidnapped and placed in an unknown location (see Fig. 4(a)). The second jump in the innovation signals in Fig. 4(b) corresponds to this kidnapping. Again, replanning from the new point, the robot follows the feedback and reaches the goal point.



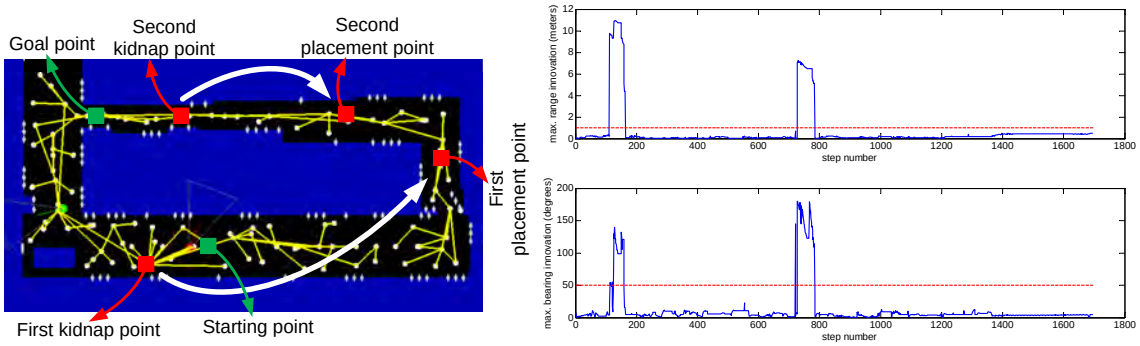
(a)

(b)

Fig. 3. (a) The back door is open at this snapshot. The feedback guides the robot toward goal through the back door. (b) The back door is closed at this snapshot. Robot detects the door is closed and updates the obstacle map (adds door). Accordingly robot replans and computes the new feedback. The new feedback guides the robot through the front door.

D. A Longer and more complex experiment

We next demonstrate the ability of the system to perform long-term tasks in a complex scenario that consists of visiting several goals (each time therobot reaches a goal, a user submits a new goal). The replanning ability allows the robot to change the plan online in belief space as the goal location changes. Moreover, the robot frequently encounters changes in the obstacle map (open/closed doors and new obstacles in the environment) as well as missing information sources and kidnapped robot situations. Thus, the robot frequently needs to perform a replanning operation in belief space to deal with such frequent changes. A 25 minute video of this run is recorded and available in [17] (a shorter version has been submitted along with the paper) that shows the robot's performance in this complex scenario. In this video, the robot faces three changes in the goal location, three changes in the door's state (open/closed), several new obstacles in



(a) (b)

Fig. 4. (a) The set up for the experiment containing two kidnapping. (b) Innovation signals $\widehat{r}_k$ and $\widehat{\theta}_k$ during this run. When both of the signals are below specified thresholds r_{max} and θ_{max} (dashed red lines), robot follows the FIRM feedback. Otherwise, the system enters the information gathering mode.

the environment, three kidnapping situations, and numerous failures of the sensory systems due to missing landmarks, blur in image, and etc.

VII. Conclusion

In this paper, we present an application of the FIRM motion planning method to a physical robotic system. This paper proposes a robust method for belief space planning based on efficient online replanning. Such replanning is a key ability in handling discrepancies between real world models and computational models, changes in the environment and obstacles, large deviations, and changes in information sources. We implemented this belief space planner on a physical system and demonstrate the robustness to such discrepancies that occur in practice. We believe this work provides an important step toward making POMDP methods applicable to real world robotic systems.

References

[1] Aliakbar Agha-mohammadi, Saurav Agarwal, Aditya Mahadeval, Daniel Tomkins, Jory Denny, Suman Chakravorty, and Nancy M. Amato. Dynamic real-time replanning in belief space: An experimental study on physical mobile robots. *Technical Report: TR13-007, Parasol Lab., CSE Dept., Texas A&M University*, 2013.

[2] Aliakbar Agha-mohammadi, Suman Chakravorty, and Nancy Amato. FIRM: Feedback controller-based Information-state RoadMap -a framework for motion planning under uncertainty-. In *International Conference on Intelligent Robots and Systems (IROS)*, 2011.

[3] Aliakbar Agha-mohammadi, Suman Chakravorty, and Nancy Amato. FIRM: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements. *International Journal of Robotics Research*, 33(2):268–304, Feb. 2014.

[4] Haoyu Bai, David Hsu, Wee Sun Lee, and Vien A. Ngo. Monte carlo value iteration for continuous-state pomdps. In *WAFR*, volume 68 of *Springer Tracts in Advanced Robotics*, pages 175–191. Springer, 2010.

[5] Dimitri Bertsekas. *Dynamic Programming and Optimal Control: 3rd Ed.* Athena Scientific, 2007.

[6] Robert Bohlin and Lydia E Kavraki. Path planning using lazy prm. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 521–528. IEEE, 2000.

[7] Adam Bry and Nicholas Roy. Rapidly-exploring random belief trees for motion planning under uncertainty. In *ICRA*, pages 723–730, 2011.

[8] S. Chakravorty and R. Scott Erwin. Information space receding horizon control. In *IEEE Symposium on Adaptive Dynamic Programming And Reinforcement Learning (ADPRL)*, April 2011.

[9] Pratik Chaudhari, Sertac Karaman, David Hsu, and Emilio Frazzoli. Sampling-based algorithms for continuous-time pomdps. In *the American Control Conference (ACC)*, Washington DC, 2013.

[10] Tom Erez and William D Smart. A scalable method for solving high-dimensional continuous pomdps using local approximation. In *the International Conference on Uncertainty in Artificial Intelligence*, 2010.

[11] Devin Grady, Mark Moll, and Lydia E. Kavraki. Automated model approximation for robotic navigation with POMDPs. In *ICRA*, 2013.

[12] R. He, E. Brunskill, and N. Roy. Efficient planning under uncertainty with macro-actions. *Journal of Artificial Intelligence Research*, 40:523–570, February 2011.

