

---

**COOPERATIVE CONTROL OF MULTIPLE  
SPACECRAFT SUBJECT TO MEASUREMENT  
UNCERTAINTIES AND TIME DELAYS**

**Kamesh Subbarao**

**University of Texas at Arlington  
Office of Research Administration  
400 S Corn Street  
Arlington, TX 76019-0001**

**12 July 2019**

**Final Report**

**APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED.**



**AIR FORCE RESEARCH LABORATORY  
Space Vehicles Directorate  
3550 Aberdeen Ave SE  
AIR FORCE MATERIEL COMMAND  
KIRTLAND AIR FORCE BASE, NM 87117-5776**

---

## NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research which is exempt from public affairs security and policy review in accordance with AFI 61-201, paragraph 2.3.5.1. This report is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (<http://www.dtic.mil>).

AFRL-RV-PS-TR-2019-0061 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

//signed//

---

CHRISTOPHER PETERSEN  
Program Manager

//signed//

---

DAVID WILT  
Tech Advisor, Space Component Technology  
Branch

//signed//

---

JOHN BEAUCHEMIN  
Chief Engineer, Spacecraft Technology Division  
Space Vehicles Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

Approved for public release; distribution is unlimited.

**REPORT DOCUMENTATION PAGE**Form Approved  
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

<b>1. REPORT DATE (DD-MM-YYYY)</b> 12-07-2019		<b>2. REPORT TYPE</b> Final Report		<b>3. DATES COVERED (From - To)</b> 14 Apr 2016 – 12 Jul 2019	
<b>4. TITLE AND SUBTITLE</b> Cooperative Control of Multiple Spacecraft Subject to Measurement Uncertainties and Time Delays				<b>5a. CONTRACT NUMBER</b> FA9453-16-1-0058	
				<b>5b. GRANT NUMBER</b>	
				<b>5c. PROGRAM ELEMENT NUMBER</b> 62601F	
<b>6. AUTHOR(S)</b> Kamesh Subbarao				<b>5d. PROJECT NUMBER</b> 8809	
				<b>5e. TASK NUMBER</b> PPM00019650	
				<b>5f. WORK UNIT NUMBER</b> EF127462	
<b>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</b> AND ADDRESS(ES) University of Texas at Arlington Office of Research Administration 400 S Corn Street Arlington, TX 76019-0001				<b>8. PERFORMING ORGANIZATION REPORT NUMBER</b>	
<b>9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)</b> Air Force Research Laboratory Space Vehicles Directorate 3550 Aberdeen Ave., SE Kirtland AFB, NM 87117-5776				<b>10. SPONSOR/MONITOR'S ACRONYM(S)</b> AFRL/RVSV	
				<b>11. SPONSOR/MONITOR'S REPORT NUMBER(S)</b> AFRL-RV-PS-TR-2019-0061	
<b>12. DISTRIBUTION / AVAILABILITY STATEMENT</b> Approved for public release; distribution is unlimited.					
<b>13. SUPPLEMENTARY NOTES</b>					
<b>14. ABSTRACT</b> The purpose of the project was to develop cooperative guidance, and control laws for multiple vehicles subject to communication delays. Potential function-based methods were studied for path planning and trajectory synthesis and extended to formulate the cooperation problem. Lyapunov based control design methods and nonlinear model predictive control were studied for the cooperative control problem. The guidance, and control algorithms were implemented on wheeled mobile robots in a Cyber-Physical Systems setup to study the effect of time delays in communicating with the platforms and receiving communication from it. The performance of the algorithms was demonstrated on the experimental test platform containing three wheeled robots.					
<b>15. SUBJECT TERMS</b> Constrained Dynamics Formulation, Trajectory Synthesis, Demonstration of Performance, Ground Vehicle Platforms, Dynamical Equations					
<b>16. SECURITY CLASSIFICATION OF:</b>			<b>17. LIMITATION OF ABSTRACT</b>	<b>18. NUMBER OF PAGES</b>	<b>19a. NAME OF RESPONSIBLE PERSON</b>
<b>a. REPORT</b> Unclassified	<b>b. ABSTRACT</b> Unclassified	<b>c. THIS PAGE</b> Unclassified			Christopher Petersen
			Unlimited	92	<b>19b. TELEPHONE NUMBER (include area code)</b>

Standard Form 298 (Rev. 8-98)  
Prescribed by ANSI Std. Z39.18

(This page intentionally left blank)

## TABLE OF CONTENTS

Section	Page
<b>LIST OF FIGURES</b> .....	iii
<b>1.0 SUMMARY</b> .....	1
<b>2.0 INTRODUCTION</b> .....	2
<b>3.0 BACKGROUND, FRAMEWORK, PROBLEM DESCRIPTION</b> .....	4
3.1 Background.....	5
3.1.1 Path Planning with Control Effort and Navigation Functions.....	5
3.1.2 Nonlinear Guidance Law.....	8
3.1.2.1 Nonlinear Model Predictive Control-Based Guidance.....	8
3.1.2.2 Nonlinear Control of Rover Vehicles.....	10
3.1.3 Cooperative Control with Artificial Potential Functions .....	11
3.2 Objectives and Contributions.....	13
3.2.1 List of Contributions .....	13
3.2.2 List of Published Works .....	13
<b>4.0 METHODS, ASSUMPTIONS, AND PROCEDURES</b> .....	14
4.1 Path Planning using Minimum Control Effort based Navigation Functions .....	14
4.1.1 Initialization .....	15
4.1.2 Minimum Control Effort (MCE) Based on Linear Rover Model.....	16
4.1.3 Inverse Dynamics (ID) Based Control Effort For a Nonlinear Rover Model.....	18
4.1.4 Navigation Function Generation.....	20
4.1.5 Trajectory Generation .....	22
4.2 Nonlinear Lyapunov Based (Backstepping-Like) Control Design.....	23
4.3 Nonlinear Model Predictive Control.....	26
4.3.1 Input and State Constraints .....	30
4.3.2 Guidance Command Synthesis Using the Linear Matrix Inequality (LMI) Form.....	31
4.4 Cooperative Control of Multiple Vehicles.....	33
4.4.1 Artificial Potential Function for ‘Conflict-Free’ Trajectory Synthesis.....	33
4.4.2 Swarm Aggregation APF Design.....	34
4.4.3 Minimum Control Effort Navigation Function for Social Foraging.....	35
4.4.4 Central Difference Approximation to NF Gradient .....	36
4.4.5 APF for Collision Avoidance.....	37
4.4.6 Control Law Design for Cooperation .....	37
4.4.7 Inter-Vehicle Communication .....	39
4.5 Cooperative Control in the Presence of Measurement Time-Delays .....	39
4.5.1 Modified Rodrigues Parameters .....	40
4.5.2 Feedback Linearization of the Attitude Dynamics .....	41
4.5.3 Consensus of Formation .....	41
4.5.4 Consensus Control Law .....	42
4.5.5 Stability Analysis .....	43
4.5.5.1 Time Domain Approach.....	43
4.5.5.2 Frequency Domain Approach .....	45

<b>5.0 RESULTS AND DISCUSSION</b> .....	48
5.1 Numerical Simulations: Guidance to a Reachable State – Navigation Function.....	48
5.2 Numerical Simulations: Nonlinear MPC vs Nonlinear Backstepping like guidance law.....	53
5.3 Numerical Simulations: Cooperative Control of Multiple Vehicles (No Time Delays)	61
5.4 Numerical Simulations: Cooperative Control of Multiple Vehicles (With Time Delays) .....	64
5.5 Experimental Results: Cooperative Control of Multiple Vehicles (With Time Delays)	67
<b>6.0 CONCLUSIONS</b> .....	75
<b>REFERENCES</b> .....	76
<b>LIST OF SYMBOLS, ABBREVIATIONS, AND ACRONYMS</b> .....	82

## LIST OF FIGURES

Figure 1: Removed for administrative purposes	5
Figure 2: ASL Gremlin Mobile Robot.....	5
Figure 3: Local minimum example with traditional APF.....	6
Figure 4: Illustration of MPC control horizon.....	9
Figure 5: Control Effort Based Navigation Function Algorithm.....	15
Figure 6: Grid set up in two dimensional space for the minimum control effort.....	17
Figure 7: Grid set up in two dimensional space for the inverse dynamics based path plan problem.....	19
Figure 8: Block diagram of the information flow of the framework.....	23
Figure 9: Kinematics of the rover vehicle.....	24
Figure 10: Communication protocol.....	39
Figure 11: Feedback System.....	46
Figure 12: No Obstacle: Min. control effort based path plan.....	48
Figure 13: No Obstacle: Inverse Dynamics based path plan.....	49
Figure 14: With Obstacle: Min. control effort based path plan.....	50
Figure 15: With Obstacle: Inverse Dynamics based path plan.....	51
Figure 16: No Obstacle: Tracking performance of MCE Plan.....	52
Figure 17: No Obstacle: Tracking Performance Inverse Dynamics.....	52
Figure 18: With Obstacle: MCE Path Plan.....	52
Figure 19: With Obstacle: Tracking performance of the Inverse Dynamics.....	53
Figure 20: Reference path tracking for case 1.....	54
Figure 21: x and y position tracking for case 1.....	55
Figure 22: Heading angle plot for case 1.....	55
Figure 23: Velocity plot for case 1.....	56
Figure 24: Guidance command $u_1$ for case 1.....	56
Figure 25: Guidance command $u_2$ for case 1.....	56
Figure 26: Reference path tracking for case 2.....	58
Figure 27: x and y position tracking for case 2.....	58
Figure 28: Heading angle plot for case 2.....	59
Figure 29: Velocity plot for case 2.....	59
Figure 30: Guidance command $u_1$ for case 2.....	60
Figure 31: Guidance command $u_2$ for case 2.....	60
Figure 32: Example 1 with swarm aggregation policy - Position, Heading.....	62
Figure 33: Example 1 with social foraging - Position, Heading.....	62
Figure 34: Example 2 environment and navigation function potential field.....	63
Figure 35: Example 2 with social foraging policy - Position, Heading.....	63
Figure 36: Example 3 environment and navigation function potential field.....	64
Figure 37: Example 3 with social foraging policy - Position, Heading.....	64
Figure 38: Communication Graph.....	65
Figure 39: Spacecraft Attitude vs Time.....	66
Figure 40: Spacecraft Attitude Rates vs Time.....	66
Figure 41: Spacecraft Angular Velocities vs Time.....	67
Figure 42: Rover built by the Aerospace Systems Lab (ASL) at UT Arlington.....	67

Figure 43: Top-View of the ASL-Gremlin-Rover with all the components.....	68
Figure 44: Illustration of the system architecture making up the CPS of the experimental setup.	69
Figure 45: Relationship between angular velocity of wheel and motor PWM.....	70
Figure 46: Closed loop control system with communication delay.....	71
Figure 47: System stability analysis with different combination of delays .....	71
Figure 48: Flow of information within the GNC framework for the mobile robot platform.....	72
Figure 49: Flow of information for cooperative control experiments .....	73
Figure 50: Setup for cooperative experiment.....	73
Figure 51: Robot positions for cooperative experiment .....	74
Figure 52: Wheel speed commands for cooperative experiment.....	74
Figure 53: Relative distance between Robots for cooperative experiment.....	75



## 1.0 SUMMARY

The objective of the project was to conduct research into the topic of “*Cooperative Control of Multiple Spacecraft subject to Measurement Uncertainties and Time Delays*”. The research focused on both translational and rotational motion control using dynamic graph theory-based approaches, artificial potential functions to create virtual potential fields and navigation functions, and Lyapunov based control techniques. The framework enforced cooperation among multiple vehicles to synthesize consensus based constrained trajectories for operation, using a combined objective function that included cooperation, collision avoidance, and goal attainment. A nonlinear model predictive control strategy was explored to synthesize guidance laws to steer a group of vehicles towards a goal or to attain a formation. Additionally, robust Lyapunov based control laws were synthesized for comparison, and to accommodate uncertainties in measurements including time delays in the communication links. The multiple spacecraft control problem was reduced to a plane recognizing that upon utilizing a suitable control technique, the dynamics of each spacecraft can be reduced, and is similar to the one typically utilized in studying cooperative control problems. The problems solved included cooperative trajectory tracking, and attitude synchronization. The framework, and algorithms were verified through extensive numerical simulations. An experimental test bed that included two ground rovers, built and developed at the Aerospace Systems Laboratory served as systems upon which the framework, and the algorithms were validated. The architecture utilized a Cyber Physical Systems (CPS) approach wherein sensor data was sent to a remote location which computed the control commands and then communicated it to the operating vehicles over 4G LTE and WiFi. The Robot Operating System (ROS) served to integrate the vehicles on a desktop and characterize the effects of time delays on the performance of the vehicles, and also to verify the delay robustness as well as measurement imperfections. The purpose of this research was to enable smart deployment (and tasking) of space assets in applications such as space surveillance, and monitoring of space objects. The framework, algorithms, and results from this research were shown to be applicable in diverse applications such as, formation flight of multiple aircraft, cooperative target tracking, cooperative ground and air vehicle control for disaster site monitoring, accident investigation, traffic monitoring, automatic platooning in future highways such as those envisioned in the Smart Cities program.

## 2.0 INTRODUCTION

Consensus seeking formation of vehicles implies disciplined motion of several rigid vehicles (e.g. spacecraft, unmanned aircraft, robotic vehicles etc.) maintaining a desired geometric shape and geometry. The applications of formations are numerous as with spacecraft for unprecedented image resolution in astronomy and surveillance [1, 2], modeling of environment, surveillance and rescue missions in military applications, monitoring of forests and agricultural lands, health-care applications, collaborative information processing, energy saving from vortex forces [3] and fuel efficiency via induced drag reduction [4]. Spacecraft formations have been envisioned for distributed sensing for gravitational field mapping, atmospheric data sampling, co-observations (i.e., near-simultaneous observations of the same science target by instruments on multiple platforms), and synthetic radio-frequency and radar apertures. Formation flying also applies to airborne refueling, and quick deployment of troops and vehicles using several aircraft. Ref. [5] summarizes some compelling examples of formations of a number of small, low cost structures instead of one big instrument.

Formations can be established in several ways such as, leader-follower, behavioral methods, and virtual structures [1, 6]. In the proposed work we deal with position tracking, attitude (orientation) synchronization, and tracking of a group of space vehicles under communication constraints which helps in maintaining the formation geometry. We develop cooperative control [1, 3, 4, 7] laws using the theory of constraint forces to build formations from arbitrary initial conditions of the rigid bodies in the group. This is accomplished by developing general non-linear governing equations of motion for a group of rigid bodies [8,9] using the extended Euler-Lagrange Method [1, 2, 10]. The formation shape (geometry) is maintained due to the constraints acting between the rigid bodies which in the present context serves like information exchange among the individual units. Communication [11] between these units helps in maintaining the formation; loss (deterioration) of which would make the overall formation unstable and would annihilate the structure as a consequence. In the present context, we will focus on time delays (constant as well as time varying) to model ‘deteriorated (lossy)’ communication. Graph theory has been shown to be very useful for modeling communication between cooperating agents in recent years. Information consensus strategies based on graph theory for multiple vehicle control has been extensively addressed in [12, 13]. Active constraints and forces ([13, 14]) help in the local interactions between the units of this system and can be used to determine the total force required on each rigid body to maintain the formation. To keep these intact, robust nonlinear control laws will be derived to enforce a Baumgarte [1] like stabilization procedure for constrained dynamical systems. Similar work was done in [1, 7, 10] wherein constraints are stabilized via a proportional-derivative structure.

Among other approaches to enforce coordination (coordination control of multiple mobile robots [15]), we will also investigate the potential field approach to achieve coordination of multiple vehicles and shape the dynamics of the formation. Coordinated control using potential functions can be found in [16-20]. The essential idea is to create an energy-like function (potential function) to enforce position constraints between vehicles and use the negative gradient of the potential function as a restoring force on each vehicle to achieve coordination. The constrained dynamical approach in this proposed work utilizes this basic idea to set up the control laws.

For a large interconnected system with arbitrary connection topologies, the synthesis of a constraint potential is non-trivial. We solve this by exploiting the connection topology and synthesizing a potential energy function based on the graph Laplacian for this interconnected system. Once chosen, this still poses an additional difficulty of synthesizing the constraint Jacobians as analytical solution of the Jacobian each time for a different connection topology is not possible. We address this issue by numerically synthesizing the Jacobian. Thus, the framework only needs to know the connection topology and the control laws are derived accordingly. In the present research, several candidate scenarios will be evaluated such as tracking and finally unaided consensus. For detailed descriptions of these scenarios also see [12, 21, 22].

In this project, all the above concepts are applied to mobile robot vehicles in simulation. Additionally, some of the concepts, such as the nonlinear guidance laws and cooperative control framework are validated experimentally using mobile robot testing platforms in the Aerospace Systems Laboratory at The University of Texas at Arlington.

The following goals were laid out for Year 1 of grant period (April 13, 2016 - April 12, 2017).

[BA1]

- Develop representative governing equations of motion for single space vehicle.
- Develop constrained dynamics formulation for multiple vehicles (Completed)
- Architect the object-oriented multiple space vehicles simulation framework (Completed)
- Submit article for publication/presentation at an AIAA/AAS conference (Completed)

[BA2]

- Continue development of the constrained dynamics formulation.
- Develop potential functions for path planning and trajectory synthesis in the absence of uncertainties. (Completed)
- Formulate the cooperation problem. (Completed)
- Begin preparing unmanned ground vehicle platform. (Completed)
- Submit article for publication/presentation at an AIAA/AAS conference (Completed)

The following goals were laid out for Year 2 of grant period (April 13, 2017 - April 12, 2018).

[BA3]

- Continue development of the constrained dynamics formulation.
- Develop potential functions for path planning and trajectory synthesis in the absence of uncertainties. (Completed)
- Formulate the cooperation problem for multiple vehicle aggregation and social foraging (Completed)
- Maintain and update unmanned ground vehicle platforms. (Completed)
- Submit articles for publication in peer reviewed journals (Journal of Astronautical Sciences and Journal of Robotics and Autonomous Systems). (Completed)

[BA4]

- Formulate dynamic graph networks - time varying connection topologies. (Completed)
- Study cooperation problem, implement in simulation, and obtain preliminary results. (Completed)
- Implement preliminary algorithm elements on ground vehicle platforms (2 vehicles). (Completed)
- Prepare and submit paper for presentation at AIAA/AAS conference. (Completed)

The following goals were laid out for Year 3 of grant period (April 13, 2018 - April 12, 2019).

[BA5]

- Develop robust control laws for output delayed cooperative control. (Completed)
- Study performance of control laws for different delay cases. (Completed)
- Conduct detailed simulation studies. (Completed)
- Prepare and submit paper for presentation at AIAA/AAS conference. (Completed)

[BA6]

- Characterize performance of cooperative controller for all the cases discussed. (Completed)
- Test algorithm elements on experimental platforms. (Completed)
- Prepare paper for presentation at AIAA/AAS conference. (Completed)
- Prepare and submit final report. (Completed)

### **3.0 BACKGROUND, FRAMEWORK, AND PROBLEM DESCRIPTION**

The focus of this research is to present a framework for path planning and guidance for co-operating autonomous vehicles. The quantifiable measures of autonomy recognized in this work are the vehicle's ability to observe, orient, make decisions, and act [23]. Of those autonomy measures, this research focused primarily on expanding a vehicle's ability to make decisions and to control its actions within a given environment.

There is an increasing need for reliable path planning and guidance algorithms for autonomous vehicles [24–28]. The need for guidance, in the form of path planning, for arises from the kinds of environments the vehicles may encounter. The vehicle must be able to account for these details and suggest safe paths for it to follow. To help the vehicles make decisions and find safe paths in the environment, a numerical navigation function algorithm is presented in this research. Another important issue is, how to follow the safe paths which have been generated? The vehicle must have the capability to arrive at its destination safely in the presence of uncertainty in the environment or its own physical limitations. The ability to act within the vehicle's confines is of particular interest in this research and is addressed in the form of an improved nonlinear model predictive control derivation. Additionally, the ability to have guidance in how to act when multiple vehicles are acting in the same environment is of interest and is addressed through a cooperative control policy. The components of the framework are verified in simulation and extended to real-time testing platforms, such as the vehicle in Figure 2, to provide experimental validation.

### 3.1 Background

#### 3.1.1 Path Planning with Control Effort and Navigation Functions

The first component of the framework presented in this research is a path planning algorithm to help a vehicle make decisions in finding safe reference paths through the environment. In general, path planning is the process of finding a safe path between two points for a vehicle to travel. There are a variety of methods for path planning found in textbooks and papers alike. Many of the path planning methods currently being researched consider collision and obstacle avoidance as a primary objective. And while obstacle avoidance is considered in this research, its novel contribution is an investigation into how to include control information into the path plan through a grid-based numerical potential field construction.



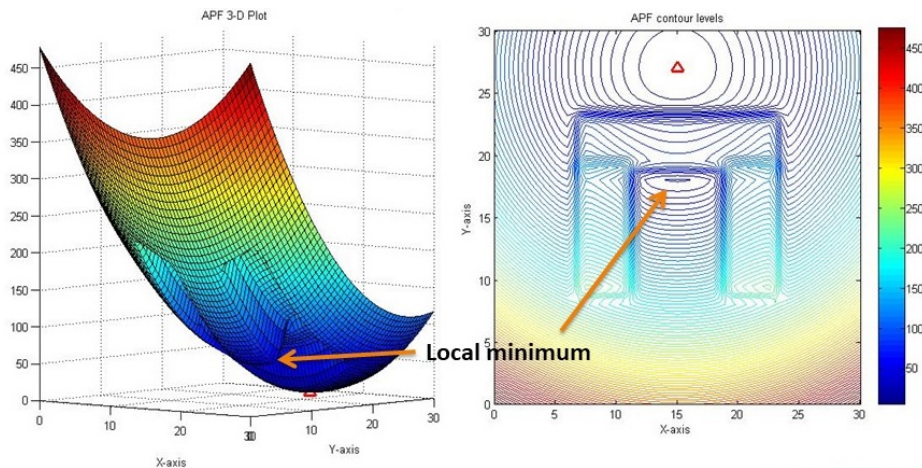
**Figure 2: ASL Gremlin Mobile Robot**

Other path planning techniques that make use of the control effort and the kinematic model of the system are discussed in terms of rapidly exploring randomized trees (rrt and rrt\*) in [29, 30], kinodynamic rrt in [31], and probabilistic roadmap approach in [32]. These methods use a randomized approach with state information of the system to determine the path plans, and the paths are designed with information of the environment and the initial location. While these methods can be computationally intensive, depending on the model, they also generate plans without needing to know where the objective is located. The path planning methods discussed in references [29–32] can incorporate knowledge of a system’s control effort to generate a path plan in a randomized sampling-based manner. In contrast, the methodology discussed with this work makes use of a special class of artificial potential functions called navigation functions. This construction method will make use of the control effort of the system and enable a path plan to be formed from almost any point in the environment.

In general, potential fields for path planning purposes are generated such that their structure, potential levels and shapes, can intuitively reflect the virtual makeup of the workspace. And a vehicle within a potential field is treated as a particle under the influence of gravity. The paths with these methods are then generated through a method that is similar to a steepest descent

optimization problem [33–36]. Traditional potential field methods were first designed as an online collision avoidance scheme in which the attractive potential is represented by a parabolic well at the goal and the repulsive potentials are defined in the constrained space which will tend towards infinity for the points in their vicinity [33]. Then, a path, as well as the control input, can be found by following the negative gradient of this function (similar to steepest descent). Although traditional Artificial Potential Field (APF) methods can effectively create collision-free paths, there is one possible drawback. Since the method by which the paths are generated is similar to a steepest descent problem, the paths derived for the system could reach equilibrium at a configuration that is not its goal. This is known as the local minimum problem [34]. An example of the local minimum problem arises when considering the dynamical system at a configuration where the attractive potential from the goal is equal to the repulsive potential of the constrained space. An illustrative example of a potential field with a local minimum and a U-shaped obstacle using the traditional APF method is shown in Figure 3. It is evident from Figure 3 that not every path plan following the negative gradient of the potential field will arrive at the goal. The local minima problem with potential field methods is what gave rise to the development of navigation functions, which possess only a global minimum in its potential field that is located at the goal [34–36].

There are two main approaches to create a navigation function. The first method defines an analytic function which possesses an attractive component associated with the objective and repulsive components attached to the obstacles. This is motivated by the research introduced in [33] for online collision avoidance using artificial potential functions. Examples of the analytical approach with a NF are presented in [37–39] whose results originate from the NF definition introduced in [36]. With this approach, the navigation function is defined as a composition of several functions each designed to satisfy specific properties established in [36]. The specific properties established to define a navigation function analytically are summarized as:



**Figure 3: Local minimum example with traditional APF**

- The function is continuous and differentiable (smooth function) on the path connected set to the goal
- The function is uniformly maximal on the boundaries
- The function must have a unique and global minimum at the goal
- The function must be a Morse function

And while effective, this method requires proper tuning of several parameters within the function before the local minima can be removed and for the NF to be properly defined.

The second method for constructing a viable navigation function is to construct it numerically in a discrete grid. This can be done either by using numerical solutions to partial differential equations (PDE) as seen in references [40–42] or by assigning potentials to a discrete workspace based on their distance from the objective [34, 35, 33–36]. The numerical navigation functions described in references [41] and [42] use harmonic functions to represent the workspace with the boundary conditions enforced to ensure that a viable path is found. The navigation function in [40] is constructed similarly in that it uses the finite difference method to solve a PDE representation of the Hamilton-Jacobi-Bellman (HJB) equation over the workspace. Additionally, the method described in reference [18] can be made to rely upon the system’s dynamics. Numerical potential functions, such as those described in references [34, 35, 43–46], have an advantage of being constructed in such a way that the goal location is given the minimum value and the rest of the potential values are propagated through-out the remainder of the free operating space. These approaches are done through wave-front expansion with counting and logic involved. Navigation function path planners are effective in generating safe paths to the goal, however the methods that have been introduced in the past are primarily formed only with knowledge of the distance to the goal and the connectivity of the free regions in the workspace. Conversely, the novel algorithm introduced in this project will generate the navigation function based on the control effort of a given model to go from one grid point to another in a given environment. The novel algorithm will leverage the construction method for the navigation functions described in references [34] and [35] but will use the system’s control effort to determine the contour levels.

The path plan algorithm introduced in this project can generate reference paths from anywhere in the free environment. However, the algorithm will require knowledge of both the objective location as well as a final desired state and would need to be regenerated if new information is acquired. The result, however, is a path plan that will guide the vehicle safely to its objective while also considering the control effort to get there as well as how to form its approach to a goal state. The planner in this research considers the model of the vehicle to consist of four state variables, but it generates the path plan within a two-dimensional environment making it computationally cheap. While the navigation function path planner is able to generate path plan from anywhere in the free space, the fact that it will need to be regenerated whenever new knowledge is obtained can make it inefficient in the presence of uncertainty. To overcome this issue, extensions of the path planning algorithm are presented using modified versions of RRT\* and D\* with a minimum control effort-based metric to determine the cost to move between configurations.

The RRT\* algorithm, as presented in reference [30] is designed to form its tree from the starting configuration of the robot and can use the tree to find a minimum cost traversal to anywhere in the free space. This ability gives the planner the ability to determine a path with an uncertain goal and eliminates the dependence on grid resolution. The D\* algorithm is another grid-based path planner and is discussed in references [47, 48]. This algorithm is designed as a dynamic A\* algorithm, discussed in references [49] and [50], where it has the ability to dynamically re-plan a path in the presence of an uncertain environment. These algorithms are chosen due to their ability to account for uncertain objectives, terrain types and can be used to find optimal traverses.

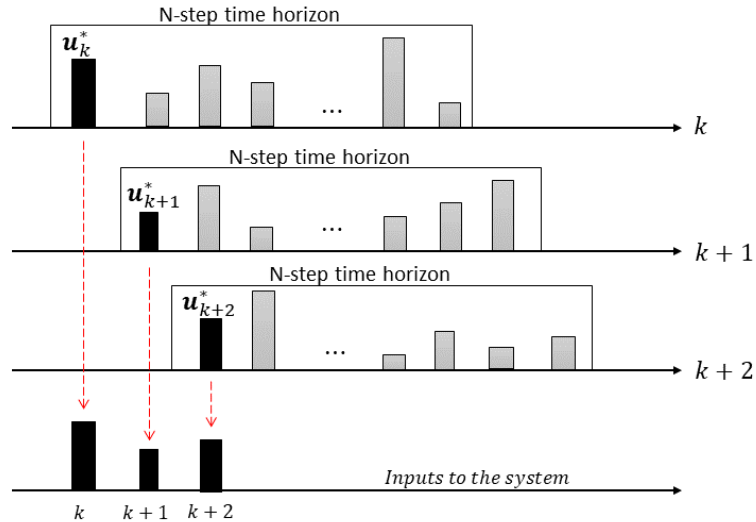
### 3.1.2 Nonlinear Guidance Law

The next component of the framework applied in this research is a nonlinear guidance technique. This part of the research is intended to provide vehicle with an increased ability to act in the given scenarios. The guidance laws are designed to provide commands to ensure that the vehicle can track the reference paths it is given.

#### 3.1.2.1 Nonlinear Model Predictive Control-Based Guidance

The main nonlinear guidance technique that is applied within this framework is based on nonlinear model predictive control (NMPC). The NMPC algorithm is chosen due to its ability to incorporate constraints on the inputs and outputs of the system being studied. Additionally, the NMPC algorithm is solved by making use of the State Dependent Riccati Equation (SDRE) and its associated state-dependent coefficient formulation. The use of model predictive control, or receding horizon control (RHC), is a widely researched topic in controls engineering for a variety of applications. Model predictive control is a process control method where the current inputs to the system are determined by forecasting the behavior of the system model over a finite horizon. The control is designed to minimize a cost function over the finite horizon and can be used for regulation or tracking of a reference trajectory [51]. For MPC to be implemented, a continuous system is discretized given a sampling time based on the process being studied. Then, a prediction horizon is taken as the number of time steps into the future being considered in the forecast. This allows for a control input to be determined at each time step over the horizon. Although the model behavior is considered over the time horizon, only the first input is applied to the mobile robot using the technique in this research, as illustrated in Figure 4.





**Figure 4: Illustration of MPC control horizon**

This approach is used to generate high-level guidance commands for the vehicle to track a given trajectory. The derived guidance commands will be unique since it will make use of the state-dependent coefficient (SDC) form of the nonlinear kinematic equations and the system's inputs will be found by quadratic programming. The SDC formulation is used because it can preserve the nonlinear nature of the system being studied. The SDC form for nonlinear systems as it pertains to controller, observer or filter design is discussed in references [52–55]. The focus of using the SDC form is to transform a nonlinear system into a pseudo-linear form and then implement optimal control or estimation techniques through solving the SDRE. As it relates to control design, the SDC form allows for synthesis of nonlinear feedback controllers that are similar to the LQR structure. In references [52] and [53], the problems are posed to use the SDRE to solve problems such as the infinite horizon quadratic regulator for some example nonlinear dynamics. The authors use direct parameterization to place the dynamics into the SDC form, which they note is not unique for a given system.

The SDC form and control designs using the SDRE discussed in references [52–55] center on solutions over an infinite time horizon. In contrast, the results described in references [56] and [57] look at solutions for control problems with a finite horizon. Both papers include a change of variables to solve for the control over a finite horizon as well as finite terminal constraints, so that they can achieve their respective objectives [56,57]. The results presented in references [52–57] highlight the use of the SDC form and the nonlinear feedback control laws that can be found through solving the SDRE. These results, however, do not consider constraints on the system's inputs or outputs (other than some terminal constraints, shown in references [56] and [57]). The contribution of the work presented in this research will make use of an SDC formulation and nonlinear model predictive control, which will enforce input and output constraints while performing reference trajectory tracking. Traditionally, nonlinear model predictive control involves linearizing the system's dynamics about a nominal trajectory [51, 58, 59]. References [51] and [58] give an overview of the traditional formulation for NMPC. And reference [60]

discusses important results for MPC in general with considerations towards both stability and optimality of the results. An alternate approach for NMPC employs a Control Lyapunov Function (CLF) to help with achieving stability and applying an approximation to the terminal cost to the tail of the infinite horizon problem [61–66]. The CLF approach for receding horizon control along with an SDC factorized system is shown in references [63] and [64], without considering constraints. Reference [61] presents the CLF technique in detail with receding horizon control and focuses on time varying and input constrained systems. Also, the author in [61] propose that finding an appropriate CLF is equivalent to finding a continuous stabilizing control law for the system. Then, the author in reference [40] further extends the discussion from [61] covering the CLF approach with RHC. The developments found in [62] pertain to the stabilization of unconstrained nonlinear systems and use the CLF as a terminal cost function. Furthermore, the authors conclude that there is no need for constraints on the system or the CLF to achieve stability, with the proposed methodology.

The contribution of the work covered with this research is motivated by the developments in reference [67]. However, the research presented in this project distinguishes itself in that the NMPC guidance design as applied to the vehicle is verified both in simulations and through experiments by applying the design to a real-time mobile robot testing platform.

### 3.1.2.2 Nonlinear Control of Rover Vehicles

Another nonlinear guidance method that will be derived and used in the framework is a backstepping-like guidance law. This design is provably stable and guarantees bounded trajectory tracking errors. This will ensure that the mobile robot can safely and accurately follow a path plan generated by the navigation function. Also, this method is widely-researched and can be used for comparison with the NMPC design. There have been implementations of different nonlinear control designs in a wide variety of mobile robot applications [68–77]. Some of the applications of non-linear control with wheeled robots have only been verified through simulation, as in references [68–74], while others have been validated by hardware experiments such as references [75–77]. Reference [78] provides a control design based on a virtual structure approach to follow a simple user defined trajectory. And in reference [69], an adaptive control design is implemented in simulation to control a mobile robot where it is also proven to be robust to input saturation and disturbances.

The experimental results presented in reference [75] validates a learning based nonlinear model predictive control design that is constructed so that it must learn a path through repetition to improve its model parameters. The experimental results discussed in reference [76] are for an adaptive dynamic control design that applies its control inputs to the dynamics in order to govern the kinematics of a wheeled robot. The applications of this work are for an autonomous load carrying wheeled mobile robot in an industrial setting [76]. Also, in reference [77] the results show a velocity scheduling-based controller that utilizes dynamic feedback linearization of a mobile robot with only two out of four wheels being actuated.

Backstepping control designs for wheeled mobile robots are discussed in references [70–74]. The control laws in references [70] and [71] apply backstepping to control the kinematic model directly from the dynamics of the system and torques on the vehicle are applied as the control

inputs. In contrast, the backstepping control design in references [72,73] account for commanding the robot's heading angle turn rate and its forward acceleration. Also, a backstepping-like control design is proposed in reference [74], which is derived so that the control inputs are given in terms of the wheel speeds of a robot in order to facilitate its implementation on an experimental testbed.

The backstepping-like guidance law discussed in this framework will be motivated by the developments presented in references [72, 73]. The results from [72, 73] are expanded upon in this dissertation with updated stability considerations and presentation of real-time implementation results using the derived guidance laws.

### 3.1.3 Cooperative Control with Artificial Potential Functions

It is possible that more than one vehicle may be present in a given scenario and they will need to collaborate with one another. Therefore, it is practical to consider a guidance methodology for scenarios involving multiple vehicles needing to cooperate. The chosen approach to investigate the interactions between the vehicles will be evaluated based on an artificial potential function (APF) approach. With this approach, the commanded velocity and heading angle guidance commands will be determined from a composite potential function. The composite potential function will consist of a numerical navigation function, an analytical potential function governing the interaction forces between the vehicles and an additional repulsive potential function. The numerical navigation function will represent the given environment, given obstacle locations and an objective gathering location. The analytical APF component will have attractive and repulsive characteristics to dictate the behavior of the individual agents within the group. Finally, the additional repulsive term is used as an extra layer to ensure the vehicles do not collide. In general, the goal of this cooperative control policy is to derive guidance commands that govern how individuals within a group move so that they stay together and/or avoid collisions. The set of rules called Reynold's Rules were defined in order to capture the collective motion of large groups based on observations in nature. Formally, Reynolds' rules for collective motion are [78]:

- Collision avoidance
- Velocity matching - matching speed and motion direction
- Flock centering

Traditional approaches for cooperative control frameworks involve portraying the communication topology of the group as a graph of nodes and edges, called the graph theoretic framework [56]. In this arrangement, the flow of information is structured within a communication graph. The resulting graph helps to illustrate which agents communicate with each other and which information is available to the group. In the graph theoretic framework, the feedback control laws are derived based on graph theory, involving the formation of an adjacency matrix, based on the information flow, and an in-degree matrix, based on the number of agents in communication with a particular node. Then, the feedback control law is found by evaluating the graph Laplacian, which is the difference between the in-degree matrix and the adjacency matrix, multiplied by the state of the group considering integrator dynamics [56]. An alternate approach to the graph theoretic framework for cooperative control involves defining potential energy functions, also called artificial potential functions (APF). The APF method was initially introduced as an online collision avoidance algorithm, as detailed in reference [33]. The design is such that it possesses an attractive

component as well as a repulsive component. The attractive component is designed to draw the system to a desired state and the repulsive component is designed to avoid potential hazards or undesired states for the system [33]. For cooperative control scenarios, however, the APF is used to influence the behavior of the group in a decentralized approach. Thus, the individual vehicles act in accordance with their respective locations relative to the other vehicles present. There are some defined objectives in the literature for cooperative control using APFs, which are similar to Reynold's rules. The main behavioral objectives using the potential function approach for a group of vehicles are aggregation, social foraging and formation control [79, 80]. For aggregation, the objective is to bring the group together while avoiding inter-vehicle collisions. And in social foraging, the behavior resembles the search of an environment for areas of interest while avoiding areas of potential danger. And formation control is to have the vehicles achieve a finite geometrical structure while moving together [80].