[13] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

[14] H. Kurniawati, T. Bandyopadhyay, and N.M. Patrikalakis. Global motion planning under uncertain motion, sensing, and environment map. *Autonomous Robots*, pages 1–18, 2012.

[15] H. Kurniawati, D. Hsu, and W.S. Lee. SARSOP: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In *Proceedings of Robotics: Science and Systems*, 2008.

[16] Duan Li, Fucai Qian, and Peilin Fu. Variance minimization approach for a class of dual control problems. *IEEE Trans. Aut. Control*, 47(12):2010–2020, 2002.

[17] Video on Parasol Lab. webpage. http://youtu.be/m3t3udm0ftu.

[18] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *International Joint Conference on Artificial Intelligence*, pages 1025–1032, 2003.

[19] R. Platt. Convex receding horizon control in non-gaussian belief space. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2012.

[20] Robert Platt, Russ Tedrake, Leslie Kaelbling, and Tomas Lozano-Perez. Belief space planning assuming maximum likelihood observations. In *Proceedings of Robotics: Science and Systems (RSS)*, June 2010.

[21] Sam Prentice and Nicholas Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *International Journal of Robotics Research*, 28(11-12), October 2009.

[22] Shridhar K. Shah, Chetan D. Pahlajani, Nicholaus A. Lacock, and Herbert G. Tanner. Stochastic receding horizon control for robots with probabilistic state constraints. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[23] R. D. Smallwood and E. J. Sondik. The optimal control of partially observable markov processes over a finite horizon. *Operations Research*, 21(5):1071–1088, 1973.

[24] T. Smith and R. Simmons. Point-based pomdp algorithms: Improved analysis and implementation. In *Proceedings of Uncertainty in Artificial Intelligence*, 2005.

[25] M. Spaan and N. Vlassis. Perseus: Randomized point-based vallue iteration for pomdps. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.

[26] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, 2005.

[27] Noel Du Toit and Joel W. Burdick. Robotic motion planning in dynamic, cluttered, uncertain environments. In *ICRA*, May 2010.

[28] Jur van den Berg, Pieter Abbeel, and Ken Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *International Journal of Robotics Research*, 30(7):895–913, 2011.

[29] Steven A Wilmarth, Nancy M Amato, and Peter F Stiller. MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 1024–1031, 1999.

Multi-agent Generalized Probablistic RoadMaps : MAGPRM

Sandip Kumar and Suman Chakravorty

***Abstract*— In this paper, the generalized motion planning algorithm (Generalized PRM : GRPM [1, 2, 3]) is extended to a class of multi-agent motion planning problem in presence of process uncertainty and stochastic maps.**

The proposed algorithm is a hierarchical approach towards constructing a passive coordination strategy which utilizes an existing multiple traveling salesman problem (MTSP) solution methodology in conjunction with the GPRM framework to solve the multi-agent motion planning problem. The proposed algorithm is generalized to tackle multi-agent problems involving heterogeneous agents. The algorithm is used to solve multi-agent motion planning problems involving 2-dimensional and 3-dimensional agents in stochastic maps with uncertainty in the motion model. Results indicate that the algorithm successfully solves the problem under uncertainty, and generates a solution having high probability of success. It also demonstrates that the algorithm is scalable in terms of number of start and goal locations, the number of agents and their dynamics.

I. Introduction

The motion planning problem for multiple agents, in environments with obstacles is a challenging problem in robotics. The challenge arises due to searching for a solution in the joint state and control spaces of the agents. Introduction of motion model uncertainty increases the complexity even further. Finding a solution to the coordination problem [4], and hence, developing coordination strategies for multi-agent systems in the presence of uncertainties has been a challenge.

The multiple agents motion planning problem in the presence of uncertainty and coordination strategies has been addressed previously in the literature. Some of the related work is mentioned here. In [5], motion primitives are used to handle multiple agent motion planning problem. In [6], distributed cooperative strategies for a group of robotic manipulators was proposed using neural networks. This work accounted for control input uncertainties. In [7], the multiagent motion planning problem is setup as a Markov decision process (MDP) with constraints and uncertainty, where the coupling between agents only occur in the rewards and constraints. The optimal move in the presence of constraints are computed using optimization algorithms, however, the dynamics of the agents is not considered. In [8], a cooperative control technique is proposed which creates a communication graph. Under certain assumptions, the dynamics of non-holonomic agents are modeled as kinematic point robot chains, and the cooperative laws are defined on this reduced model of multi-agent system (MAS). In [9], the author has developed a decentralized multiagent RRT, and a merit-based token passing coordination strategy. With respect to team formation strategies, uncertainties were considered in communications, but not in the motion model. In [10], the authors developed a non-cooperative approach for collision detection and avoidance. Genetic algorithm and Monte Carlo techniques were used to address uncertainties in motion model. In [11], an online receding horizon motion planner is developed for solving the decentralized navigation problem for multiple agents. Artificial potential fields and sliding mode control techniques are used to handle uncertainties in the motion model.

Some other work, involving multiple agents coordination without uncertainty in the system, are worth mentioning here. In [12], recent work related to the multiple agent motion coordination problem is summarized. In [13], the distributed path consensus (DPC) algorithm is extended to multiple task execution. The DPC was developed to address multiagent motion planning problem with time parameterized constraints on the distances between a pair of robots. In this work, the state space is combined with a task graph, and optimal trajectories for the agents are searched in this graph using a heuristic function. In [14], the motion planning problem for multiple agents was addressed using coordination graphs and decoupled planning. The agent works on path following and obstacle avoidance. In [15], the authors developed a navigation function methodology for decentralized navigation which leads to reduced computational complexity and increased robustness to agent failures. They deal with holonomic agents. In [16], the authors were able to decompose global payoff function for multiple agent scenario into local terms using a coordination graph. They used reinforcement learning techniques, known as, sparse cooperative Q-learning, and used edge-based decomposition of the actions in the coordination graph. In [17], coordination between agents is developed using the theory of collective intelligence through probability collectives.

The work mentioned above solves the coordination problem for multi-agents, using various different methodologies. However, a systematic framework to incorporate motion model uncertainty, non-trivial non-linear dynamics of robots, and solve for high-dimensional state-space systems, is still missing in these efforts. In the work presented in this paper, a systematic hierarchical approach is presented to solve the multiagent motion planning problem in the presence of motion model uncertainty, stochastic maps, and with non-trivial agent dynamics. We solve a multi-agent Markov decision process (MMDP) by decoupling the problem using a hierarchical planning structure and multiple MDPs. In this

Sandip Kumar is an Employee of MathWorks, 3 Apple Hill Dr, Natick, MA 01760, USA to.sandip@gmail.com

Suman Chakravorty is a Faculty of Department of Aerospace Engineering, Texas A&M University, College Station, TX - 77840, USA schakrav@aeromail.tamu.edu

paper, we propose to solve the coordination problem, between agents, by developing a passive coordination strategy, using the Generalized PRM (GPRM) [1, 2], developed by the authors for the single agent feedback motion planning problem, in conjunction with a well proven multiple traveling salesman problem solution methodology [18], for a class of multiagent systems.

In section II, the framework developed by the authors to handle single agent motion planning problem under process uncertainty, with non-linear dynamics, and in high dimensional configuration space, is briefly presented [2]. In section III, the actual multi-agent motion planning problem we intend to solve in this paper, is proposed. In section IV, the solution methodology is presented. In section V, the solution of the "Routing Problem" is detailed. In section VI, results from simulations are presented.

II. Motion planning for a single agent under process uncertainty

The motion planning problem, for a single agent, is to find a collision free path for a robot in a given obstacle space. In the presence of stochastic model uncertainty, there is a need for feedback control, which then can be associated with a probability that the robot reaches the goal without hitting the obstacles. Generalized Sampling Based Algorithms [1, 2] were introduced, by the authors, to address the problem of feedback motion planning in such constrained work spaces. The notion of collision avoidance and collision-free paths as the solution to the motion planning problem, can no longer be satisfied, and therefore the above criteria need to be replaced by a solution/ path with a high probability of success. The motion planning problem is then re-framed as : *To solve the motion planning problem in the presence of stochastic maps, and model uncertainty, generate a feedback solution with a probability of success above an a-priori specified probability,* p_{min}.

A. Generalized PRM : GPRM

If the uncertainties in the robot model and environment can be modeled probabilistically, the robot motion planning problem can be formulated as Markov Decision Problem (MDP). These MDPs are computationally intractable for anything but small state/ control spaces, and especially hard to solve in continuous state and control spaces. Hierarchical Methods can be used to break down the complexity of the problem. The *Generalized Probabilistic Roadmaps*(GPRM) [2, 1], is a sampling based hierarchical method which extends the *Probabilistic Roadmaps* (PRM) [19] technique for deterministic path planning, to systems with stochastic model and map uncertainty. GPRM incorporates feedback controllers into the topological graph construction phase.

The authors have also developed adaptive sampling technique, termed as "adaptive GPRM" (AGPRM), [3, 20] to increase the efficiency and overall success probability of these planners, especially in high dimensional spaces. The technique have been implemented on high-dimensional n-link manipulators, with up to 8 links. The results demonstrate the ability of the proposed algorithm to handle the feedback motion planning problem for highly non-linear systems, in very high-dimensional state spaces.

In this paper, this work is extended to solving stochastic motion planning problem for a class of heterogeneous multi-agent systems.

III. Multi-agent systems

Multi-agent system (MAS) consists of multiple agents which execute actions and influence their surroundings. Each agent receives observations and selects actions individually, but it is the resulting *joint action* which influences the environment and generates the reward[1] for the agents. This has extremely important consequences on the characteristics and the complexity, of the problem.

A. Coordination Problem

The multi-agent motion planning problem in presence of process uncertainty and stochastic maps can be posed as an *multi-agent Markov decision process* (MMDP). In traditional methods of solving an MMDP, the problem is treated as a single large MDP and standard solution techniques available for MDPs are applied, [21]. The goal of solving an MMDP should be that of finding the best/optimal policy for the system of agents. However, actions are taken at the individual level of the agents, and thus, it should be ensured that, using limited communication, the combined actions of all the agents should result in an optimal policy for the system of agents. The problem of identifying individual policies for each agent, which results in an optimal joint policy, is called the **coordination problem**.

Researchers working on solving this problem have come up with possible solutions such as *coordination graphs* (CGs) [22] and *max-plus* algorithm in conjunction with CGs [23]. In [22], the solution to cooperative action selection for a system of agents (or coordination problem) is solved by constructing a *coordination graph*, and optimizing over it using the variable elimination algorithm. In [23], the researchers proposed an improved optimization technique, the *max-plus* algorithm, which replaces the variable elimination procedure in optimizing over the *coordination graphs*.

B. A Class of Multi-Agent Problems in the Presence of Uncertainty

We want to solve the multi-agent motion planning problems, in the following scenario :

- m agents, with m initial configurations $q_I = \{q_{I_1}, q_{I_2}, \ldots, q_{I_m}\}$,
- n goal locations $q_G = \{q_{G_1}, q_{G_2}, \ldots, q_{G_n}\}$,
- Process uncertainty[2] present in robot motion model,
- Environment given by a stochastic map, i.e., static obstacle probabilities.

The problem statement can be stated as : *Given a stochastic map with static obstacle probabilities, a system of m heterogeneous robots each equipped with perfect state sensors, the*

[1] cost of transition

[2] also called motion model uncertainty

initial configurations (q_I) of all the robots, a set of n final goal configurations (q_G), to solve, the motion planning problem for the set of robots in the presence of process uncertainty such that at least one robot visits each of the goal locations, while the total cost of operation for the system is minimized

In order to solve the multi-agent problem, the following sub-problems have to be solved :

- **Routing problem** : The number of agents and number of goal locations might not be same, i.e. $m \neq n$ (general case). The goal locations are different in number, some agents will have to go to more than one goal and some might not have to go to any goal. Hence, with the given scenario, one has to solve a *routing problem* (or the *coordination problem* as discussed in subsection III-A) for the multi-agent system.
  This is the problem of identifying which agents will go to which goal locations. Hence, given m agents and their initial configurations, $q_I = \{q_I(i)\},\ i = 1,\ldots,m$, and n target final configurations, $q_G = \{q_G(i)\},\ i = 1,\ldots,n$, and given $m \neq n$ (general case), how to determine which set of goals any given agent will go to.
- **Heterogeneous and homogeneous agents** : In a general multi-agent system, there exist heterogeneous agents, i.e. agents with different capabilities (or multiple types of agents). Thus, homogeneous, as well as the heterogeneous agent scenario must be addressed in a multi-agent motion planning problem.