One approach using APF cooperative control is demonstrated in reference [81]. In [81], double integrator dynamics are assumed and the potential function defines the interaction between neighboring vehicles with a virtual leader providing a moving reference trajectory to track. The virtual leader is introduced to provide direction and possibly manipulate the group's geometry. Another example using a point-mass dynamics is shown in reference [82] where guidelines for constructing a potential energy function is discussed. In [82], the potential function is defined as a composition of different functions each designed to attain a certain performance. Both references [81] and [82] discuss flocking behavior of multiple agents moving together to different objectives in a given environment. While references [81] and [82] consider vehicles with point mass dynamics, the author in reference [83] provides a cooperative APF framework for unicycle mobile robots. In [83], each robot is assumed to have a safety area and communication area which are designed to dictate the communication protocol of the group. The APF introduced in [83] is a smooth  $p$ -differentiable bump function designed such that the vehicles in the group can track a reference trajectory while avoiding collisions with all other robots. The research presented in reference [79] gives an alternate APF formulation along with guidance for defining such a potential function for cooperative control. The authors in [79] define a general class of odd functions that have attractive and repulsive components that can be solved to find the equilibrium distances between the agents with tuning of several design parameters. Several objectives can be reached due to the properties of the APF defined by the authors in [79] such as stable aggregation, and formation control for a group. The work in reference [80] extends the results in [79] to include social foraging considerations and applies the results to non-holonomic agents where the APF provides reference values to be tracked with a sliding mode controller. Other extensions inspired by the research in reference [79] can be found in references [84] and [85]. Reference [84] presents guidelines for overcoming some potential pitfalls with APF frameworks. The problem areas addressed in [84] consider a non-reachable goal (local minimum), obstacle collision (when the attractive potential overwhelms the repulsive component), obstacle collisions in swarms and inter-agent collisions. The guidelines in [84] combines additive and multiplicative configurations of the APF to address the problems with a point mass system. Then, reference [85] makes use of the APF guidelines found in [79] for use on a system of quadcopters with multi-loop control and some considerations towards obstacle avoidance. The research in this project is influenced by the APF design found in reference [79]. An additional contribution from this aspect of the project is the insertion of the numerical navigation function for social foraging tasks.

## 3.2 Objectives and Contributions

### 3.2.1 List of Contributions

- Developed a path planning algorithm that considers a system's kinematics and control effort.
- Designed path planning algorithm that plans to a reachable state.
- Derived a stable backstepping guidance law that ensures bounded tracking errors.
- Derived a stable nonlinear model predictive control-based guidance law that can enforce constraints on the inputs and outputs of the system.
- Implemented a combination of the path planning algorithm and either of the nonlinear controllers to construct a guidance and control framework.
- Combined the path planning algorithm's potential  $\phi_i$  with cooperative potential functions for a group of rover vehicles for aggregation and social foraging tasks.
- Applied guidance techniques, individual and cooperative, to real-time mobile robot platforms.

### 3.2.2 List of Published Works

- P. Quillen, "A Framework for Optimal Path Planning and Nonlinear Guidance for Autonomous Mobile Robots", PhD Dissertation, Aerospace Engineering, The University of Texas at Arlington, August 2018
- P. Quillen, K. Subbarao and J. Muñoz, "Cooperative Control with Minimum Control Effort Based Navigation Functions," in *Journal of Intelligent Robotics*. (submitted)
- P. Quillen, K. Subbarao, "Real-Time Nonlinear Model Predictive Control for Wheeled Mobile Robots," in *Journal of Advanced Robotics Systems*. (submitted)
- P. Quillen, K. Subbarao and J. Muñoz, "Path Planning to a Reachable State Using Minimum Control Effort Based Navigation Functions," accepted for publication in *Journal of Astronautical Sciences*.
- P. Quillen, K. Subbarao, "Minimum Control Effort Based Path Planning and Nonlinear Guidance for Autonomous Mobile Robots", *International Journal of Advanced Robotic Systems (Sage Publications)*, Volume 15, issue 6, November, 2018.
- VNV, Murali, DK. Baman, K. Subbarao, "Nonlinear Guidance Laws for Trajectory Tracking over a Mobile Communication Network applied to Unmanned Ground Vehicles", IEEE – ICCA, Anchorage, AK, June 2018
- VNV. Murali. "Real-time Minimum Jerk Optimal Trajectory Synthesis and Tracking for Ground Vehicle Applications", MS Thesis, Mechanical Engineering, The University of Texas at Arlington, December 2017
- DK, Baman, "Pseudo-Spectral Methods based Real-Time Optimal Path Planning for Unmanned Ground Vehicles", MS Thesis, Mechanical Engineering, The University of Texas at Arlington, December 2017

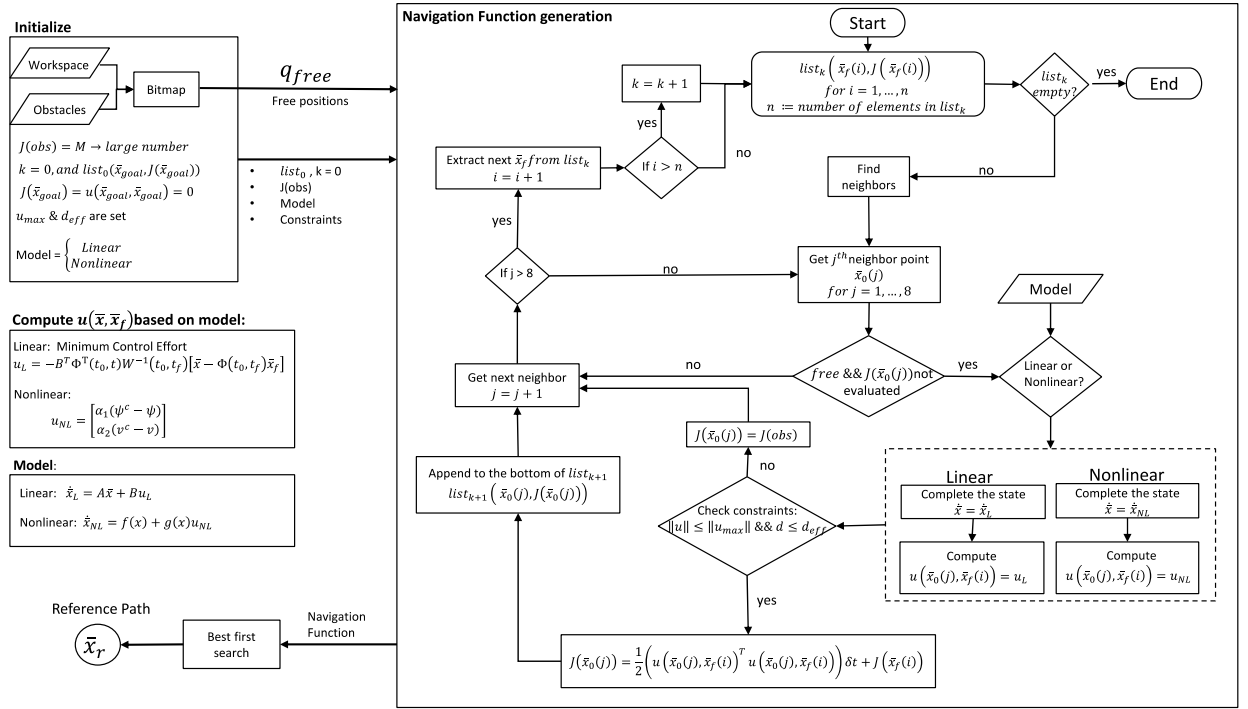
- P. Quillen, J. Muñoz and K. Subbarao, “Path Planning to a Reachable State Using Inverse Dynamics and Minimum Control Effort Based Navigation Functions,” in AAS/AIAA Space Flight Mechanics Meeting, no. AAS 17-849, 2017. (Reference [73])
- Godbole, V. Murali, P. Quillen and K. Subbarao, “Optimal Trajectory Design and Control of a Planetary Exploration Rover,” in Advances in the Astronautical Sciences Spaceflight Mechanics, vol. 160, 2017. (Reference [74])
- P. Quillen, K. Subbarao and J. Muñoz, “Guidance and Control of a Mobile Robot via Numerical Navigation Functions and Backstepping for Planetary Exploration Missions,” in AIAA Space 2016, AIAA Space Forum, (AIAA 2016-5237). (Reference [72])

## 4.0 METHODS, ASSUMPTIONS, AND PROCEDURES

In this section, the different problems discussed above are elaborated upon on detail.

### 4.1 Path Planning using Minimum Control Effort based Navigation Functions

The main contribution of this work is the construction of the navigation function, which is motivated by the simple wavefront expansion described in references [35] and [35]. However, the novel design discussed in this work is different in that it uses a metric based on the control effort of the system to form the contour levels in the potential field as opposed to the traditional distance-based metric. Two new methods are introduced for constructing the navigation function. The first method finds a minimum control effort path to a reachable state. This method uses the solution to the minimum control effort problem given a fixed initial and final state for a linear system. Then, the second method takes an inverse dynamics approach from a reachable state. This approach is considered for a nonlinear model. Both methods are constructed to reach an objective reachable state, expressed as  $\bar{\mathbf{x}}_{goal}$ . The full algorithm is outlined in Figure 5.



**Figure 5: Control Effort Based Navigation Function Algorithm**

#### 4.1.1 Initialization

The initialization block of this numerically constructed potential field begins by discretizing the workspace into an evenly spaced grid. This grid can be scaled to fit over any working environment. The discretized grid and the obstacle positions are then used to form a bitmap representation of the environment. This allows the free points to be identified and extracted. The free space is denoted as  $\mathbf{q}_{free}$  in the algorithm, where  $\mathbf{q}$  in general represents a configuration in the workspace. For example, in the two-dimensional workspace considered,  $\mathbf{q} \in \mathcal{R}^2$ , and is given by the vector  $\mathbf{q} = [x, y]^T$ . Also, during initialization, the obstacle potential levels are set uniformly to a large number and these configurations are ignored for the rest of the algorithm. The potential level of the desired final state is set to zero, to ensure that it is the global minimum, and these values are inserted into  $list_k$ , where the index  $k = 0$  initially, i.e.  $list_0(\mathbf{x}_{goal}, 0)$ .

Another attribute that is set during initialization is the model to base the navigation function generation. The model will motivate how to set the objective state,  $\mathbf{x}_{goal}$ , how the state is approximated in the neighboring points and how the cost associated with the control effort is computed. There are two different models considered in this work. The first model is a linear double integrator model used to determine the minimum control effort path to a reachable state and the second is a nonlinear rover model used to find an inverse dynamics based path to a reachable state. Each model is described in the following subsections.

Approved for public release; distribution is unlimited.

#### 4.1.2 Minimum Control Effort (MCE) Based on Linear Rover Model

The minimum control effort based path plan in this paper is designed based on the solution to the optimal control problem for finding the minimum control energy. The cost function associated with this problem is given by

$$\min_{\mathbf{u} \in \mathbb{R}^m} J = \frac{1}{2} \int_{t_0}^{t_f} \mathbf{u}^T \mathbf{u} dt \quad (1)$$

subject to

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{x}(t_0) &= \mathbf{x}_0 \\ \mathbf{x}(t_f) &= \mathbf{x}_f \end{aligned} \quad (2)$$

The solution to this problem can be found in most optimal control textbooks such as reference [86]. With the problem and constraints posed in equation 2, the analytical solution for the minimum energy controller is given by

$$\mathbf{u}_L = -\mathbf{B}^T \mathbf{\Phi}^T(t_0, t) \mathbf{W}^{-1}(t_0, t_f) [\mathbf{x}_0 - \mathbf{\Phi}(t_0, t_f) \mathbf{x}_f] \quad (3)$$

Where  $\mathbf{W}(t_0, t_f)$  is the controllability grammian and  $\mathbf{\Phi}(t, t_0)$  is the state transition matrix of the system [86-88]. The state transition matrix for the system (of state dimension  $n$ ) represented by equation 2 is obtained by solving the matrix differential equation,

$$\dot{\mathbf{\Phi}} = \mathbf{A}\mathbf{\Phi}, \quad \mathbf{\Phi}(t_0, t_0) = \mathbf{I}_{n \times n} \quad (4)$$

The solution to equation 4 is  $\mathbf{\Phi}(t, t_0) = \exp(\mathbf{A}(t - t_0))$

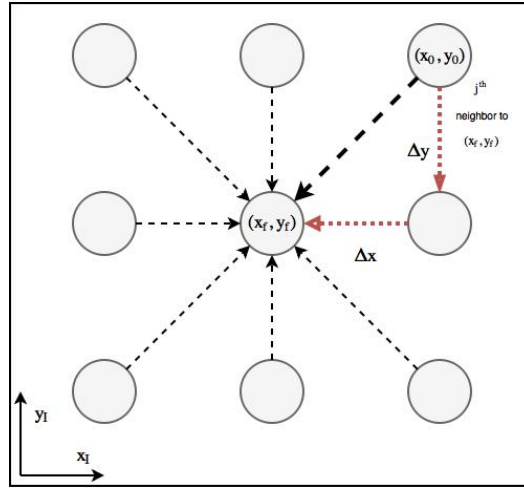
The controllability grammian is computed by

$$\mathbf{W}(t_0, t_f) = \int_{t_0}^{t_f} \mathbf{\Phi}(t_0, \tau) \mathbf{B}(\tau) \mathbf{B}^T(\tau) \mathbf{\Phi}^T(t_0, \tau) d\tau \quad (5)$$

Which, needs to be non-singular in order for the system to be controllable. The system here is considered as a double-integrator with the two dimensional examples. The control input,  $\mathbf{u}_L$  is defined as a two dimensional vector given by  $\mathbf{u}_L = [u_x, u_y]^T$ , where  $u_x$  and  $u_y$  are the control inputs for the  $x$  and  $y$  accelerations of the system.



In order to compute the control effort with equation 1, the state information at  $t_0$  and at  $t_f$ , first needs to be obtained. For the path planning algorithm, the final state is taken as the  $i^{th}$  state extracted from the  $k^{th}$  list, i.e.  $\mathbf{x}_f(i)$ , which is initially set as the desired final reachable state. Then, for  $t_0$ , the state information is taken from the neighboring points of  $\mathbf{x}_f(i)$  and are denoted by  $\mathbf{x}_0$ . During the execution of the algorithm, the state at  $t_0$  for the neighboring points needs to be completed based on the assumption that it is approaching the state at  $t_f$ . **Figure 6** illustrates how the state is completed for a system with double integrator dynamics.



**Figure 6: Grid set up in two dimensional space for the minimum control effort**

The equations of motion for the double integrator is given in matrix form as

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1/m & 0 \\ 0 & 1/m \end{bmatrix} \mathbf{u} \quad (6)$$

Where  $m$  is the mass of the rover and the state at any given instant for a double integrator system is given by the vector  $\mathbf{x} = [x, y, \dot{x}, \dot{y}]^T$ , denoting the position in the two dimensional workspace and the velocities in the  $x$  and  $y$  directions. The objective reachable state in this instance is given by the vector,  $\mathbf{x}_{goal}^{mce} = [x_g, y_g, \dot{x}_g, \dot{y}_g]^T$ . This model is considered as a point mass rover with dynamics.

The position and velocity information for the neighbor points is set based on the model. The position information for the neighbor points is gathered from the discretized grid. Thus, the values that need to be computed in order to complete the state,  $\mathbf{x}_0$  are the velocities in the  $x$  and  $y$  directions. These values are computed by first specifying a time step,  $\delta t$  to complete a maneuver going from  $\mathbf{x}_0$  to  $\mathbf{x}_f$ . Then, the velocity in the  $x$  and  $y$  directions for  $\mathbf{x}_0$  are calculated as

$$\begin{aligned}\dot{x}_0 &= \dot{x}_f - \frac{x_f - x_0}{\delta t} \\ \dot{y}_0 &= \dot{y}_f - \frac{y_f - y_0}{\delta t}\end{aligned}\tag{7}$$

This is done for each of the free neighbors of  $\bar{\mathbf{x}}_r$ . This approach will influence the shaping of the navigation functions contours based on the final reachable state that is given based on a linear point mass rover model with dynamics.

#### 4.1.3 Inverse Dynamics (ID) Based Control Effort For a Nonlinear Rover Model

If the rover model is chosen to be nonlinear during initialization, then an inverse dynamics based method for finding the control effort is used. The rover's state vector is given by  $\mathbf{x} = [x, y, \psi, v]^T$  which represents the position, heading angle, and speed at a given instant. The equations of motion for the nonlinear rover model are as follows,

$$\begin{aligned}\dot{x} &= v \cos(\psi) \\ \dot{y} &= v \sin(\psi) \\ \dot{\psi} &= u_1 \\ \dot{v} &= u_2\end{aligned}\tag{8}$$

For the path plan algorithm, the control terms are found in the rate of change of heading angle (turn rate) and the acceleration of the system ( $u_1$  and  $u_2$  respectively). The control terms can be rewritten as

$$\begin{aligned}u_1 &= \alpha_1 (\psi^c - \psi) \\ u_2 &= \alpha_2 (v^c - v)\end{aligned}\tag{9}$$

Where,  $\alpha_1 > 0$ ,  $\alpha_2 > 0$  are constants;  $\psi^c$  and  $v^c$  are the guidance commands for the heading angle and speed of the system respectively. Then, the two dimensional control vector is

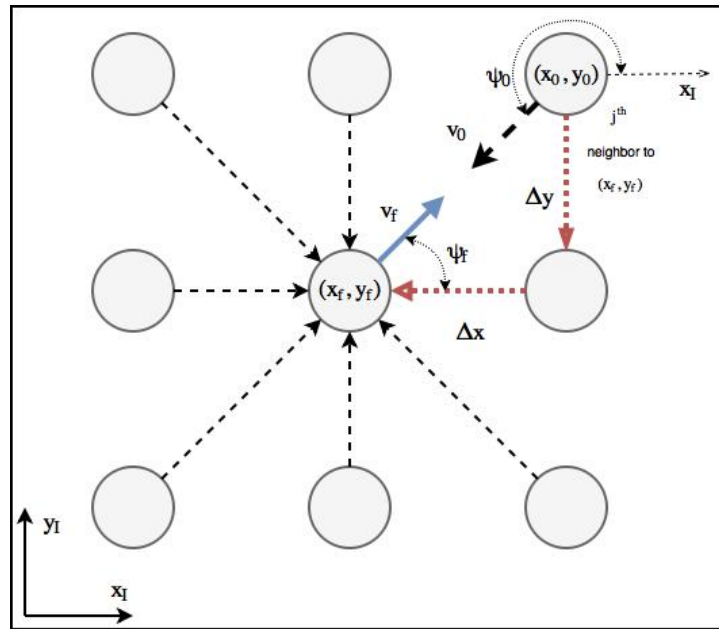
$$\mathbf{u}_{\text{NL}} = \begin{bmatrix} \alpha_1 (\psi^c - \psi) \\ \alpha_2 (v^c - v) \end{bmatrix}\tag{10}$$

The control commands ( $\psi^c$  and  $v^c$ ) used in equation 10 are derived from a stable nonlinear trajectory tracking guidance law design}. The derivation of these control inputs with this rover model is given in the backstepping-like control design section along with a proof of its stability.

The actual commanded heading angle and speed values are computed based on the stability analysis of the system. However, for the navigation function algorithm, it is assumed that the commanded heading angle and speed values are set as constants in the state at the end of a maneuver, at  $t_f$ . Therefore, it is included as part of the final state, denoted as  $\mathbf{x}_f$ . The final state  $\mathbf{x}_f$

is hence set as the desired reachable state given by  $\mathbf{x}_{goal}^{ID} = [x_g, y_g, \psi_g, v_g]^T$ .