IV. Solution Approach to Multiagent Motion Planning Problem in Presence of Uncertainty

This section discusses the solution approach to solve the sub-problems of the multi-agent motion planning problem.

A. Routing Problem

The *routing problem*, in a deterministic framework, has been solved extensively in the *traveling salesman problem* (TSP) research community [24]. The generalized multi-agent routing problem, in deterministic framework, has been posed as a *multiple traveling salesman problem* (MTSP) [25, 18], and there have been multiple approaches to solve it. We aim to use existing multiple agents routing problem solution techniques developed in [18], and synergistically, apply it along with the GPRM to the multi-agent systems, in presence of process uncertainty, and stochastic maps. The solution of GPRM, between any pair of goal locations[3] for any given agent, generates transition costs, and probabilities. The MTSP algorithm, uses these costs and transition probabilities from the GPRM, to solve the "routing problem", and hence, solve the multi-agent motion planning problem under uncertainty. We expect that this generalized technique, termed - *the multi-agent adaptive sampling based generalized probabilistic roadmaps* (MAGPRM), will help us solve the feedback motion planning problems in high dimensional state spaces, under uncertainty, in the multiple agent scenario.

[3]Goal locations here includes the initial configurations and the desired configurations

B. Homogeneous and Heterogeneous Agents

In the case of all agents being homogeneous in dynamics and capabilities, a single roadmap (GRPM) is adequate, for solving the motion planning problem for an agent between any pair of the goal/start locations. For heterogeneous agents, solving the motion planning problem will involve constructing roadmaps (GPRMs) for every type of agent present in the system. (see Figure 1)

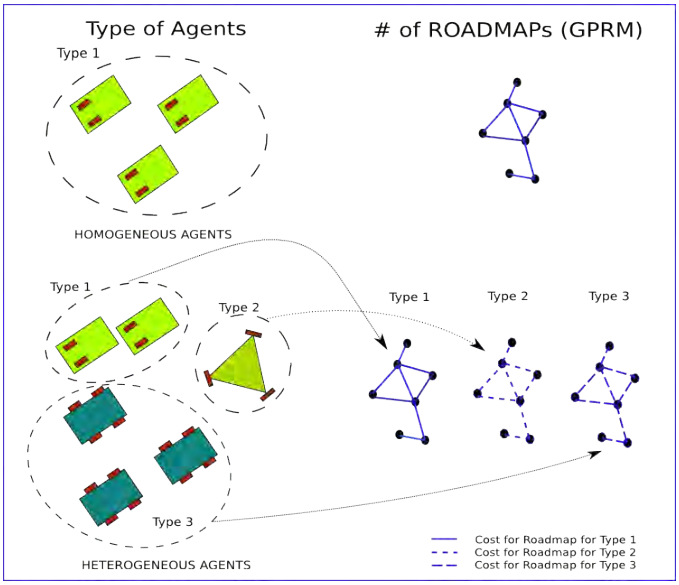


Fig. 1. Multiple Agents and the number of associated GPRMs

V. The Solution

In this section we develop the solution to multi-agent motion planning problem by developing it for the sub-problems of *routing* the agents and having *heterogeneous agents* in a multi-agent scenario. We develop a synergistic coupling of MTSP solutions with GPRM to solve the *routing problem*.

A. Definitions

1) General:

$\mathscr{C}$: The configuration space of the agent. A configuration is given by q, i.e a generalized position.

$\mathscr{X}$: The state-space of the agent. A state is given by, $x = (q, \dot{q})$, i.e. comprised of generalized position and generalized velocity.

l^i : i^{th} landmark, i.e. a sample in state-space ($l^i \in \mathscr{X}$).

$\mathscr{L}$: Set of landmarks, i.e. $\mathscr{L} = \{l^i\}, \forall i$, on a given stochastic map.

$\mathscr{G}$: Set of start and goal locations[4] in a multiple agents scenario on a given stochastic map. This set containing these start and goal locations, is a sub-set of the set of landmarks, i.e. $\mathscr{G} \subset \mathscr{L}$

$\mathscr{A}$: Set of all agents, $\{a_i\}, \forall i$. (a_i is the i^{th} agent)

$\mathscr{U}$: Set of controls. ($u \in \mathscr{U}$)

$\mathscr{M}$: Set of lower level controllers. ($\mu \in \mathscr{M}$)

[4]landmarks

2) Controls:

$\mu(\cdot)$: The lower level ($Level_1$ in MAGPRM in Figure 2) controller for the agents. In MAGPRM it is a feedback controller, parametrized using the landmark. Also $\mu \in \mathscr{M}$ and :

$$\mu(\cdot) : \mathscr{X} \mapsto \mathscr{U}$$

$\pi(\cdot)$: Policy operator at $Level_2$ of MAGPRM (Figure 2), i.e. solution of GPRM for a single agent.

$$\pi(\cdot) : \mathscr{L} \mapsto \mathscr{M}$$

Given a goal landmark, l^{goal}, π is a solution provided by GPRM. This solution is dependent on l^{goal} and hence the operator π can be rigorously written as follows:

$$\pi(\cdot \ ; \ l^{goal}) : \mathscr{L} \mapsto \mathscr{M}$$

$\gamma(\cdot)$: An operator at $Level_3$ of MAGPRM.

$$\gamma(\cdot) : \mathscr{A} \times \mathscr{G} \mapsto \mathscr{G}$$

Given a particular agent $a_i \in \mathscr{A}$, and the location($\in \mathscr{G}$) of a_i, say $g \in \mathscr{G}$, the operator outputs the next goal location for a_i, i.e. $g' \in \mathscr{G}$. This g' parametrizes the $Level_2$ $\pi(\cdot)$ operator, i.e.:

$$\pi(\cdot \ ; \ g') : \mathscr{L} \mapsto \mathscr{M}$$

Furthermore in terms of goal landmark, $l_i^{goal} \in \mathscr{L}$ for the i^{th} agent, the location $g' \in \mathscr{G}$ where $\mathscr{G} \subset \mathscr{L}$, is given by :

$$g' = l_i^{goal}, \text{ and hence}$$
$$\pi(\cdot \ ; \ l_i^{goal}) : \mathscr{L} \mapsto \mathscr{M}, \text{ for } i^{th} \text{ agent}$$

This operator provides the agent, starting at a start location ($\in \mathscr{G}$), the goal location ($\in \mathscr{G}$) it is supposed to go, and a sequence of feedback controllers ($\in \mathscr{M}$) to reach there.

B. Solution of MTSP

In [18], a solution methodology is proposed for the generalized MTSP problem formulation. The solution methodology involves two transformation steps:

- Converting a generalized MTSP to a *one-in-a-set ATSP* (where ATSP : Asymmetric Traveling Salesman Problem).
- Converting a one-in-a-set ATSP to a single ATSP. This is done using the *Noon-Bean Transformation* [26].

The generalized MTSP is posed as a single ATSP by the proposed transformations which involve cost modifications. The single ATSP can be solved using the well-known TSP solver, Lin-Kernighan heuristic (LKH) [24]. Solving the single ATSP and working backwards gives the solution to the generalized MTSP. Details of the algorithm developed can be seen in [18].

C. Solving Multi-Agent Systems in Presence of Uncertainty

The GPRM was developed [2] as a hierarchical approach. The proposed algorithm, MAGPRM, for solving the multi-agent motion planning involves the introduction of a new level in the existing hierarchy. The lowest level (say $Level_1$ in Figure 2) solves the motion planning problem between one landmark to another and inherently generates the cost of transition and transition probabilities between two landmarks. These transition probabilities and costs induced an abstract MDP, on the discrete set of landmark states, i.e. the higher level ($Level_2$ in Figure 2). We use Dynamic Programming to solve the $Level_2$ abstract MDP with the transition cost and probabilities generated by the $Level_1$.

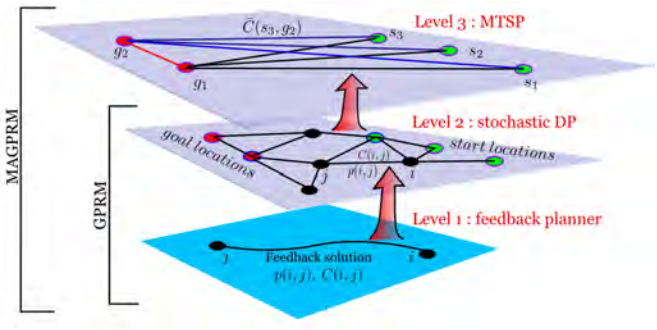


Fig. 2. Depicting Hierarchical Planning in Levels (for Multi Agents)

In a multi-agent motion planning scenario, having m agents and n goal locations, an additional problem needs to be solved, namely the *routing problem*. In order to solve the *routing problem* we introduce $Level_3$ which comprises of a graph whose vertices are high level goal locations ($g \in \mathscr{G}$), i.e. the m agents' initial locations and the n desired goal locations. These set of goal locations ($\mathscr{G}$) is a sub-set of the set of landmarks (i.e., $\mathscr{G} \subset \mathscr{L}$. See Figure 2). At $Level_2$, these goal locations ($\mathscr{G}$) are treated as landmarks. Using $\mathscr{L}$ on $Level_2$, multiple GPRMs (i.e., generalized roadmaps) are constructed over which the 'single-agent' motion planning problem is solved. Whereas, using $\mathscr{G}$, a MAGPRM (i.e., a multi-agent roadmap) is constructed on $Level_3$ over which the 'multi-agent' motion planning problem is solved. The edges of the graph in $Level_3$ are abstract connections from "agent locations to goal locations" and "goal to goal locations". "Agent locations to agent locations" connections are avoided as a part of assumption that an agent should not go to another agent's location.

Using GPRMs, the cost of transition and the path probability associated with all these edges in $Level_3$, can be computed (Figure 2). These costs and transition probabilities associated with every edge is specific to agents. The number of GPRMs that needs to be solved are $2n(m+n)$[5]. Using the computed costs[6] and after two cost transformations, the prospective

[5]These are the number of edges that needs to be evaluated. m agents to n goals and vice versa gives $2mn$ edges, n goals to n goals gives $2n^2$ edges and hence the total is $2n(m+n)$

[6]The MTSP algorithm only takes costs as input, the costs computed by GPRM do take into account the transition probabilities.

MTSP algorithm [18] via the LKH solver solves the *routing problem*, i.e. allotment of sequence of goal locations to different agents.

In this MTSP level, i.e. $Level_3$ of MAGPRM, the solution is the operator $\gamma(\cdot)$. For each agent, the operator outputs a sequence of goal locations to be visited. We mention here that unlike in $Level_1$ and $Level_2$, the solution ($\gamma(\cdot)$) is not a feedback policy at $Level_3$, since if the sequence is broken by the agents due to some plausible reason, the policy does not remain optimal, and the paths of the different agents need to be replanned using the MTSP solver.

D. Multi-Agent GPRM (MAGPRM) Algorithm

The proposed methodology of solving the multi-agent motion planning problem is summarized in algorithm 1.