In order to compute the control effort with equation 10, the state for the neighboring points must be completed based on the nonlinear model. The position information is extracted from the discretized workspace, but the heading angle and velocity values still need to be calculated in order to complete the neighbor's state  $\bar{\mathbf{x}}_0$ . **Figure 7** illustrates how the heading and velocity values are found.



**Figure 7: Grid set up in two dimensional space for the inverse dynamics based path plan problem**

The values for  $\psi_0$  and  $v_0$  at the neighbor points of  $\mathbf{x}_f$  are found assuming an approaching maneuver from  $\mathbf{x}_0$  to  $\mathbf{x}_f$  over a fixed time step. From **Figure 7**, the heading angle at the neighboring point is taken as an approach angle directed towards the desired final position, and it is measured from the inertial  $\mathbf{x}$ -direction. So, the heading angle can be found by  $\psi_0 = \tan^{-1} \left( \frac{\dot{y}_0}{\dot{x}_0} \right)$  Where

$$\begin{aligned} \dot{y}_0 &= v_f \sin(\psi_f) + \frac{y_f - y_0}{\delta t} \\ \dot{x}_0 &= v_f \cos(\psi_f) + \frac{x_f - x_0}{\delta t} \end{aligned} \quad (11)$$

Approved for public release; distribution is unlimited.

And the velocity at the neighboring point is found using  $v_0 = \sqrt{\dot{x}_0^2 + \dot{y}_0^2}$

These values are computed for each of the free neighboring points of  $\mathbf{x}_f$ . This approach will influence the shaping of the navigation functions contours based on the final reachable state that is given based on the nonlinear rover model.

#### 4.1.4 Navigation Function Generation

Once set, the information from the initialization block is passed into the algorithm to generate the navigation function. Within the algorithm, the states and potentials within  $list_k$  are used to evaluate the values of their neighbors,  $\mathbf{x}_0(j)$ , where the index  $j$  denotes the  $j^{th}$  neighbor point. For the two-dimensional workspace examples in this paper, the algorithm considers each of the 8 neighbors of the  $i^{th}$  state,  $\mathbf{x}_f(i)$ , in  $list_k$ . First, it must be determined if the neighbor point is in the free space and if the potential has not been computed. If the neighbor point is both free and has not been evaluated then the algorithm continues, otherwise the next neighbor point is considered. This step serves to ensure that there are no overlapping values and that the navigation function is only evaluated in the free space.

For the free neighbor points, the algorithm continues by completing the state since the only information in  $\mathbf{x}_0(j)$ , at this point, is its position in the workspace. The state information is completed based on which model was designated in the initialization block, either the linear minimum control effort approach or nonlinear inverse dynamics approach. The description for how the state  $\mathbf{x}_0(j)$  is completed is given in the preceding subsections of this paper.

Also, the control effort to go from  $\mathbf{x}_0(j)$  to  $\mathbf{x}_f(i)$ , is computed based on the model according to the methods described in the previous subsections. Therefore, the value for  $\mathbf{u}(\mathbf{x}_0(j), \mathbf{x}_f(i))$  is calculated using either equation 3 or 7. The resulting value is then compared with the constraints

$$\|\mathbf{u}_{0,j}\| \leq u_{\max} \quad (12)$$

$$d_j \leq d_{\text{eff}} \quad (13)$$

where  $\mathbf{u}_{0,j} = \mathbf{u}(\mathbf{x}_0(j), \mathbf{x}_f(i))$ , and  $d_j$  is the Euclidean distance between the position coordinates of  $\mathbf{x}_0(j)$  and the objective location  $(x_g, y_g)$ , i.e.  $d_j = \sqrt{(x_{0,j} - x_g)^2 + (y_{0,j} - y_g)^2}$

Also  $u_{\max}$  and  $d_{\text{eff}}$  are the maximum control limits (includes both maximum acceleration and turn rate) and effective distance respectively, which are set during initialization. These constraints are user defined and help to direct the reference paths generated by restricting configurations which approach the final state from an undesired direction. The effective distance keeps the constraints close to the reachable objective state and ensures that it will not affect the entire workspace.

If the constraints are violated, then that  $\mathbf{x}_0(j)$  is considered as an obstacle and its cost is set equal to that of the obstacle configurations, i.e.  $J(\mathbf{x}_0(j)) = J(\text{obs})$  whereas the cost for the neighboring states that satisfy the constraints are computed by

$$J(\mathbf{x}_0(j)) = \frac{1}{2}(\mathbf{u}_{0,j}^T \mathbf{u}_{0,j}) \delta t + J(\mathbf{x}_f(i)) \quad (14)$$

The cost computed in equation 14 is used to create the contour levels making up the navigation function. One can observe that each neighboring state will have a different cost associated with it and it depends upon the control effort term  $\frac{1}{2}(\mathbf{u}_{0,j}^T \mathbf{u}_{0,j})$ . Also, the cost equation can be expressed so that it may incorporate the constraints in its formulation. Thus, accounting for all the cases mentioned earlier, the composite cost function for a candidate neighboring state  $j$  is considered as

$$J(\mathbf{x}_0(j)) = \frac{1}{2}(\mathbf{u}_{0,j}^T \mathbf{u}_{0,j}) \delta t + J(\mathbf{x}_f(i)) + \delta_{\text{con}} \quad (15)$$

And the  $\delta_{\text{con}}$ -function is defined as

$$\delta_{\text{con}} = \begin{cases} J(\text{obs}) & \text{if } \mathbf{u}_{0,j}^T \mathbf{u}_{0,j} - u_{\text{max}} \geq 0 \ \& \ d_j \leq d_{\text{eff}} \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

Finally, when the cost for  $\mathbf{x}_0(j)$  is computed, it is appended to the bottom of a new list, indexed at  $k+1$ , written as  $\text{list}_{k+1}(\mathbf{x}_0(j), J(\mathbf{x}_0(j)))$  in Figure 5. This process is continued for each neighboring configuration of  $\bar{\mathbf{x}}_f(i)$ . If all the neighbors have been visited, then the next final state and cost in  $\text{list}_k$  is used, i.e.  $\mathbf{x}_f(i+1)$  and  $J(\mathbf{x}_f(i+1))$ . However, if all the elements in  $\text{list}_k$  have been used, then the next list,  $\text{list}_{k+1}$  is considered. The algorithm continues until  $\text{list}_k$  has no information within left. If  $\text{list}_k$  is empty, then all the points in the workspace that are connected to the objective position have been visited by the wavefront expansion and the navigation function generation is finished.

Once the navigation function has been generated, the reference path,  $\mathbf{x}_r$  is obtained through a best-first graph search following the negative gradient of the resulting cost (or potential) field from a given initial location to the objective location. Through this method, the paths are attained by observing the cost values of the neighboring points and then choosing the neighbor with the lowest value. This is done in an iterative manner until the objective is found. Hence, the path is found by starting at some initial location in the environment and continuing along a path of minimal cost until it reaches the goal.

#### 4.1.5 Trajectory Generation

Once a reference path,  $\mathbf{x}_r$  has been gathered from the navigation function planner, a trajectory needs to be created to provide the necessary reference signals for the guidance law. The reference signals needed for the trajectory tracking guidance law are  $[x_r, y_r, \dot{x}_r, \dot{y}_r, \ddot{x}_r, \ddot{y}_r]$ , which represent the desired position, speed and acceleration in the  $x$  and  $y$  directions. In order to generate the trajectory, a desired time to reach the goal needs to be set. With the goal time set, a time stamp to reach each point can be assigned by creating a time vector that can be evenly spaced across all the points in the reference path. In other words the time step,  $\delta t$  it takes to go from one point to the next along the path is held constant.

Then, the reference speeds along the path can be computed by

$$\begin{aligned}\dot{x}_r(i+1) &= \frac{x_r(i+1) - x_r(i)}{\delta t} \\ \dot{y}_r(i+1) &= \frac{y_r(i+1) - y_r(i)}{\delta t}\end{aligned}\tag{17}$$

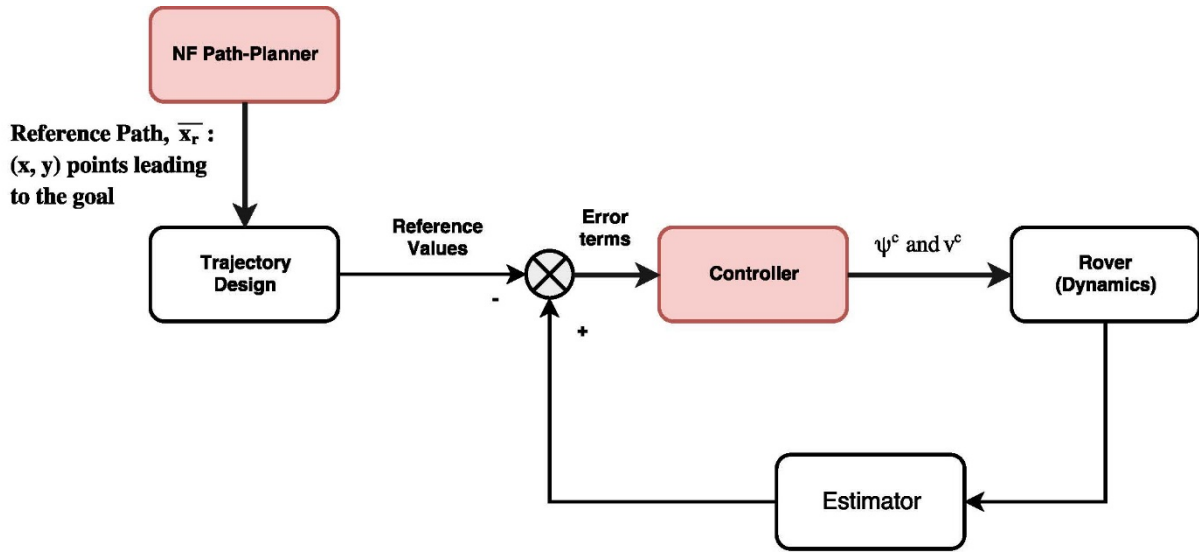
Where  $i$  is the index number of the point in the path. And it is assumed that the initial velocity is zero, i.e.  $\dot{x}_r(0) = 0$  and  $\dot{y}_r(0) = 0$ . The reference acceleration signals are computed by

$$\begin{aligned}\ddot{x}_r(i+1) &= \frac{\dot{x}_r(i+1) - \dot{x}_r(i)}{\delta t} \\ \ddot{y}_r(i+1) &= \frac{\dot{y}_r(i+1) - \dot{y}_r(i)}{\delta t}\end{aligned}\tag{18}$$

Where  $i$  is the index number and it is assumed that the initial acceleration is zero, i.e.  $\ddot{x}_r(0) = 0$  and  $\ddot{y}_r(0) = 0$ . Finally, when the vehicle is traveling between points in  $\mathbf{x}_r$  the reference values are found by linear interpolation.

## 4.2 Nonlinear Lyapunov Based (Backstepping-Like) Control Design

An illustration of the framework for the guidance and control of a planetary exploration rover is given in **Figure 8**. The contributions of this work are the NF path planner and the backstepping-like control design, highlighted in **Figure 8**. The NF path planner is the main contribution which will supply a reference path for the system to follow. The path is then sent to the trajectory generation block. Once a trajectory is fitted along the path, the reference values for the control design become available. Then, the backstepping-like controller provides high level actuator guidance commands for the heading angle and velocity to drive the rover along the trajectory to the objective.

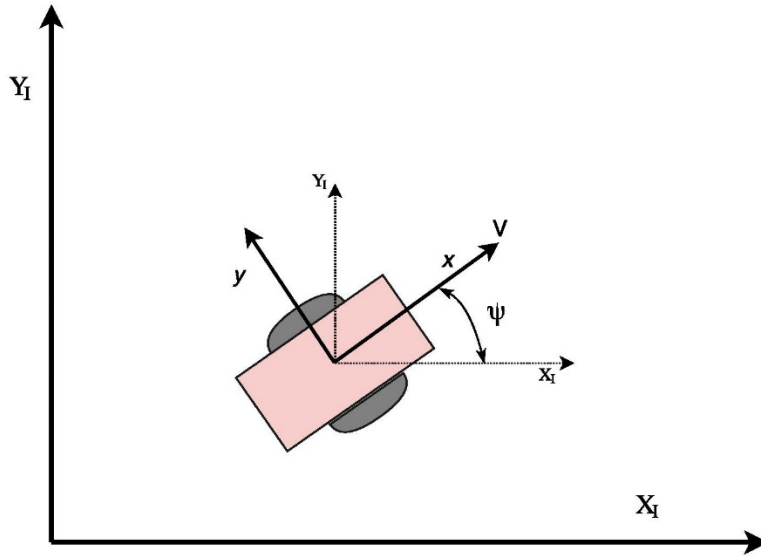


**Figure 8: Block diagram of the information flow of the framework**

The kinematic model of the rover is illustrated in **Figure 9**. For the results discussed, an East-North-Up (ENU) convention is used. The heading angle,  $\psi$ , is defined positively going in a counter clockwise direction about the  $z$ -axis (up) going from the inertial  $x$ -axis ( $x_I$ ) to the body  $x$ -axis ( $x_b$ ). The state of the rover model is taken as the inertial position in  $x$  and  $y$  coordinates, its heading angle and its forward velocity, and it can be represented by the vector  $\mathbf{x} = [x, y, \psi, v]^T$ . Without loss of generality, the subscript  $I$  for the inertial  $x$  and  $y$  positions of the rover is dropped. The equations of motion for the rover, considered solely in the path planner, are given by

$$\begin{aligned}
 \dot{x} &= v \cos(\psi) \\
 \dot{y} &= v \sin(\psi) \\
 \dot{\psi} &= \alpha_1 (\psi^c - \psi) \\
 \dot{v} &= \alpha_2 (v^c - v)
 \end{aligned} \tag{19}$$

Where  $\psi^c$  and  $v^c$  as mentioned previously are the commanded heading and speed values. These equations of motion can be used as a representative model for a wheeled rover with control over the heading angle and speed. Recall the control terms in equation 19, were used a bit differently in the path planning section. The rationale there was to constrain the vehicle turn rate and linear acceleration so as to derive feasible reference trajectories. Stability of these equations is now shown to prove their feasibility in the navigation function algorithm.



**Figure 9: Kinematics of the rover vehicle**

**Theorem 1:**

Given a 2-D,  $C^2$  trajectory with reference values,  $x_r, y_r, \dot{x}_r, \dot{y}_r, \ddot{x}_r, \ddot{y}_r$ . Then, the guidance commands in equations 13, and 14 along with the equations of motion given in equation 11 will guarantee that  $\|x - x_r\| \rightarrow 0$ , and  $\|y - y_r\| \rightarrow 0$  as  $t \rightarrow \infty$

$$\psi^c = \frac{1}{\alpha_1} (\dot{\psi}_d - \lambda_\psi e_\psi) + \psi \tag{20}$$

$$v^c = \frac{1}{\alpha_2} (\dot{v}_d - \lambda_v e_v) + v \tag{21}$$

where  $\lambda_\psi > 0, \lambda_v > 0$ , and where  $\psi_d, v_d, \dot{\psi}_d$  and  $\dot{v}_d$  are nonlinear functions of the reference trajectory, the state and the state tracking errors; and  $e_\psi = \psi - \psi_d, e_v = v - v_d$ .

Proof: The proof can be found in Reference [73].

**[QED]**



The virtual heading angle and velocity signals are synthesized as follows. The position tracking errors are given by  $e_x = x - x_r$ ,  $e_y = y - y_r$  and their time derivatives as  $\dot{e}_x = \dot{x} - \dot{x}_r$  and  $\dot{e}_y = \dot{y} - \dot{y}_r$ . The desired heading and velocity signals, are prescribed such that the position tracking error dynamics is exponentially stable, i.e.  $\dot{e}_x = -\lambda_x e_x$  and  $\dot{e}_y = -\lambda_y e_y$  for  $\lambda_x > 0$ , and  $\lambda_y > 0$ . It follows that

$$\begin{aligned} v_d \cos(\psi_d) &= \dot{x}_r - \lambda_x e_x \\ v_d \sin(\psi_d) &= \dot{y}_r - \lambda_y e_y \end{aligned} \quad (22)$$

Then, using equation 22 the desired heading and velocity signals for exponentially stable position tracking are obtained as

$$\psi_d = \tan^{-1} \left( \frac{\dot{y}_r - \lambda_y e_y}{\dot{x}_r - \lambda_x e_x} \right) \quad (23)$$

$$v_d = \sqrt{(\dot{x}_r - \lambda_x e_x)^2 + (\dot{y}_r - \lambda_y e_y)^2} \quad (24)$$

The heading and speed commands are derived such that the respective tracking error dynamics,  $e_\psi = \psi - \psi_d$  and  $e_v = v - v_d$ , are exponentially stable. Thus the error dynamics are prescribed as  $\dot{e}_\psi = -\lambda_\psi e_\psi$  and  $\dot{e}_v = -\lambda_v e_v$ . Combining equation 22 with the state tracking error time derivatives leads to

$$\dot{\psi} = \alpha_1 (\psi^c - \psi) = \dot{\psi}_d - \lambda_\psi e_\psi \quad (25)$$

$$\dot{v} = \alpha_2 (v^c - v) = \dot{v}_d - \lambda_v e_v \quad (26)$$

Thus,

$$\psi^c = \psi + \frac{1}{\alpha_1} (\dot{\psi}_d - \lambda_\psi e_\psi) \quad (27)$$

$$v^c = v + \frac{1}{\alpha_2} (\dot{v}_d - \lambda_v e_v) \quad (28)$$

Which are the control terms introduced in equations 20, and 21. The time derivatives for the desired heading angle and velocity values are obtained by differentiating equations 23 and 24 with respect to time, leading to

$$\dot{\psi}_d = \frac{1}{v_d} \left[ \cos(\psi_d) (\ddot{y}_r - \lambda_y \dot{e}_y) - \sin(\psi_d) (\ddot{x}_r - \lambda_x \dot{e}_x) \right] \quad (29)$$

$$\dot{v}_d = \cos(\psi_d)(\ddot{x}_r - \lambda_x \dot{e}_x) + \sin(\psi_d)(\ddot{y}_r - \lambda_y \dot{e}_y) \quad (30)$$

Where  $\dot{e}_x = v \cos(\psi) - \dot{x}_r$  and  $\dot{e}_y = v \sin(\psi) - \dot{y}_r$ .

Note that when  $v_d = 0$ ,  $\dot{\psi}_d$  is not defined. Therefore, a singularity avoidance scheme must be in place to handle this situation in practice. The closed loop stability of the system is verified through a Lyapunov stability analysis. For a function to be considered a valid Lyapunov function it must be positive definite, its derivative must be negative definite and both the function and its derivative must be equal to zero at the equilibrium points (i.e. the origin) [73,89].

### 4.3 Nonlinear Model Predictive Control

An alternate guidance scheme that accommodates system (state, and control) constraints uses a nonlinear model predictive control (NMPC) approach and is presented here. The procedure is motivated by the work discussed in [67] where the performance of the traditional approach of NMPC is compared with that of a state dependent coefficient (SDC) approach. The objective in this section is to present this method for an autonomous mobile robot and to verify its trajectory tracking capability in simulation and then compare its performance with guidance scheme derived in section 4.2.

Given a general nonlinear system of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u} \quad (31)$$

a state-space representation of the system is obtained wherein the system matrices are given as functions of the current state of the system as

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}(\mathbf{x})\mathbf{x} + \mathbf{B}(\mathbf{x})\mathbf{u} \\ \mathbf{y} &= \mathbf{C}(\mathbf{x})\mathbf{x} + \mathbf{D}(\mathbf{x})\mathbf{u} \end{aligned} \quad (32)$$

where  $\mathbf{x} \in \mathcal{R}^n$ ,  $\mathbf{u} \in \mathcal{R}^m$ ,  $\mathbf{y} \in \mathcal{R}^{n_o}$  are respectively the state, input and output vectors and  $\mathbf{A}(\mathbf{x}) \in \mathcal{R}^{n \times n}$ ,  $\mathbf{B}(\mathbf{x}) \in \mathcal{R}^{n \times m}$ ,  $\mathbf{C}(\mathbf{x}) \in \mathcal{R}^{n_o \times n}$ ,  $\mathbf{D}(\mathbf{x}) \in \mathcal{R}^{n_o \times m}$  are the continuous state dependent system matrices in the SDC factored form. The pairs  $(\mathbf{A}(\mathbf{x}), \mathbf{B}(\mathbf{x}))$  and  $(\mathbf{A}(\mathbf{x}), \mathbf{C}(\mathbf{x}))$  are considered controllable and observable  $\forall \mathbf{x} \in \mathcal{R}^n$ .

In general, the formation of SDC matrices for the system are not unique unless observing a scalar system [51-54]. Therefore, different SDC forms may be obtained for a given system and solutions to the optimization problems posed may vary. Two particular SDC factorizations of the kinematics, are derived.

For the results, it is assumed that the full state is available at each instant, i.e. the system output matrix  $\mathbf{C}(\mathbf{x}) = \mathbf{C}$  is considered as the constant identity matrix, i.e.  $\mathbf{I} \in \mathfrak{R}^{4 \times 4}$ . Also, from the kinematic equations, it is evident that  $\mathbf{D}(\mathbf{x}) = \mathbf{0}$  and the input matrix  $\mathbf{B}(\mathbf{x}) = \mathbf{B}$  is a constant matrix given by

$$\mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (33)$$

And one possible solution for the SDC factored  $\mathbf{A}(\mathbf{x})$  matrix can be given by

$$\mathbf{A}(\mathbf{x}) = \begin{bmatrix} \mathbf{0}_{2 \times 2} & \mathbf{A}_{12}(\mathbf{x}) \\ \mathbf{0}_{2 \times 2} & \mathbf{0}_{2 \times 2} \end{bmatrix} \quad (34)$$

where

$$\mathbf{A}_{12}(\mathbf{x}) = \begin{bmatrix} v \left( \frac{\psi^3}{4!} - \frac{\psi}{2!} \right) & 1 \\ v \cos\left(\frac{\psi}{2}\right) \left( \frac{1}{2} - \frac{\psi^2}{2^3 3!} + \frac{\psi^4}{2^5 5!} \right) & 2 \sin\left(\frac{\psi}{2}\right) \cos\left(\frac{\psi}{2}\right) \end{bmatrix} \quad (35)$$

This result is derived using the following trigonometric identity and expansions

$$\sin(2\psi) = 2 \sin(\psi) \cos(\psi)$$

$$\sin(\psi) = \psi - \frac{\psi^3}{3!} + \frac{\psi^5}{5!}$$

$$\cos(\psi) = 1 - \frac{\psi^2}{2!} + \frac{\psi^4}{4!}$$

The above derivation of the  $\mathbf{A}_{12}$  quadrant in equation 35 has some issues in implementation where the heading angle is limited from  $-\frac{\pi}{2} \leq \psi \leq \frac{\pi}{2}$ . Therefore, another formulation of the  $\mathbf{A}_{12}$  quadrant is derived as

$$\mathbf{A}_{12}(\mathbf{x}) = \begin{bmatrix} v \left( \frac{\psi^3}{4!} - \frac{\psi}{2!} \right) & 1 \\ v \left( -\frac{\psi^2}{3!} + \frac{\psi^4}{5!} \right) & \psi \end{bmatrix} \quad (36)$$

Note that with both SDC matrix results from equations 35 and 36, the pseudo-linear system  $\mathbf{A}(\mathbf{x})$  matrix becomes rank deficient when  $v = 0$ . This complication can be addressed by adding a state constraint on the velocity so that  $v > 0$  in the solution. Also, note that if  $\psi = 0$ , the resulting SDC matrix with  $\mathbf{A}_{12}$  coming from equation 36 becomes uncontrollable. Therefore, in the simulations, the  $\mathbf{A}(\mathbf{x})$  matrix is switched from using the  $\mathbf{A}_{12}$  quadrant from equation 36 to using the formulation in equation 35 whenever  $\|\psi\| \leq \psi_{tol}$ . Where  $\psi_{tol}$  is a tolerance value defined in the interval  $[-\frac{\pi}{2}, \frac{\pi}{2}]$ .

Next, a discrete-time equivalent of the system shown in equation 32 can be obtained by introducing a zero-order-hold (ZOH) with a specified sample time ( $\Delta t$ ). With the ZOH, a sampled point is held constant over a sampled time interval. The smaller the sample time, the more accurate the approximation of the continuous signal. The discrete-time equivalent system is given in the following form

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{\Phi}(\mathbf{x}_k)\mathbf{x}_k + \mathbf{\Gamma}\mathbf{u}_k \\ \mathbf{y}_k &= \mathbf{C}\mathbf{x}_k\end{aligned}\tag{37}$$

where  $\mathbf{\Phi}(\mathbf{x}_k)$  and  $\mathbf{\Gamma}$  are discrete approximations of the continuous  $\mathbf{A}(\mathbf{x})$  and  $\mathbf{B}$  matrices respectively. The discrete system matrices in equation 22 are of the SDC form and can be considered as constants over the sampling time,  $\Delta t = t_{k+1} - t_k$ , where the interval is given by  $[t_k, t_{k+1})$ .

The discrete-time system in equation 22 can be placed in batch form for an N-step prediction horizon as

$$\mathbf{X}_k = \mathbf{F}(\mathbf{x}_k)\mathbf{x}_k + \mathbf{H}(\mathbf{x}_k)\mathbf{U}_k\tag{38}$$

where  $\mathbf{X}_k$ ,  $\mathbf{F}(\mathbf{x}_k)$ ,  $\mathbf{H}(\mathbf{x}_k)$ ,  $\mathbf{U}_k$  are defined as:

$$\mathbf{X}_k = \begin{bmatrix} \mathbf{x}_k \\ \mathbf{x}_{k+1} \\ \vdots \\ \mathbf{x}_{k+N-1} \end{bmatrix}, \quad \mathbf{U}_k = \begin{bmatrix} \mathbf{u}_k \\ \mathbf{u}_{k+1} \\ \vdots \\ \mathbf{u}_{k+N-1} \end{bmatrix}, \quad \mathbf{F}(\mathbf{x}_k) = \begin{bmatrix} \mathbf{I} \\ \mathbf{\Phi}(\mathbf{x}_k) \\ \vdots \\ \mathbf{\Phi}^{N-1}(\mathbf{x}_k) \end{bmatrix}$$

$$\mathbf{H}(\mathbf{x}_k) = \begin{bmatrix} \mathbf{0} & & & & & \\ & \mathbf{\Gamma} & & & & \\ & \mathbf{\Phi}(\mathbf{x}_k)\mathbf{\Gamma} & \mathbf{0} & & & \\ & \vdots & \vdots & \ddots & \ddots & \\ \mathbf{\Phi}^{N-2}(\mathbf{x}_k)\mathbf{\Gamma} & \mathbf{\Phi}^{N-3}(\mathbf{x}_k)\mathbf{\Gamma} & \dots & \mathbf{\Gamma} & \mathbf{0} & \end{bmatrix}$$

And the terminal state at the end of the prediction horizon is defined as

$$\mathbf{x}_{k+N} = \mathbf{\Phi}^N(\mathbf{x}_k)\mathbf{x}_k + \bar{\mathbf{\Gamma}}(\mathbf{x}_k)\mathbf{U}_k \quad (39)$$

Where

$$\bar{\mathbf{\Gamma}}(\mathbf{x}_k) = \begin{bmatrix} \mathbf{\Phi}^{N-1}(\mathbf{x}_k)\mathbf{\Gamma} & \mathbf{\Phi}^{N-2}(\mathbf{x}_k)\mathbf{\Gamma} & \dots & \mathbf{\Phi}(\mathbf{x}_k)\mathbf{\Gamma} & \mathbf{\Gamma} \end{bmatrix}$$

The guidance commands are obtained as the solution to the minimization of the finite-horizon linear quadratic tracking cost function with free-final state subject to state and input constraints. The cost function is given as

$$J(\mathbf{x}_k, \mathbf{x}_k^r, \mathbf{u}_k) = \sum_{j=0}^{N-1} \left[ (\mathbf{x}_{k+j} - \mathbf{x}_{k+j}^r)^T \mathbf{Q}(\mathbf{x}_{k+j} - \mathbf{x}_{k+j}^r) + \mathbf{u}_{k+j}^T \mathbf{R} \mathbf{u}_{k+j} \right] + (\mathbf{x}_{k+N} - \mathbf{x}_{k+N}^r)^T \mathbf{Q}_f (\mathbf{x}_{k+N} - \mathbf{x}_{k+N}^r) \quad (40)$$

where  $\mathbf{x}_k^r \in \mathcal{R}^n$  in general denotes reference trajectory at the  $k^{\text{th}}$  time step. The above cost function can be placed into batch form for the SDC system as

$$J(\mathbf{x}_k, \mathbf{x}_k^r, \mathbf{u}_k) = (\mathbf{X}_k - \mathbf{X}_k^r)^T \bar{\mathbf{Q}}_N (\mathbf{X}_k - \mathbf{X}_k^r) + \mathbf{U}_k^T \bar{\mathbf{R}}_N \mathbf{U}_k + (\mathbf{x}_{k+N} - \mathbf{x}_{k+N}^r)^T \mathbf{Q}_f (\mathbf{x}_{k+N} - \mathbf{x}_{k+N}^r)$$

where

$$\mathbf{X}_k^r = \begin{bmatrix} \mathbf{x}_k^r \\ \mathbf{x}_{k+1}^r \\ \vdots \\ \mathbf{x}_{k+N-1}^r \end{bmatrix}$$

is the batch form of the reference trajectory over the N-step horizon and  $\bar{\mathbf{Q}}_N = \text{diag}\{\mathbf{Q}, \dots, \mathbf{Q}\}$  and  $\bar{\mathbf{R}}_N = \text{diag}\{\mathbf{R}, \dots, \mathbf{R}\}$  are block diagonal matrices consisting of the state and input weighting matrices respectively.

After proper substitution, the objective function can be rewritten in a quadratic-like form as

$$\begin{aligned}
J(\mathbf{x}_k, \mathbf{x}_k^r, \mathbf{u}_k) = & \mathbf{U}_k^T \left( \mathbf{H}(\mathbf{x}_k)^T \bar{\mathbf{Q}}_N \mathbf{H}(\mathbf{x}_k) + \bar{\mathbf{R}}_N + \bar{\mathbf{\Gamma}}(\mathbf{x}_k)^T \mathbf{Q}_f \bar{\mathbf{\Gamma}}(\mathbf{x}_k) \right) \mathbf{U}_k \\
& + 2 \left[ \left( \mathbf{F}(\mathbf{x}_k) \mathbf{x}_k - \mathbf{X}_k^r \right)^T \bar{\mathbf{Q}}_N \mathbf{H}(\mathbf{x}_k) + \left( \Phi(\mathbf{x}_k)^N \mathbf{x}_k - \mathbf{x}_{k+N}^r \right)^T \mathbf{Q}_f \bar{\mathbf{\Gamma}}(\mathbf{x}_k) \right] \mathbf{U}_k \\
& + \left( \mathbf{F}(\mathbf{x}_k) \mathbf{x}_k - \mathbf{X}_k^r \right)^T \bar{\mathbf{Q}}_N \left( \mathbf{F}(\mathbf{x}_k) \mathbf{x}_k - \mathbf{X}_k^r \right) \\
& + \left( \Phi(\mathbf{x}_k)^N \mathbf{x}_k - \mathbf{x}_{k+N}^r \right)^T \mathbf{Q}_f \left( \Phi(\mathbf{x}_k)^N \mathbf{x}_k - \mathbf{x}_{k+N}^r \right)
\end{aligned} \tag{41}$$

The objective function depends on the current state of the system and is calculated at the beginning of every sample interval. Also, the weight matrix for the final state,  $\mathbf{Q}_f$  is obtained by solving the state-dependent discrete algebraic Riccati equation (SDDARE) at time instant  $k$ . The SDDARE is of the form

$$\mathbf{P}(\mathbf{x}_k) = \Phi(\mathbf{x}_k)^T \mathbf{P}(\mathbf{x}_k) \Phi(\mathbf{x}_k) - \Phi(\mathbf{x}_k)^T \mathbf{P}(\mathbf{x}_k) \Gamma \left( \mathbf{R} + \Gamma^T \mathbf{P}(\mathbf{x}_k) \Gamma \right)^{-1} \Gamma^T \mathbf{P}(\mathbf{x}_k) \Phi(\mathbf{x}_k) + \mathbf{Q} \tag{42}$$

Where  $\mathbf{Q}_f$  is set equal to the solution  $\mathbf{P}(\mathbf{x}_k)$ . The solution to equation 42 is found at each sample instant  $k$ .

#### 4.3.1 Input and State Constraints

The constraints are used in this formulation to ensure that the tracking capabilities of the system are feasible in simulation and in practice. The nonlinear kinematic equations for the vehicle enable constraints to be placed on the heading angle turn rate and acceleration of the vehicle without added complication. Overall, the form of the kinematics chosen enables us to place constraints on the system's velocity, heading angle, its turn rate and its forward acceleration. In general, the input and state constraints are represented by the following inequalities

$$\begin{aligned}
\mathbf{u}_{lb} & \leq \mathbf{u}_{k+j} \leq \mathbf{u}_{ub}, \quad j = 0, 1, \dots, N-1 \\
\mathbf{g}_{lb} & \leq \mathbf{G} \mathbf{x}_{k+j} \leq \mathbf{g}_{ub}, \quad j = 0, 1, \dots, N
\end{aligned} \tag{43}$$

where the subscripts  $lb$  and  $ub$  denote the lower and upper bounds respectively of the constraints and the matrix  $\mathbf{G}$  is considered as an output matrix for the states that are constrained. The constraints can be placed in batch form as

$$\begin{bmatrix} \mathbf{u}_{lb} \\ \mathbf{u}_{lb} \\ \vdots \\ \mathbf{u}_{lb} \end{bmatrix} \leq \mathbf{U}_k \leq \begin{bmatrix} \mathbf{u}_{ub} \\ \mathbf{u}_{ub} \\ \vdots \\ \mathbf{u}_{ub} \end{bmatrix}, \quad \begin{bmatrix} \mathbf{g}_{lb} \\ \mathbf{g}_{lb} \\ \vdots \\ \mathbf{g}_{lb} \end{bmatrix} \leq \bar{\mathbf{G}}_N (\mathbf{F} \mathbf{x}_k + \mathbf{H} \mathbf{U}_k) \leq \begin{bmatrix} \mathbf{g}_{ub} \\ \mathbf{g}_{ub} \\ \vdots \\ \mathbf{g}_{ub} \end{bmatrix} \tag{44}$$

where  $\bar{\mathbf{G}}_N = \text{diag}\{G, \dots, G\}$  is a block diagonal matrix formed by the output matrix  $\mathbf{G}$ . Then, the constraints of the system need to be incorporated into a single matrix equation of the form

$$\mathbf{M}(\mathbf{x}_k)\mathbf{U}_k \leq \mathbf{Y}(\mathbf{x}_k) \quad (45)$$

Where

$$\mathbf{M}(\mathbf{x}_k) = \begin{bmatrix} \mathbf{M}_U \\ \bar{\mathbf{G}}_N \mathbf{H}(\mathbf{x}_k) \end{bmatrix}, \quad \mathbf{Y}(\mathbf{x}_k) = \begin{bmatrix} \mathbf{U}_b \\ \mathbf{g} - \bar{\mathbf{G}}_N \mathbf{F}(\mathbf{x}_k) \end{bmatrix}$$

and

$$\mathbf{M}_U = \begin{bmatrix} \begin{bmatrix} \mathbf{I}_{m \times m} \\ -\mathbf{I}_{m \times m} \end{bmatrix} & & & & \\ & \begin{bmatrix} \mathbf{I}_{m \times m} \\ -\mathbf{I}_{m \times m} \end{bmatrix} & & & \\ & & \ddots & & \\ & & & \begin{bmatrix} \mathbf{I}_{m \times m} \\ -\mathbf{I}_{m \times m} \end{bmatrix} & \end{bmatrix}$$

where  $\mathbf{I}_{m \times m}$  is the  $m \times m$  identity matrix. Also,

$$\mathbf{U}_b = \begin{bmatrix} \begin{bmatrix} \mathbf{u}_{ub} \\ -\mathbf{u}_{lb} \end{bmatrix} \\ \begin{bmatrix} \mathbf{u}_{ub} \\ -\mathbf{u}_{lb} \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \mathbf{u}_{ub} \\ -\mathbf{u}_{lb} \end{bmatrix} \end{bmatrix}, \quad \mathbf{g} = \begin{bmatrix} \begin{bmatrix} \mathbf{g}_{ub} \\ -\mathbf{g}_{lb} \end{bmatrix} \\ \begin{bmatrix} \mathbf{g}_{ub} \\ -\mathbf{g}_{lb} \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \mathbf{g}_{ub} \\ -\mathbf{g}_{lb} \end{bmatrix} \end{bmatrix}$$

The guidance commands are synthesized by minimizing the quadratic cost function in equation 41, subject to the input and state constraints in equation 45 using quadratic programming.