Algorithm 1: Multi-Agent GPRM (MAGPRM)

Data: Set of agents ($\mathscr{A}$), start locations x_0, goal locations x_g, p_{min} for the environment

1 **for** i^{th} *agent at start location* $x_{0_i} \in x_0$ **do**
2 **for** j^{th} *goal location,* $x_{g_j} \in x_g$ **do**
3 **while** $p_s(x_{0_i} \rightarrow x_{g_j}) < p_{min}$ **do**
4 **if** i^{th} *agent's type already evaluated* **then**
5 Use already existing roadmap to build and connect further;
6 Construct AGPRM, parametrized with goal location x_{g_j} and agent-type of i^{th} agent;
7 Construct a cost of transitions matrix for each agent-type (i.e. cost of transitions between *agents* $\leftrightarrow$ *goals* and *goals* $\leftrightarrow$ *goals*);
8 Solve the *routing problem* for each agent, using the above generated costs in prospective MTSP algorithm;

The proposed algorithm solves the general[7] multiple agent motion planning problem in presence of process uncertainty and stochastic maps.

The algorithm constructs a roadmap using AGPRM between each pair of start and goal locations. The loop at *line 3* depicts this, i.e., with each combination of start and goal locations, a new AGPRM (which is parametrized at the current goal location), is solved.

In the case of multiple agents of same type, *lines 4-5* ensure that the existing roadmap, is either extended further, or is used to find a solution, between the ith agent's location and the jth goal location.

In the case of heterogeneous agents, different roadmaps for different types of agents have to be constructed and hence, is more computationally intensive. The landmarks might still be shared but transition costs and transition probabilities calculations will involve running the simulations and constructing a different roadmap each time a new type of agent comes into the system. *Line 7* emphasizes this feature of the algorithm.

Once the cost of transitions between each start and goal locations is computed, the routing problem is solved using the prospective MTSP algorithm. The solution of MAGPRM is the sequence of goal locations to be visited by individual agents, which takes into account the process uncertainty in the dynamics of the agents, and their traversal along a stochastic map.

VI. Results and Discussion

In this section, we will detail the application of the multi-agent GPRM (i.e. MAGPRM) algorithm to scenarios with homogeneous and heterogeneous agents. The agents involved in these numerical experiments are a unicycle and a simplified three dimensional vehicle, mimicking a helicopter.

A. Vehicle Models Used

The numerical experiments done using MAGPRM involves the following two types of robot models used along with their specific feedback controllers.

1) Nonholonomic Unicyle robot: The equations of motion are given by :

$$\dot{x} = v\cos\theta + w_x \tag{1}$$
$$\dot{y} = v\sin\theta + w_y \tag{2}$$
$$\dot{\theta} = \omega + w_\theta \tag{3}$$

where (x, y, θ) represents the pose of the robot, the velocity v, and the angular velocity ω, represents the control inputs to the problem, and w_x, w_y and w_θ are the uncorrelated noise terms for the different states of the robot. A sampled pose is in the (x, y, θ) spaces and the local feedback controller used to stabilize the robot about these sampled equilibrium configurations is given by [27], a dynamic feedback linearization-based controller.

2) Simplified 3D helicopter robot: A simplified three-dimensional helicopter robot is constructed using a Dubins car for inplane motion (as in [1]) and a decoupled double integrator in the $z-$direction. Hence the dynamics of this simplified robot can be given by:

$$\dot{x} = v\cos\theta + w_x \tag{4}$$
$$\dot{y} = v\sin\theta + w_y \tag{5}$$
$$\dot{\theta} = \omega + w_\theta \tag{6}$$
$$\ddot{z} = u_z + w_z \tag{7}$$

Apart, from the definitions discussed above, u_z is the input force in $z-$direction, and, w_z is uncorrelated noise in the same direction. Our sampled poses are in the $(x, y, \theta, z, \dot{z})$ spaces. The local feedback controllers can stabilize the robot about any of the equilibrium configurations sampled.

A dynamic feedback linearization-based controller design is chosen as in [1] for the Dubins car model, for in-plane motion. An LQR based feedback controller is designed for the double integrator in $z-$direction as in [2].

The values of w_x, w_y, w_θ and w_z are similar to those dicussed in [1, 2].

B. Homogeneous Agents

In these numerical experiments, multiple homogeneous agents[8] starting at different locations on a stochastic map

[7] Involving heterogeneous agents

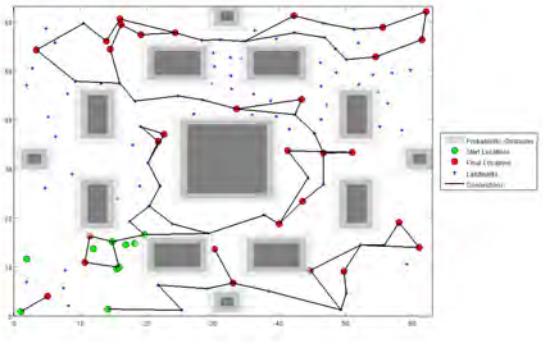
[8] All agents are Dubins car

were supposed to cover a given number of goal locations. As the agents are homogeneous, the *cost of transition* and the *probability of transition* from one landmark to another is the same, given that all agents are working with the same map and the same sampled landmarks.

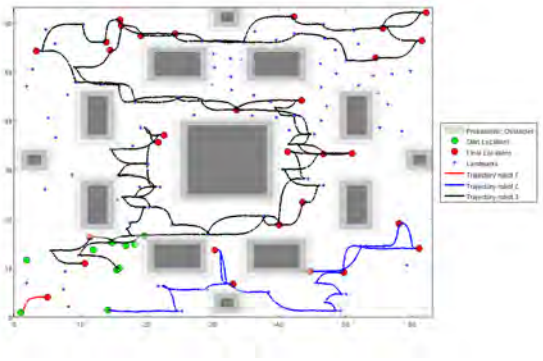
The result of our simulation experiments are shown in Figure 3 and Figure 4. Figure 3(a), Figure 4(a) and Figure 4(b) show three different cases, i.e. different number of agents starting at different start locations and that have to visit a different number of goal locations.

Figure 3 depicts a case in which all the start locations for the robots were constricted to a smaller region compared to the spread of the goal locations. Figure 3(a) shows the solution of MAGPRM (i.e. at $Level_2$ of MAGPRM) in terms of the goal locations to be visited by the active[9] agents and the various landmarks used to navigate through those assigned goal locations. Figure 3(b) shows the actual trajectories (i.e. at $Level_1$ of MAGPRM) of the active agents based on the dynamics and the corresponding feedback controller.

The solution shows that MTSP (i.e. at $Level_3$ of MAGPRM) is driven by space partitioning. And hence, there are some agents who do not move from their initial locations.



(a) MAGPRM - Solutions for individual vehicles

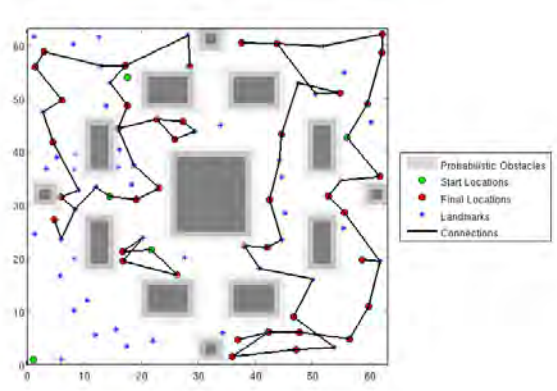


(b) MAGPRM - Trajectories for individual vehicles

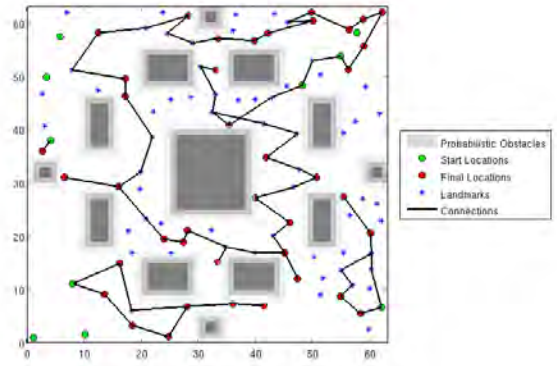
Fig. 3. MAGPRM Solutions and Trajectories - 10 vehicles and 30 goal locations

Figure 4(a) and Figure 4(b) show results obtained by MAGPRM, depicting coverage of goal location by the agents

[9]In MAGPRM solution not all agents need to move and hence active agents are the ones which have been assigned atleast one goal location.



(a) MAGPRM - 5 vehicles and 40 final locations



(b) MAGPRM - 10 vehicles and 40 final locations

Fig. 4. MAGPRM Solutions

starting from different start locations. Both the solutions have 40 goal locations and the number of agents are 5 and 10 respectively. The solution of MAGPRM in these solutions also show the space partitioning behavior.

C. Heterogeneous Agents

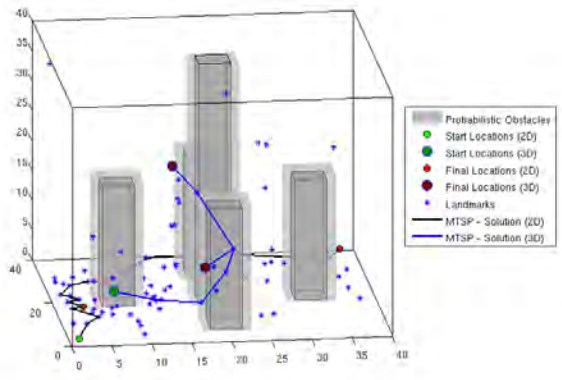
In these set of experiments, heterogeneous[10] agents are present in the map. A three dimensional static stochastic map is used for these simulations. In each of these simulations there are several 3-dimensional and 2-dimensional goal locations. The Dubins car can only cover the 2-dimensional goal locations, and the simplified 3D helicopter robot can traverse to both 2-dimensional, as well as, 3-dimensional goal locations.

The initial set of landmarks sampled were 2-dimensional goals. The connections of 3-dimensional goal locations was facilitated using AGPRM [3], hence the solutions shown below has less number of 3-dimensional sampled landmarks.

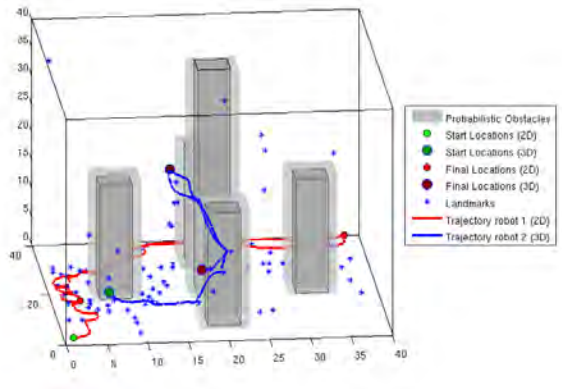
Figure 5 and Figure 6 shows the MAGPRM solution, for different sets of start and goal locations. The Dubins car covers all the 2-dimensional goal locations, while the simplified 3D helicopter robot covers only the 3-dimensional goal locations. Figure 5(a) shows the solution of MAGPRM

[10]A Dubins car and a simplified 3D robot

and Figure 5(b) shows the actual trajectories of the robots. A partitioning of space is again visible.



(a) MTSP solution for the robots



(b) Trajectories of the robots

Fig. 5. MAGPRM with Dubins' Car and 3D Vehicle with 5-Obstacles : Case 1

The heterogeneous agents case discussed in this section depicts the power of MAGPRM. The stochastic decision making problem in presence of heterogeneous agents, for which the computation of the cost of transitions are different, for each agent type, is solved. In order to solve the heterogeneous agents problem, the MAGPRM utilizes multiple GPRMs constructed on the underlying state-space of each type of agent.