#### 4.3.2 Guidance Command Synthesis Using the Linear Matrix Inequality (LMI) Form

For readability with the following LMI formulation, the notation for state dependence is changed from using brackets to a subscript, in other words  $(\cdot)(\mathbf{x}_k)$  will be cast as  $(\cdot)_k$ . First, note that the quadratic-like cost function in equation 41 can be decomposed into the following form

$$J(\mathbf{x}_k, \mathbf{x}_k^r, \mathbf{u}_k) = J_1(\mathbf{x}_k, \mathbf{x}_k^r, \mathbf{u}_k) + J_2(\mathbf{x}_k, \mathbf{x}_k^r, \mathbf{u}_k) \quad (46)$$

Where

Approved for public release; distribution is unlimited.

$$J_1(\mathbf{x}_k, \mathbf{x}_k^r, \mathbf{u}_k) = \mathbf{U}_k^T \mathbf{W}_k \mathbf{U}_k + \boldsymbol{\omega}_k^T \mathbf{U}_k + [\mathbf{F}_k \mathbf{x}_k - \mathbf{X}_k^r]^T \bar{\mathbf{Q}}_{Nk} [\mathbf{F}_k \mathbf{x}_k - \mathbf{X}_k^r]$$

$$J_2(\mathbf{x}_k, \mathbf{x}_k^r, \mathbf{u}_k) = (\Phi_k^N \mathbf{x}_k + \bar{\Gamma}_k \mathbf{U}_k - \mathbf{x}_{k+N}^r)^T \mathbf{Q}_{fk} (\Phi_k^N \mathbf{x}_k + \bar{\Gamma}_k \mathbf{U}_k - \mathbf{x}_{k+N}^r)$$

and  $\mathbf{W}_k$  and  $\boldsymbol{\omega}_k$  are defined as

$$\mathbf{W}_k = \mathbf{H}_k^T \bar{\mathbf{Q}}_{Nk} \mathbf{H}_k + \bar{\mathbf{R}}_N$$

$$\boldsymbol{\omega}_k = 2\mathbf{H}_k^T \bar{\mathbf{Q}}_{Nk}^T [\mathbf{F}_k \mathbf{x}_k - \mathbf{X}_k^r]$$

Then, for each  $\mathbf{x}_k$ , there must exist a set of  $(\mathbf{Q}_{fk}, \mathbf{Q}_k, \mathbf{R}_k, \mathbf{U}_k)$  that satisfy the following conditions

$$J_1(\mathbf{x}_k, \mathbf{x}_k^r, \mathbf{u}_k) \leq \gamma_1 \quad (47)$$

$$J_2(\mathbf{x}_k, \mathbf{x}_k^r, \mathbf{u}_k) \leq \gamma_2 \quad (48)$$

$$\begin{bmatrix} \gamma_1 - 2\mathbf{x}_k^T \mathbf{F}_k^T \bar{\mathbf{Q}}_{Nk} \mathbf{H}_k \mathbf{U}_k - \mathbf{x}_k^T \mathbf{F}_k^T \bar{\mathbf{Q}}_{Nk} \mathbf{F}_k \mathbf{x}_k & \mathbf{U}_k^T \\ \mathbf{U}_k & (\mathbf{H}_k^T \bar{\mathbf{Q}}_{Nk} \mathbf{H}_k + \bar{\mathbf{R}}_{Nk})^{-1} \end{bmatrix} \geq 0 \quad (49)$$

$$\begin{bmatrix} \gamma_2 & [\Phi_k^N \mathbf{x}_k + \bar{\Gamma}_k \mathbf{U}_k]^T \\ \Phi_k^N \mathbf{x}_k + \bar{\Gamma}_k \mathbf{U}_k & \mathbf{Q}_{fk}^{-1} \end{bmatrix} \geq 0 \quad (50)$$

$$\begin{bmatrix} \mathbf{Q}_{fk}^{-1} & (\Phi_k \mathbf{Q}_{fk} - \Gamma \mathbf{Y}_k)^T & (\sqrt{\mathbf{Q}_k} \mathbf{Q}_{fk}^{-1})^T & (\sqrt{\mathbf{R}_k} \mathbf{Y}_k)^T \\ \Phi_k \mathbf{Q}_{fk} - \Gamma \mathbf{Y}_k & \mathbf{Q}_{fk}^{-1} & \mathbf{0}_{n \times n} & \mathbf{0}_{n \times n} \\ \sqrt{\mathbf{Q}_k} \mathbf{Q}_{fk} & \mathbf{0}_{n \times n} & \mathbf{I}_{n \times n} & \mathbf{0}_{n \times n} \\ \sqrt{\mathbf{R}_k} \mathbf{Y}_k & \mathbf{0}_{n \times n} & \mathbf{0}_{n \times n} & \mathbf{I}_{n \times n} \end{bmatrix} \geq 0 \quad (51)$$

$$\mathbf{M}(\mathbf{x}_k) \mathbf{U}_k \leq \mathbf{Y}(\mathbf{x}_k) \quad (52)$$

where  $\mathbf{Y}_k = \mathbf{K}_k \mathbf{Q}_{fk}^{-1}$ , and  $\mathbf{I}_{n \times n}$ ,  $\mathbf{0}_{n \times n}$  denote the  $n \times n$  identity matrix and  $n \times n$  null matrix respectively. The set of LMIs given above are used to establish the feasibility of the control design and are important for proving stability. When a suitable set of  $(\mathbf{Q}_{fk}, \mathbf{Q}_k, \mathbf{R}_k)$  satisfying the above conditions are obtained, the input over the N-step horizon,  $\mathbf{U}_K^*$ , can then be found through the following quadratic programming problem:



$$\mathbf{U}_k^* = \min_{\mathbf{U}_k} \gamma_1 + \gamma_2$$

subject to the constraints in equations 49-52. And upon solving the quadratic programming problem,  $\mathbf{u}_k^*$  can be obtained by

$$\mathbf{u}_k^* = [\mathbf{I}_{m \times m} \ \mathbf{0}_{m \times m} \ \cdots \ \mathbf{0}_{m \times m}] \mathbf{U}_k^* \quad (53)$$

which will extract the guidance commands corresponding to the  $k^{\text{th}}$  time step to the system in equation 37. The stability of the closed loop system using the SDC-based NMPC guidance law is given in [94].

#### 4.4 Cooperative Control of Multiple Vehicles

The strategy for employing multiple mobile robots cooperatively within the framework of this project is discussed in this section. The objective is to demonstrate the ability of the numerical navigation function (described previously) to be utilized by multiple vehicles in the same environment. The navigation function algorithm can represent the operational environment of the vehicles and can form path plans leading to the goal from any point in the obstacle free space. Hence, only a single potential field needs to be generated for the vehicles to use and it can incorporate each of their sensed information. The fact that a single potential field can be used for each vehicle can conceivably decrease the computational burden of the process.

The cooperative control policy described in this chapter will handle tasks concerning vehicle aggregation and social foraging. For cooperative aggregation tasks, the objective is to bring the vehicles together at a set location while avoiding collisions among the group. And for social foraging, the objective becomes having the robots find ‘conflict-free’ trajectories to areas of interest while avoiding collisions with hazards present in the environment [79,80]. The cooperative control policy will include different components designed to accomplish each task.

##### 4.4.1 Artificial Potential Function for ‘Conflict-Free’ Trajectory Synthesis

The cooperative trajectory synthesis for multiple vehicles using artificial potential functions (APF) is considered in this section. Assume that there are  $N$  mobile robots present in a given environment and  $\mathbf{r}^i \in \mathfrak{R}^2$  is the position vector of each vehicle and  $\mathbf{r}^T = [\mathbf{r}^{1,T}, \dots, \mathbf{r}^{N,T}]$  is the vector containing the position vectors for the group. Then, a steering command for the  $i^{\text{th}}$  robot, using the APF design can be given by

$$\dot{\mathbf{r}}_{pot}^i = -\nabla_{\mathbf{r}^i} \mathbf{J}(\mathbf{r}) \quad (54)$$

where  $\mathbf{J}(\mathbf{r})$  is a composite potential function consisting of the cooperative potentials, constraints, the navigation function potential as well as other suitable potentials employed for the tasks specified.

The composite potential function has the following form

Approved for public release; distribution is unlimited.

$$\mathbf{J}(\mathbf{r}) = \delta \left( \mathbf{J}_{coop}(\mathbf{r}) + \mathbf{J}_{goal}(\mathbf{r}) \right) + (1 - \delta) \mathbf{J}_{NF}(\mathbf{r}) + \mathbf{J}_{rep}(\mathbf{r}) \quad (55)$$

where  $\mathbf{J}_{coop}(\mathbf{r})$ ,  $\mathbf{J}_{goal}(\mathbf{r})$ ,  $\mathbf{J}_{NF}(\mathbf{r})$  and  $\mathbf{J}_{rep}(\mathbf{r})$  denote the potential functions for swarm aggregation, goal position, navigation function, and collision avoidance respectively. The scalar quantity  $\delta$  is user defined based on the cooperative task assigned to the group, and is set to either 1 for swarm aggregation at a goal position or 0 for social foraging using the navigation function. Notice that regardless of the choice of  $\delta$ , the collision avoidance potential is always present to ensure the robots do not collide.

Then, the steering command for the  $i^{th}$  robot using the composite potential function in equation 55 is given by

$$\begin{aligned} \dot{\mathbf{r}}_{pot}^i &= \delta \left( -\nabla_{\mathbf{r}^i} \mathbf{J}_{coop}(\mathbf{r}) - \nabla_{\mathbf{r}^i} \mathbf{J}_{goal}(\mathbf{r}) \right) + (1 - \delta) \left( -\nabla_{\mathbf{r}^i} \mathbf{J}_{NF}(\mathbf{r}) - \nabla_{\mathbf{r}^i} \mathbf{J}_{rep}(\mathbf{r}) \right) \\ &= \delta \left( \dot{\mathbf{r}}_{coop}^i + \dot{\mathbf{r}}_{goal}^i \right) + (1 - \delta) \dot{\mathbf{r}}_{NF}^i + \dot{\mathbf{r}}_{rep}^i \end{aligned} \quad (56)$$

The design of each component in equations 54 and 55 are detailed in the following subsections.

#### 4.4.2 Swarm Aggregation APF Design

This subsection will detail a suitable potential function used for swarm aggregations and is based on the work presented in references [79,80]. The APF design has the following form

$$\mathbf{J}_{coop}(\mathbf{r}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^N \mathbf{J}_{ij}(\|\mathbf{r}^i - \mathbf{r}^j\|) \quad (57)$$

making it a function of the norm of the relative position of the respective vehicles. The potential function given in references [79, 80] possesses both an attractive and a repulsive component. The potential function is designed to simultaneously draw the vehicles in the group together while keeping them apart at a safe distance. The cooperative potential function is defined as

$$\mathbf{J}_{coop}(\mathbf{r}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^N \left[ \frac{a}{2} \|\mathbf{r}^i - \mathbf{r}^j\|^2 + \frac{bc}{2} \exp\left(-\frac{\|\mathbf{r}^i - \mathbf{r}^j\|^2}{c}\right) \right] \quad (58)$$

where  $a, b, c \in \mathfrak{R}^+$  are positive scalar shaping parameters for the cooperative potential.  $a$  and  $b$  are used to set the strength of their respective components while  $c$  sets the region of influence of the repulsive component. The steering command for the  $i^{th}$  robot generated with equation 56 is

$$\dot{\mathbf{r}}_{coop}^i = - \sum_{j=1, j \neq i}^N \left[ \|\mathbf{r}^i - \mathbf{r}^j\| \left( a - b \exp\left(-\frac{\|\mathbf{r}^i - \mathbf{r}^j\|^2}{c}\right) \right) \right] \quad (59)$$

The aggregation component of the composite potential function also contains an attractive potential defined at an objective gathering point. This enables a mission planner to designate where in the environment the group should converge. The attractive potential function has the following form

$$\mathbf{J}_{goal}(\mathbf{r}) = \sum_{i=1}^N \mathbf{J}_{i,goal}(\|\mathbf{r}^i - \mathbf{r}^{goal}\|)$$

where  $\mathbf{r}^{goal} \in \mathfrak{R}^2$  is the position vector of the gathering point. The attractive potential for the gathering point is given as a parabolic well defined as

$$\mathbf{J}_{goal}(\mathbf{r}) = \sum_{i=1}^N \left[ k_a \|\mathbf{r}^i - \mathbf{r}^{goal}\|^2 \right] \quad (60)$$

where  $k_a \in \mathfrak{R}^+$  is a design parameter to adjust the strength of the potential function. The steering command for the  $i^{th}$  robot generated by the attractive potential is

$$\dot{\mathbf{r}}_{goal}^i = -k_a \|\mathbf{r}^i - \mathbf{r}^{goal}\| \quad (61)$$

The combination of the steering commands in equations 59 and 61 allows for the collective to travel together and ultimately gather at a desired location. However, one aspect that was not addressed with this aggregation design is obstacle avoidance. Thus, the aggregation component of the steering commands in equation 58 assumes the robots are working in an obstacle free space. The issue of obstacle avoidance leading to an objective gathering point is addressed by the navigation function in the following section.

#### 4.4.3 Minimum Control Effort Navigation Function for Social Foraging

In short, the task of social foraging refers to having locations in a given environment that are either considered favorable or unfavorable for a group of vehicles to consider. In this project, these regions are defined through the navigation function framework discussed previously. In the navigation function algorithm, the favorable region is set as the goal location and the unfavorable regions are considered as obstacles.

It is assumed that the vehicles share information such as obstacle positions or other constraints with each other. This information is then used to form the navigation function's potential field. Hence, each robot knows the potential levels generated from the navigation function algorithm.

The navigation function algorithm details how to form a numerical potential field in an evenly spaced grid given knowledge of the operational environment. The algorithm, as outlined previously, generates a potential field which is then used to find a path plan using an iterative graph search method (best-first search). So, the reference path is found before the robot has the ability to act. This approach does not allow for a reactive element, i.e. generating steering commands, directly based on the environment. The task then becomes how to use the resulting potential field from the algorithm to generate steering commands allowing a robot to react to its environment. This result will allow the group of mobile robots to navigate a constrained environment towards a desired gathering point.

#### 4.4.4 Central Difference Approximation to NF Gradient

This section will detail how to find the numerical gradient of the navigation function and how it is used within the cooperative control policy. The gradient of the navigation function needs to be generated numerically since it is defined over a discrete representation of the environment. The method chosen to compute the gradient of the navigation function is the Central-Difference method [90].

In order to use the grid-based potential function and the central-difference approximation of the gradient, the position of a robot needs to be defined in terms of the evenly spaced grid. With this in mind, let  $\tilde{\mathbf{r}}^i \in \mathfrak{R}^2$  denote the approximate position of the  $i^{\text{th}}$  robot in the evenly spaced grid. And let  $\tilde{x}^i$  and  $\tilde{y}^i$  be the approximate position components of  $\tilde{\mathbf{r}}^i$ , i.e.  $\tilde{\mathbf{r}}^i = [\tilde{x}^i, \tilde{y}^i]^T$ . Also, let  $\Delta x$  and  $\Delta y$  denote the spacing between grid points in the  $x$  and  $y$  directions respectively. And since the grid is evenly spaced,  $\Delta x = \Delta y$ . Then, the components of the numerical gradient, computed with the central-difference method, is determined in terms of the robot's approximate position in the grid as

$$\begin{aligned} \frac{\partial \mathbf{J}_{NF}(\tilde{\mathbf{r}}^i)}{\partial x} &= \frac{\partial \mathbf{J}_{NF}([\tilde{x}^i, \tilde{y}^i]^T)}{\partial x} \approx \frac{\mathbf{J}_{NF}([\tilde{x}^i + \Delta x, \tilde{y}^i]^T) - \mathbf{J}_{NF}([\tilde{x}^i - \Delta x, \tilde{y}^i]^T)}{2\Delta x} \\ \frac{\partial \mathbf{J}_{NF}(\tilde{\mathbf{r}}^i)}{\partial y} &= \frac{\partial \mathbf{J}_{NF}([\tilde{x}^i, \tilde{y}^i]^T)}{\partial y} \approx \frac{\mathbf{J}_{NF}([\tilde{x}^i, \tilde{y}^i + \Delta y]^T) - \mathbf{J}_{NF}([\tilde{x}^i, \tilde{y}^i - \Delta y]^T)}{2\Delta y} \end{aligned}$$

And the steering command based on the numerical gradient of the navigation function can be set as

$$\mathbf{r}_{NF}^i = \begin{bmatrix} -\frac{\partial \mathbf{J}_{NF}(\tilde{\mathbf{r}}^i)}{\partial x} \\ -\frac{\partial \mathbf{J}_{NF}(\tilde{\mathbf{r}}^i)}{\partial y} \end{bmatrix} \quad (62)$$

#### 4.4.5 APF for Collision Avoidance

Within the cooperative framework, there is an extra collision avoidance potential denoted as  $\mathbf{r}_{rep}^i$ . The collision avoidance term consists of only a repulsive term designed to prevent the robots from colliding with one another. Note that this repulsive potential function may also be used to avoid collisions with obstacles in the environment. The collision avoidance potential is of the form

$$\mathbf{J}_{rep}(\mathbf{r}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^N \mathbf{J}_{ij}(\|\mathbf{r}^i - \mathbf{r}^j\|) \quad (63)$$

which is similar to the cooperative potential form in equation 58 and is a function of the position information of the robots. The repulsive potential chosen for the results covered is consistent with the repulsive potential component in equation 59 given by

$$\mathbf{J}_{rep}(\mathbf{r}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^N \left[ \frac{b_r c_r}{2} \exp\left(-\frac{\|\mathbf{r}^i - \mathbf{r}^j\|^2}{c_r}\right) \right] \quad (64)$$

where the parameters  $b_r, c_r \in \mathfrak{R}$  are used to respectively set the strength and region of influence of the repulsive potential. The subscript  $r$  is used to distinguish the parameters in equation 64 from the parameters in equation 63. The resulting steering command for the  $i^{th}$  vehicle is given by

$$\dot{\mathbf{r}}_{rep}^i = \sum_{j=1, j \neq i}^N \left[ b_r \|\mathbf{r}^i - \mathbf{r}^j\| \exp\left(-\frac{\|\mathbf{r}^i - \mathbf{r}^j\|^2}{c_r}\right) \right] \quad (65)$$

#### 4.4.6 Control Law Design for Cooperation

Typical control designs with artificial potential functions, including cooperative control designs, are applied to integrator dynamic systems (or sometimes double integrator). The goal of this work is to apply the control design to a system with the nonlinear vehicle kinematics, as described previously. Thus, the kinematics for the  $i^{th}$  vehicle is

$$\begin{aligned} \dot{x}^i &= v^i \cos(\psi^i) \\ \dot{y}^i &= v^i \sin(\psi^i) \\ \dot{\psi}^i &= \alpha_1 (\psi_c^i - \psi^i) \\ \dot{v}^i &= \alpha_2 (v_c^i - v^i) \end{aligned} \quad (66)$$

Where  $\alpha_1, \alpha_2 \in \mathfrak{R}$ , are positive proportional control gains and  $\psi_c^i, v_c^i$  are the guidance commands for the  $i^{\text{th}}$  vehicle's heading angle and velocity. The control design affects the robot's heading angle turn rate and forward acceleration, which are proportional control laws designed to drive the heading angle and velocity to the commanded values which are obtained through the steering commands from the APF described in the previous section.