VII. Conclusion

In this paper, we have presented a solution to the motion planning problem under uncertainty for multiple agents. In order to solve the overall problem, in conjunction with our solution methodology for a single agent (GPRM), a *routing problem* needs to be solved. The *routing problem* is solved using an existing solution to the *multiple traveling salesman problem*. The MTSP solution methodology, in conjunction with GPRM, results in the MAGPRM algorithm that solves the motion planning problem for multiple agents in presence of process uncertainty, and stochastic maps. Numerical experiments were performed on sets of *homogeneous*, and *heterogeneous agents*, for maps of different difficulty levels,

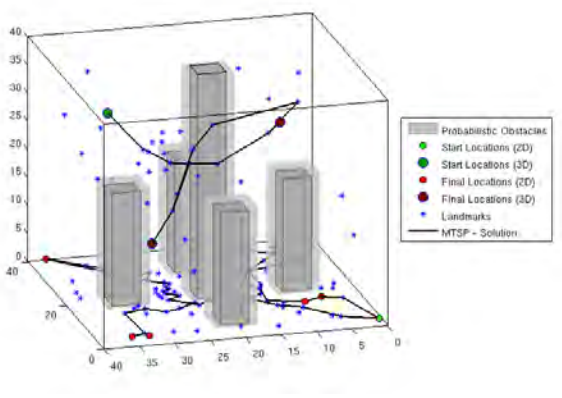


Fig. 6. MAGPRM with Dubins' Car and 3D Vehicle with 5-Obstacles : Case 2

with different number of start and goal locations, and different number of agents. Results show that the algorithm is indeed capable of solving the motion planning problem for multiple agents, with non-trivial nonlinear dynamics, in the presence of process uncertainty and stochastic maps.

References

[1] S. Chakravorty and S. Kumar. Generalized sampling based motion planners with application to nonholonomic systems. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 4077–4082. IEEE, 2009.

[2] S. Chakravorty and S. Kumar. Generalized sampling-based motion planners. *IEEE Transactions on Systems, Man, and CybernaticsPart B: Cybernetics*, 41(3):855, 2011.

[3] S. Kumar and S. Chakravorty. Adaptive sampling for generalized probabilistic roadmaps. *Journal of Control Theory and Applications*, 10(1):1–10, 2012.

[4] J.R. Kok. *Coordination and learning in cooperative multiagent systems*. PhD thesis, Dept. Comput. Sci., Univ. of Amsterdam, Amsterdam, The Netherlands, 2006.

[5] E. Frazzoli. Maneuver-based motion planning and coordination for multiple uavs. In *Digital Avionics Systems Conference, 2002. Proceedings. The 21st*, volume 2, pages 8D3–1. IEEE, 2002.

[6] Q. Li and S. Payandeh. Multi-agent cooperative manipulation with uncertainty: a neural net-based game theoretic approach. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 3, pages 3607–3612. IEEE, 2003.

[7] A. Undurti and J.P. How. A decentralized approach to multi-agent planning in the presence of constraints and uncertainty. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2534–2539. IEEE, 2011.

[8] W. Dong and J.A. Farrell. Decentralized cooperative control of multiple nonholonomic dynamic systems with uncertainty. *Automatica*, 45(3):706–710, 2009.

[9] V.R. Desaraju. *Decentralized Path Planning for Multiple Agents in Complex Environments Using Rapidly-exploring Random Trees*. PhD thesis, Massachusetts Institute of Technology, Dept. of Aeronautics and Astronautics, 2010.

[10] JA Cobano, R. Conde, D. Alejo, and A. Ollero. Path planning based on genetic algorithms and the monte-carlo method to avoid aerial vehicle collisions under uncertainties. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4429–4434. IEEE, 2011.

[11] M. Defoort, A. Doniec, and N. Bouraqadi. Decentralized robust collision avoidance based on receding horizon planning and potential field for multi-robots systems. *Informatics in Control Automation and Robotics*, pages 201–215, 2011.

[12] L.E. Parker. Path planning and motion coordination in multiple mobile robot teams. *Encyclopedia of complexity and system science*, pages 1–24, 2009.

[13] S. Bhattacharya, M. Likhachev, and V. Kumar. Multi-agent path planning with multiple tasks and distance constraints. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 953–959. IEEE, 2010.

[14] Y. Li, K. Gupta, and S. Payandeh. Motion planning of multiple agents in virtual environments using coordination graphs. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 378–383. IEEE, 2005.

[15] D.V. Dimarogonas, S.G. Loizou, K.J. Kyriakopoulos, and M.M. Zavlanos. Decentralized feedback stabilization and collision avoidance of multiple agents. *NTUA, http://users. ntua. gr/ddimar/TechRep0401. pdf, Tech. Report*, 2004.

[16] J.R. Kok and N. Vlassis. Collaborative multiagent reinforcement learning by payoff propagation. *The Journal of Machine Learning Research*, 7:1789–1828, 2006.

[17] A. Kulkarni and K. Tai. Probability collectives: a decentralized, distributed optimization for multi-agent systems. *Applications of Soft Computing*, pages 441–450, 2009.

[18] P. Oberlin, S. Rathinam, and S. Darbha. Today's Traveling Salesman Problem. *Robotics & Automation Magazine, IEEE*, 17(4):70–77, 2010.

[19] LE Kavraki, P. Svestka, J.C. Latombe, and MH Overmars. Probabilistic roadmaps for path planning in high-dimensionalconfiguration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[20] S. Kumar and S. Chakravorty. Adaptive sampling for generalized sampling based motion planners. In *49th IEEE Conference on Decision and Control*, pages 7688–7693. IEEE, 2010.

[21] C. Boutilier. Planning, learning and coordination in multiagent decision processes. In *Proc. 6th conference on Theoretical aspects of rationality and knowledge*, pages 195–210. Morgan Kaufmann Publishers Inc., 1996.

[22] C.E. Guestrin. *Planning under uncertainty in complex structured environments*. PhD thesis, Dept. Comput. Sci., Stanford Univ., Stanford, CA, 2003.

[23] J. Kok and N. Vlassis. Using the max-plus algorithm for multiagent decision making in coordination graphs. *RoboCup 2005: Robot Soccer World Cup IX*, pages 1–12, 2006.

[24] K. Helsgaun. An effective implementation of the lin-kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130, 2000.

[25] T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219, 2006.

[26] C.E. Noon and J.C. Bean. An efficient transformation of the generalized traveling salesman problem. *Ann Arbor*, 1001:48109–2117, 1989.

[27] G. Oriolo, A. De Luca, and M. Vendittelli. WMR control via dynamic feedback linearization: design, implementation, and experimental validation. *IEEE Transactions on Control Systems Technology*, 10(6):835–852, 2002.