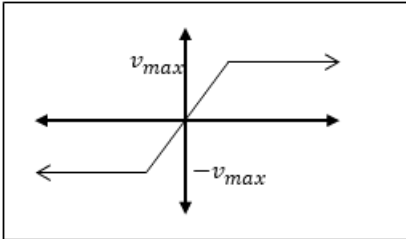
Let  $\dot{\mathbf{r}}_{pot,x}^i$  and  $\dot{\mathbf{r}}_{pot,y}^i$  denote the  $x$  and  $y$  directional components of the gradient terms that make up the steering command from equation 62. Then, the heading angle guidance command,  $\psi_c^i$ , for the  $i^{\text{th}}$  robot is defined as

$$\psi_c^i = \tan^{-1} \left( \frac{\dot{\mathbf{r}}_{pot,y}^i}{\dot{\mathbf{r}}_{pot,x}^i} \right) \quad (67)$$

And the velocity guidance command for the robot is

$$v_c^i = \|\dot{\mathbf{r}}_{pot}^i\| \quad (68)$$

which is the 2-norm of the steering command vector. It is possible that the velocity command may give an unfeasible value. Therefore, the velocity command term is saturated as

$$v_c^i = \text{sat}(v_c^i) =$$


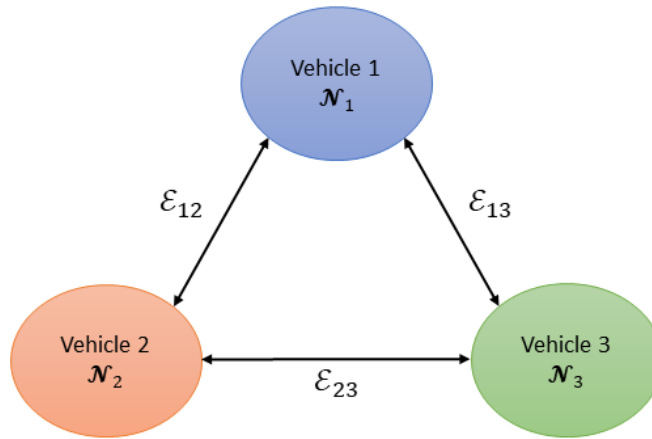
This implies that  $|v_c^i| \leq v_{max}$  where the saturation limit,  $v_{max}$  is a user defined quantity for the forward velocity of the vehicle. This signifies that the steering commands will mainly influence the heading angle of the robot. Thus, the vehicles can continue to travel forward and adjust their heading angles as needed while traversing the environment with the guidance commands generated through the cooperative control policy.

#### 4.4.7 Inter-Vehicle Communication

For this, it is assumed that there is full communication among the vehicles. A graph representation is used to illustrate the communication protocol involved in the simulations. In graph theory, a graph is described as a pair  $G = (N, E)$ , with  $N = \{N_1, \dots, N_N\}$ , a set of  $N$  nodes and  $E$  as a set of edges. The elements of  $E$  are denoted as  $(N_i, N_j)$  which is termed as an edge from node  $N_i$  to node  $N_j$  [78]. In the setting of cooperative control, the nodes represent the vehicles and the edges represent the communication of information.

An illustration of the communication graph for the cooperative control examples with three mobile robots is given in Figure 10. In the examples given, it is assumed that the navigation function information as well as the position vectors are shared among the robots through the communication protocol.

Also, the communication graph is assumed to be undirected. Making the graph undirected implies that the information shared goes in both directions and has equal importance, i.e. node  $N_i$  shares its information with node  $N_j$  and vice-versa [78]. Therefore, in Figure 10, the edges are denoted as  $E_{ij} = E_{ji}$ , which represents an undirected communication link between nodes  $N_i$  and  $N_j$ ,



**Figure 10: Communication protocol**

#### 4.5 Cooperative Control in the Presence of Measurement Time-Delays

A set of dynamical systems is said to be synchronized when there is a complete match of configuration variables describing each system. Synchronization when pertaining to a spacecraft formation refers to the state when all the spacecraft possess a common attitude. [2] formulates the spacecraft formation problem in a Lagrangian framework and proposes a decentralized tracking

law to achieve synchronization. [84] proposes a leader-follower configuration and solves the synchronization problem by designing a control strategy to make all the spacecraft track the leader (reference) spacecraft. [13] solves the attitude synchronization problem for a wider class of communication topologies (directed graphs).

In practical applications, the flow of information between spacecraft is delayed. This is often attributed to the delay in receiving data from the other spacecraft or processing of data. [69] considers the problem of synchronization of a spacecraft formation subject to constant communication delays and solves the same by using feedback linearization to track a reference trajectory and approaches the problem by proposing a backstepping controller that tracks a virtual angular velocity to achieve synchronization. Alternately, a decentralized control approach for the same problem is used by appealing to the geometric structure of the configuration space of the spacecraft formation. Time-varying communication delays are also considered along with delays in self-tracking control parts. This renders the dynamics nonlinear and the authors resort to constructing linear filters to derive an output feedback law.

This project considers a vehicle formation problem wherein there are asymmetric, bounded and time-varying delays in the communication links between the vehicles while the feedback from the vehicle's own states is instantaneous. A controller is proposed for the delayed system to achieve consensus, followed by stability analysis of the system using both time domain and frequency domain approaches. The subsequent development considers the attitude dynamics of spacecraft but we show that this can be essentially reduced to a double integrator formulation which is then amenable to be used in the cooperative consensus type problems. The analysis of the time delays is for the general dynamical systems in the double integrator form. This includes translational as well as rotational dynamics.

#### 4.5.1 Modified Rodrigues Parameters

The multiple spacecraft attitude consensus problem is considered with the states of each spacecraft being the Modified Rodrigues Parameters (MRP)  $\boldsymbol{\sigma}$  and angular velocity  $\boldsymbol{\omega}$  in the body frame. The attitude kinematics and the dynamics is assumed to be identical for all spacecraft in formation and is governed by the following nonlinear ordinary differential equations:

$$\dot{\boldsymbol{\sigma}}(t) = \mathbf{P}(\boldsymbol{\sigma})\boldsymbol{\omega}(t) \quad (69)$$

$$\dot{\boldsymbol{\omega}}(t) = \mathbf{J}^{-1} \left( -[\boldsymbol{\omega}(t)]\mathbf{J}\boldsymbol{\omega}(t) + \boldsymbol{\tau} \right) \quad (70)$$

where  $\mathbf{P}(\boldsymbol{\sigma}) = \frac{1}{4} \left( 2[\tilde{\boldsymbol{\sigma}}(t)] + 2\boldsymbol{\sigma}(t)\boldsymbol{\sigma}^T(t) + (1 - \boldsymbol{\sigma}^T(t)\boldsymbol{\sigma}(t))\mathbf{I}_{3 \times 3} \right)$ ,  $[\tilde{\mathbf{a}}] = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$

$\mathbf{J}$  is the symmetric moment of Inertia matrix and  $\boldsymbol{\tau}$  is the control torque.



The MRP is a vector defined as  $\boldsymbol{\sigma} = \hat{\mathbf{n}} \tan\left(\frac{\Phi}{4}\right)$  where  $\hat{\mathbf{n}}$  is the principal axis and  $\Phi$  is the principal angle as given by Euler's rotation theorem. The MRPs yield a unique representation for the attitude of the spacecraft for all principal rotations that lie in the interval  $[0, 2\pi)$ . At  $\Phi = 2\pi$ , a singularity is encountered that is typically addressed using the shadow MRP set. In this paper, it is assumed that the rotations are limited to principal rotation angles less than  $2\pi$ .

#### 4.5.2 Feedback Linearization of the Attitude Dynamics

The system represented by equations 69 and 70 is rewritten into a compact second order nonlinear ode:

$$\mathbf{M}(\boldsymbol{\sigma})\ddot{\boldsymbol{\sigma}} + \mathbf{C}(\boldsymbol{\sigma}, \dot{\boldsymbol{\sigma}})\dot{\boldsymbol{\sigma}} = \mathbf{u} \quad (71)$$

Let  $\boldsymbol{\sigma}_1 = \boldsymbol{\sigma}$  and  $\boldsymbol{\sigma}_2 = \dot{\boldsymbol{\sigma}}$ . Performing the transformations  $\boldsymbol{\omega} = \mathbf{P}(\boldsymbol{\sigma})^{-1}\dot{\boldsymbol{\sigma}}$  and  $\mathbf{P}(\boldsymbol{\sigma})^{-1}\boldsymbol{\tau} = \mathbf{u}$  the dynamics can be expressed as

$$\dot{\boldsymbol{\sigma}}_1 = \boldsymbol{\sigma}_2 \quad (72)$$

$$\dot{\boldsymbol{\sigma}}_2 = \mathbf{M}(\boldsymbol{\sigma}_1)^{-1}(-\mathbf{C}(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)\boldsymbol{\sigma}_2 + \mathbf{u}) \quad (73)$$

where  $\mathbf{M}(\boldsymbol{\sigma}_1) = \mathbf{P}(\boldsymbol{\sigma}_1)^{-1}\mathbf{J}\mathbf{P}(\boldsymbol{\sigma}_1)$  and  $\mathbf{C}(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2) = \mathbf{P}(\boldsymbol{\sigma}_1)^{-1}\left(\mathbf{J}\dot{\mathbf{P}}(\boldsymbol{\sigma}_1) + [\mathbf{P}(\boldsymbol{\sigma}_1)\boldsymbol{\sigma}_2]\mathbf{J}\mathbf{P}(\boldsymbol{\sigma}_1)\right)$ .

Using full state feedback and a straightforward feedback linearization based control law,  $\mathbf{u} = \mathbf{C}(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)\boldsymbol{\sigma}_2 + \mathbf{M}(\boldsymbol{\sigma}_1)\mathbf{v}$ , the above dynamics are further reduced to that of a double integrator as follows.

$$\dot{\boldsymbol{\sigma}}_1 = \boldsymbol{\sigma}_2 \quad (74)$$

$$\dot{\boldsymbol{\sigma}}_2 = \mathbf{v}$$

#### 4.5.3 Consensus of Formation

Now, consider a system of  $N$  vehicles with communication pathways among them forming a strongly connected graph, each of whom is being controlled using the control law  $\mathbf{u}(t)$  described earlier. Thus, the attitude consensus problem for  $N$  nonlinear systems can be posed using the simpler structure derived earlier in equation 74. The assumption is that for the class of vehicles under consideration, such a feedback linearizing control law exists which renders the dynamics in the form expressed in equation 74.

Let  $\mathbf{x} = [\mathbf{x}_1^T \mathbf{x}_2^T]^T$  be a vector such that  $\mathbf{x}_1$  consists of the states of the  $N$  spacecraft while  $\mathbf{x}_2$  consists of their derivatives i.e.  $\mathbf{x}_1 = [{}^1\boldsymbol{\sigma}_1^T \cdots {}^N\boldsymbol{\sigma}_1^T]^T$  and  $\mathbf{x}_2 = [{}^1\boldsymbol{\sigma}_2^T \cdots {}^N\boldsymbol{\sigma}_2^T]^T$ . This vector defines the state of the  $N$  spacecraft system and the combined dynamics can be written as:

$$\begin{aligned}\dot{\mathbf{x}}_1(t) &= \mathbf{x}_2(t) \\ \dot{\mathbf{x}}_2(t) &= \mathbf{u}(t)\end{aligned}\tag{75}$$

where the vector  $\mathbf{u} = \begin{bmatrix} \mathbf{v}^T \dots \mathbf{v}^T \end{bmatrix}^T$  is the control input obtained by cascading the control inputs of all the vehicles.

For the system of N spacecraft described above in equation 75, consensus is said to be achieved when  $\|\sigma_1^i - \sigma_1^j\| \rightarrow 0$  and  $\|\sigma_2^i - \sigma_2^j\| \rightarrow 0 \quad \forall i, j, i \neq j$ .

In the case where the spacecraft exchange state information to their connected neighbors with no delay, the following control law drives the system to consensus [12]. For all the analysis from here on, we will use the compact set of equations in equation 55 for the N connected spacecraft.

$$\mathbf{u}(t) = -\mathbf{L}\mathbf{x}_1(t) - \gamma\mathbf{L}\mathbf{x}_2(t)\tag{76}$$

where  $\mathbf{L} = [l_{ij}]$  is the Laplacian matrix for a given communication topology [12] and  $\gamma > 0$  is a damping gain.

#### 4.5.4 Consensus Control Law

In this paper, the consensus problem is analyzed when the communication is subject to time delays i.e. the state information of the  $i^{\text{th}}$  spacecraft is relayed to the  $j^{\text{th}}$  spacecraft after a delay  $\tau_{ij}$  that satisfies

$$0 \leq \tau_{ij} \leq h_{ij} \quad |\dot{\tau}_{ij}| \leq d_{ij}\tag{77}$$

Clearly the time delays are different along different communication paths (spacecraft pairs) and are time-varying.

To achieve consensus, the following control law is investigated.

$$\mathbf{u}(t) = -\mathbf{x}_1(t) - \gamma\mathbf{x}_2(t) - \sum_{\substack{i=1 \\ i \neq j}}^N \sum_{j=1}^N \overset{ij}{\mathbf{K}} \mathbf{x}_1(t - \tau_{ij}) - \sum_{\substack{i=1 \\ i \neq j}}^N \sum_{j=1}^N \gamma \overset{ij}{\mathbf{K}} \mathbf{x}_2(t - \tau_{ij})\tag{78}$$

For N spacecraft defining the communication topology, then for every pair  $(i, j)$  of spacecraft we have  $l_{ij} \leq 0$  (from the Laplacian matrix),  $\overset{ij}{\mathbf{K}} \in \mathbb{R}^{N \times N}$  and  $\overset{ij}{\mathbf{K}} \otimes \mathbf{I}_{3 \times 3}$  is the coefficient of the

delayed state, where  $\tau_{ij}$  is the time delay in sending information from the  $i^{\text{th}}$  spacecraft to the  $j^{\text{th}}$  spacecraft. The  $ji^{\text{th}}$  element of  ${}^{ij}\mathbf{K}$  is equal to  $l_{ij}$ . All other elements are zero. The Laplacian matrix specifying the communication topology between the three vehicles be given by

$$\begin{bmatrix} 1 & 0 & -1 \\ -\frac{1}{2} & 1 & -\frac{1}{2} \\ 0 & 0 & 0 \end{bmatrix}.$$

Let the delays in the communication channels be  $\tau_{12}$ ,  $\tau_{31}$  and  $\tau_{32}$ . Then, the matrices  ${}^{ij}\mathbf{K}$  are given by

$${}^{12}\mathbf{K} = \begin{bmatrix} 0 & 0 & 0 \\ -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \otimes \mathbf{I}_{3 \times 3}, \quad {}^{31}\mathbf{K} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \otimes \mathbf{I}_{3 \times 3}, \quad \text{and} \quad {}^{32}\mathbf{K} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} \\ 0 & 0 & 0 \end{bmatrix} \otimes \mathbf{I}_{3 \times 3}$$

The closed loop state dynamics are expressed as a linear differential equation with multiple time delays.

$$\dot{\mathbf{x}}(t) = \mathbf{A}_0 \mathbf{x}(t) + \sum_{i=1}^N \sum_{\substack{j=1 \\ i \neq j}}^N {}^{ij} \mathbf{A} \mathbf{x}(t - \tau_{ij}) \quad (79)$$

where  $\mathbf{A}_0 = \begin{bmatrix} \mathbf{0}_{N \times N} & \mathbf{I}_{N \times N} \\ -\mathbf{I}_{N \times N} & -\gamma \mathbf{I}_{N \times N} \end{bmatrix} \otimes \mathbf{I}_{3 \times 3}$  and  ${}^{ij} \mathbf{A} = \begin{bmatrix} \mathbf{0}_{N \times N} & \mathbf{0}_{N \times N} \\ -{}^{ij} \mathbf{K} & -\gamma {}^{ij} \mathbf{K} \end{bmatrix} \otimes \mathbf{I}_{3 \times 3}$

Note that  $\sum_{i=1}^N \sum_{\substack{j=1 \\ i \neq j}}^N {}^{ij} \mathbf{A} = \begin{bmatrix} \mathbf{0}_{N \times N} & \mathbf{0}_{N \times N} \\ \mathbf{A} & \gamma \mathbf{A} \end{bmatrix} \otimes \mathbf{I}_{3 \times 3} = \mathbf{A}_\gamma$  where  $\mathbf{A}$  is the adjacency matrix of the connection topology.

#### 4.5.5 Stability Analysis

##### 4.5.5.1 Time Domain Approach

###### **Lemma 1 [Schur's Complement] [94]**

The following linear matrix inequality (LMI)  $\mathbf{S} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{S}_{22} \end{bmatrix} < 0_\gamma$  where  $\mathbf{S} = \mathbf{S}^T$ , is equivalent to

each of the following conditions:  $\mathbf{S}_{11} < 0, \mathbf{S}_{22} - \mathbf{S}_{12}^T \mathbf{S}_{11}^{-1} \mathbf{S}_{12} < 0, \mathbf{S}_{22} < 0, \mathbf{S}_{11} - \mathbf{S}_{12} \mathbf{S}_{22}^{-1} \mathbf{S}_{12}^T < 0$

###### **Lemma 2 [Jensen's Inequality] [95]**

For any constant matrix,  $P \in \mathbb{R}^{m \times m}$ ,  $P = P^T > 0$ , scalar  $\gamma > 0$ , vector function  $\omega : [0, \gamma] \rightarrow \mathbb{R}^m$  such that the integrations concerned are well defined, then

Approved for public release; distribution is unlimited.

$$\gamma \int_0^\gamma \omega^T(\beta) P \omega(\beta) d\beta \geq \left( \int_0^\gamma \omega(\beta) d\beta \right)^T P \left( \int_0^\gamma \omega(\beta) d\beta \right)$$

The lemmas stated above are key to obtaining an LMI condition for ensuring stability of the desired consensus.

**Theorem 2**

Under the action of the control law, the solutions to the dynamics given by equation 57 converge to consensus if there exist symmetric, positive definite matrices  $Q_{ij} \in \mathbb{R}^{6(n-1) \times 6(n-1)}$ ,  $S_{ij} \in \mathbb{R}^{6(n-1) \times 6(n-1)}$  and positive constant  $\gamma$  such that the following LMIs hold

$$\Psi_1 < \mathbf{0}, \Psi_2 < \mathbf{0}, \Psi_3 < \mathbf{0}$$

where

$$\Psi_1 = \begin{bmatrix} E^T EA_0 + A_0^T E^T E & E^T EA_{12} & E^T EA_{13} & \dots & E^T EA_{nn-1} \\ A_{12}^T E^T E & 0 & 0 & \dots & 0 \\ A_{13}^T E^T E & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A_{nn-1}^T E^T E & 0 & 0 & \dots & 0 \end{bmatrix}$$

$$\Psi_2 = \begin{bmatrix} \sum_{\substack{i=1 \\ i \neq j}}^n \sum_{j=1}^n E^T Q_{ij} E & 0 & \dots & 0 \\ 0 & -E^T Q_{12} E & \vdots & \vdots \\ \vdots & 0 & \ddots & \vdots \\ 0 & \dots & \dots & -E^T Q_{nn-1} E \end{bmatrix}$$

$$\Psi_3 = \begin{bmatrix} \Psi_{11} & \Psi_{12} & \dots & \Psi_{1n} & A_0^T E^T \\ \Psi_{12}^T & \ddots & \Psi_{23} \dots & \vdots & A_{12}^T E^T \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \Psi_{1n}^T & \dots & \dots & \Psi_{nn-1} & A_{nn-1}^T E^T \\ EA_0 & EA_{12} & \ddots & EA_{nn-1} & -\left( \sum_{\substack{i=1 \\ i \neq j}}^n \sum_{j=1}^n h_{ij} S_{ij} \right)^{-1} \end{bmatrix}$$

where

$$E = \begin{bmatrix} [\mathbf{1} - \mathbf{1}_{n-1}] & \mathbf{0} \\ \mathbf{0} & [\mathbf{1} - \mathbf{1}_{n-1}] \end{bmatrix} \otimes \mathbf{I}_{3 \times 3}$$

$$\Psi_{11} = \sum_{i=1}^n \sum_{\substack{j=1 \\ i \neq j}}^n \left( \frac{1-d_{ij}}{h_{ij}} E^T (S_{ij}) E \right), \quad \Psi_{12} = \left( \frac{1-d_{12}}{h_{12}} E^T (S_{12}) E \right),$$

$$\Psi_{22} = \left( -\frac{1-d_{12}}{h_{12}} E^T (S_{12}) E \right), \quad \Psi_{23} = \mathbf{0} \dots \text{ and so on}$$

**Proof:** Please refer to [94]

Note, that finding feasible solutions for the above LMIs is a tedious task considering that the Lyapunov Krasovskii functional (see [94]) is required to be positive definite on an augmented state space while its derivative is required to be negative definite on the actual state space  $\mathbf{x}(t)$  which can't be obtained from augmented state space via an invertible transformation due to the nature of how the time delays appear in the closed loop dynamics. To guarantee feasible solutions, *a leader-follower strategy or a constant set point must be specified to obtain an invertible transformation.* To this end, we also investigate the consensus problem using a frequency domain approach to obtain delay dependent stability criteria.

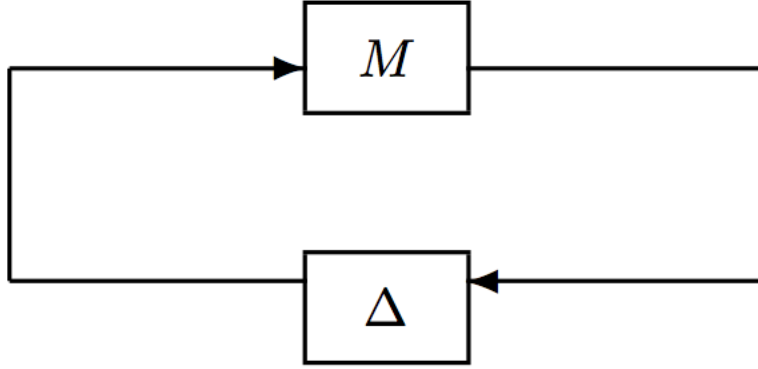
#### 4.5.5.2 Frequency Domain Approach

We adopt a frequency domain approach by making use of the small-gain theorem to show that the system achieves consensus. Before stating the main theorem, we present a few lemmas and the small-gain theorem itself.

Let  $L_2^p$  be the space of  $\square^p$  valued functions  $f : [0, \infty) \rightarrow \square^p$  such that its  $L_2$  norm is bounded, i.e,

$\|f\|_{L_2} = \int_0^\infty f^T(t)f(t)dt < \infty$ , Let  $L_{2e}^p$  be an extended space containing functions that satisfy the above inequality on finite intervals. An operator is defined as a function  $G : L_{2e}^p \rightarrow L_{2e}^q$  with an

induced norm defined as  $\|G\| = \sup \left( \frac{\|G(f)\|}{\|f\|} \right) : \forall f \in L_{2e}^p, f \neq 0$



**Figure 11: Feedback System**

**Lemma 3 [Small-Gain theorem]**

Suppose that  $M$  and  $\Delta$  are both stable systems with finite input-output gains. Let  $\|M\|$  and  $\|\Delta\|$  denote their respective induced norms. Then the feedback system in Figure 11 is stable if  $\|M\Delta\| < 1$

The small-gain theorem provides a sufficient condition for stability of feedback systems. Let us examine the form of equation 57 more closely.

$$\begin{aligned}
 \dot{\mathbf{x}}(t) &= \mathbf{A}_0 \mathbf{x}(t) + \sum_{i=1}^N \sum_{\substack{j=1 \\ i \neq j}}^N \mathbf{A}^{ij} \mathbf{x}(t - \tau_{ij}) \\
 \Rightarrow \dot{\mathbf{x}}(t) &= \mathbf{A}_0 \mathbf{x}(t) + \sum_{i=1}^N \sum_{\substack{j=1 \\ i \neq j}}^N \mathbf{A}(\mathbf{x}(t - \tau_{ij}) - \mathbf{x}(t)) + \sum_{i=1}^N \sum_{\substack{j=1 \\ i \neq j}}^N \mathbf{A} \mathbf{x}(t) \\
 \Rightarrow \dot{\mathbf{x}}(t) &= \mathbf{A}_0 \mathbf{x}(t) + \sum_{i=1}^N \sum_{\substack{j=1 \\ i \neq j}}^N \mathbf{A}(\mathbf{x}(t - \tau_{ij}) - \mathbf{x}(t)) + \mathbf{A}_\gamma \mathbf{x}(t) \\
 \Rightarrow \dot{\mathbf{x}}(t) - \mathbf{A}_\gamma \mathbf{x}(t) - \mathbf{A}_0 \mathbf{x}(t) &= \sum_{i=1}^N \sum_{\substack{j=1 \\ i \neq j}}^N \mathbf{A}(\mathbf{x}(t - \tau_{ij}) - \mathbf{x}(t))
 \end{aligned}$$

Thus, the system of spacecraft in our problem equation 57 can be viewed as a feedback system with the same structure in Figure 11 with the transfer function  $\mathbf{X}(s) = \mathbf{T}(s)\Delta(\mathbf{X}(s))$  where

$$\mathbf{T}(s) = (sI_{6N \times 6N} - \mathbf{A}_0 - \mathbf{A}_\gamma)^{-1} [\mathbf{A}_{12} \mathbf{A}_{13} \dots \mathbf{A}_{m-1}] \quad \text{and} \quad \Delta = \begin{bmatrix} \Delta_{\tau_{12}} & 0 & \dots & 0 \\ 0 & \Delta_{\tau_{13}} & \dots & 0 \\ 0 & \dots & 0 & \Delta_{\tau_{m-1}} \end{bmatrix} \text{with delay}$$

operators  $\Delta_\tau$  defined as  $\Delta_\tau(\mathbf{x}(t)) = \mathbf{x}(t) - \mathbf{x}(t - \tau)$

**Lemma 4 [95]**

Let  $\mathbf{S}$  be a set of differentiable functions such that  $\mathbf{S} = \{ \tau(t) \mid \tau(t) \in [0, \tau_0], \dot{\tau}(t) \leq d \forall t \}$ . Then the

delay operator obeys the following equality  $\sup_{\tau(t) \in \mathbf{S}} (\| \Delta_\tau \frac{1}{s} \|) = \tau_0$

**Proof:**

Let  $v(t) \in L_{2e}^p$ . Then  $\Delta_\tau \frac{1}{s} v(t) = \int_{t-\tau(t)}^t v(t) dt$  Now,  $\| \Delta_\tau \frac{1}{s} v(t) \|^2 = (\int_{t-\tau(t)}^t v(s) ds)^T (\int_{t-\tau(t)}^t v(s) ds)$

Using Jensen's Inequality, we have  $\| \Delta_\tau \frac{1}{s} v(t) \|^2 \leq \tau_0 (\int_{t-\tau_0}^t v^T(s) v(s) ds)$ . The  $L_2$  norm of  $\Delta_\tau \frac{1}{s} v(t)$

is bounded above as follows

$$\begin{aligned} \| \Delta_\tau \frac{1}{s} v(t) \|_{L_2}^2 &\leq \int_0^\infty \tau_0 (\int_{t-\tau_0}^t v^T(s) v(s) ds) dt \\ &= \tau_0 \int_{-\tau_0}^0 \int_0^\infty v^T(s) v(s) dt ds \\ &\leq \tau_0^2 \| v \|_{L_2}^2 \end{aligned}$$

The above lemma establishes a bound for the delay operator. The small-gain theorem also requires the plant  $\mathbf{T}(s)$  to be stable. The following lemma characterizes the values of the gain  $\gamma$  that render the system stable in the case where the delays are non-existent.

**Lemma 5 [12]**

The system described by equation 57 in the case where the delays are non-existent is stable for the values of  $\gamma$  which satisfy

$$\gamma > \max_{\mu_i \neq 0} \sqrt{\frac{2}{|\mu_i| \cos\left(\frac{\pi}{2} - \tan^{-1}\left(-\frac{Re \mu_i}{Im \mu_i}\right)\right)}} \quad (80)$$

if the connection topology of the spacecraft formation has a rooted directed spanning tree.

Now we make use of the above lemmas to prove the following theorem which establishes the condition under which consensus is guaranteed.

**Theorem 3:**

The system of vehicles with dynamics dictated by equation 57 connected via a network topology containing a rooted directed spanning tree, achieves consensus if along with the conditions of lemmas 3 and 4, the following holds

$$\tau_0 < \frac{1}{\omega \max_{\mu_i} \sum_{k=1}^{p_i} |(j\omega - \frac{\gamma\mu_i \pm \sqrt{\gamma^2\mu_i^2 + 4\mu_i}}{2})^{-k}| (1+\gamma)} \quad \forall \omega \in (0, \infty) \quad (81)$$

where  $\mu_i$  are the eigenvalues of  $-\mathbf{L} \otimes \mathbf{I}_{3 \times 3}$ ,  $p_i$  are the multiplicities of the corresponding eigenvalues of  $\mathbf{A}_0 + \mathbf{A}_\gamma$  and  $\tau_0 = \max_{i,j} h_{ij}$ .

**Proof:** The proof can be found in [95]

## 5.0 RESULTS AND DISCUSSION

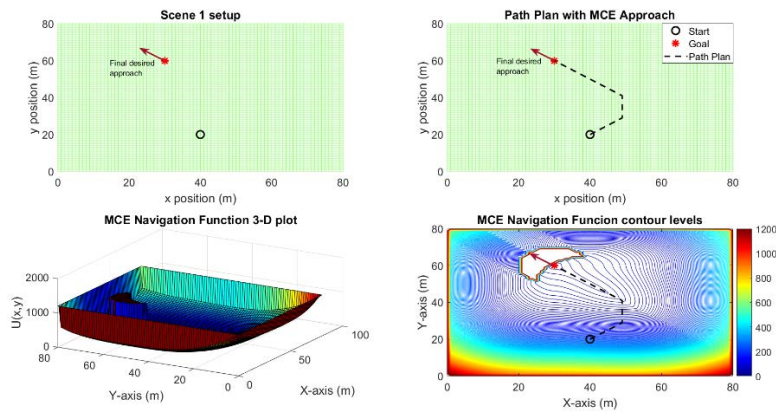
In this section we elaborate several candidate numerical simulations and appropriate cases that illustrate the efficacy of the algorithms and methods developed in this project. The results are discussed in subsections along the same lines as the technical developments.

### 5.1 Numerical Simulations: Guidance to a Reachable State – Navigation Function

There are two different scenarios presented in this section. The first will be a workspace with no obstacles present. The results for this scenario will serve as an illustration of how the control effort based contours of the navigation functions are formed and are used to guide the vehicle's approach. Then, the next scenario will have a single obstacle present in the workspace which will demonstrate the obstacle avoidance capability of this algorithm. Finally, the trajectory tracking control results for an exploration rover will be presented.

#### Scenario 1: No obstacle

For the first scenario, an 80 x 80 meter workspace is considered with no obstacles present. And the fixed time step,  $\delta t = t_f - t_0$  to go between points in the grid is set at 10 seconds for the NF algorithm.

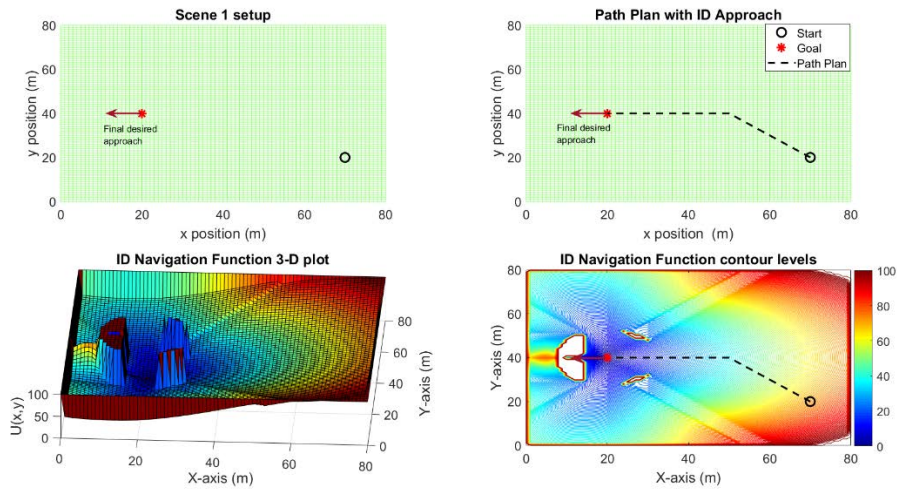


**Figure 12: No Obstacle: Min. control effort based path plan**



The first example shown in Figure 12, was generated using the minimum control effort to a reachable state. The objective reachable state is set as  $\bar{\mathbf{x}}_{goal}^{mce} = [30, 60, -1, 1]^T$ . This desired final state is designed to have the approach of the rover directed to the upper left hand corner of the workspace. The initial position in the workspace is given by the ordered pair  $(x_0, y_0) = (40, 20)$ . The navigation function's contours displayed in Figure 12 are shaped such that it will direct paths in the workspace to approach the final desired state.

Next, the result illustrated in Figure 13 is generated using the inverse dynamics approach. In this result, the objective reachable state is set at  $\bar{\mathbf{x}}_{goal}^{ID} = [20, 40, \pi, 1]^T$ , which will direct the rover's approaching path so that its heading angle will point to the left side of the workspace when it reaches the goal. The starting point for the path plan is set at  $(x_0, y_0) = (70, 20)$

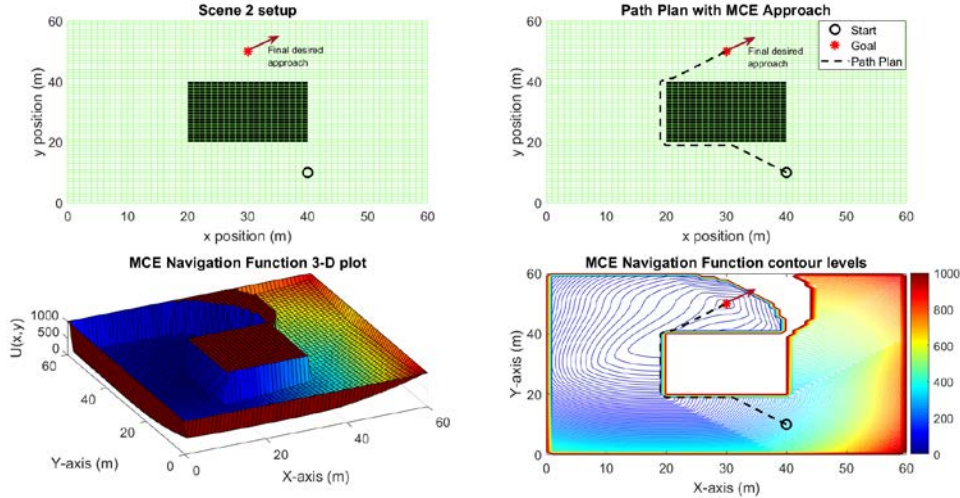


**Figure 13: No Obstacle: Inverse Dynamics based path plan**

It can be seen from the result in Figure 13 that the navigation function's contours create a path that will guide the rover to approach the desired reachable state at the end of the path.

### Scenario 2: Single Obstacle

The second scenario considers a 60 x 60 meter workspace with a single symmetrical obstacle present in the middle of the workspace. This example will demonstrate the algorithms obstacle avoidance capability as well as the effects of the constraints given in equations 8 and 9 in blocking undesired approaches to the objective state.

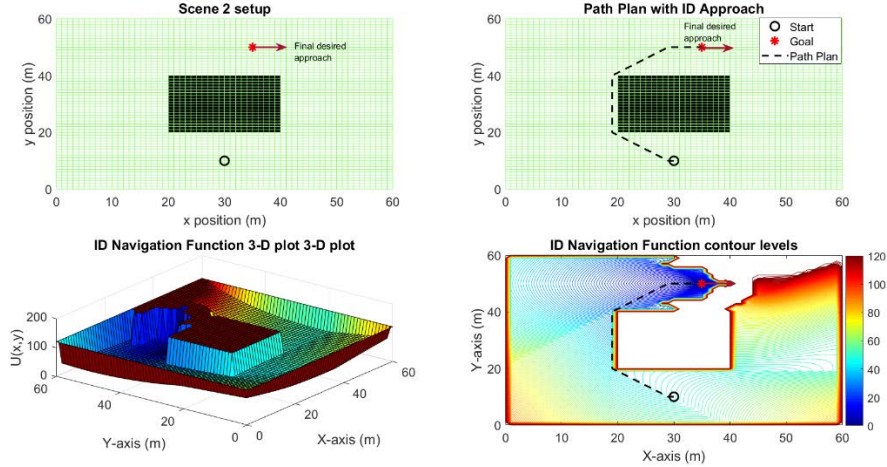


**Figure 14: With Obstacle: Min. control effort based path plan**

The first example in Figure 14 is generated using the minimum control effort method. The desired reachable state is set as  $\bar{\mathbf{x}}_{goal}^{mce} = [30, 50, 1, 1]^T$ . This objective will direct the rover's path so that its heading angle will be towards the upper right corner of the workspace as it reaches the goal. This result is shown in the path plan in Figure 14. Furthermore, the check constraints step in the algorithm blocks paths from the opposite direction as expected.

The example in Figure 15 was created with the inverse dynamics approach with an objective reachable state  $\bar{\mathbf{x}}_{goal}^{ID} = [35, 50, 0, 1]^T$ . The constraint check in the algorithm is also evident in this example with a virtual wall blocking paths to the goal that approach from the opposite direction. Also, the path plan given in Figure 15, directs the heading angle to the desired state aligned with the inertial x-axis.

The contour levels of the navigation functions in Figure 12 through Figure 15 are able to generate path plans that facilitate the motion of the rover to approach the desired reachable state. The paths are generated with the best-first graph search method along the negative gradient. It is also apparent that the constraint check in the algorithm is beneficial with symmetrical workspaces in helping decide which direction to go in order to reach the desired state.



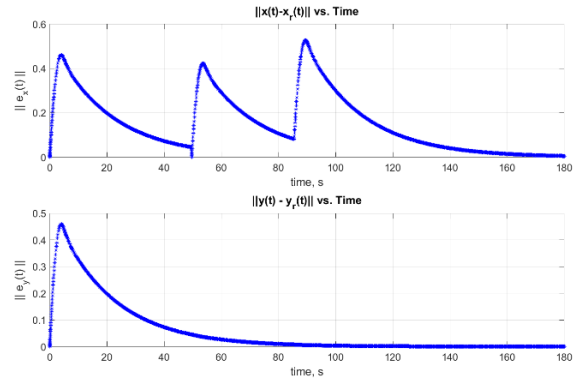
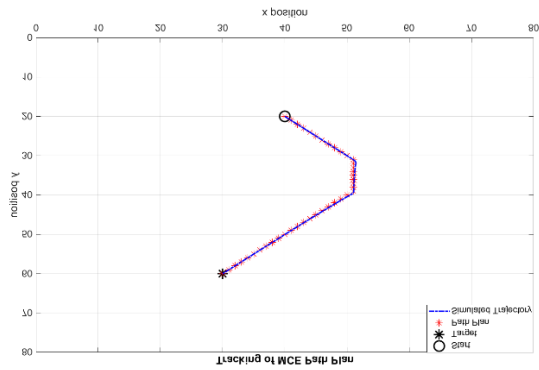
**Figure 15: With Obstacle: Inverse Dynamics based path plan**

### Trajectory Tracking

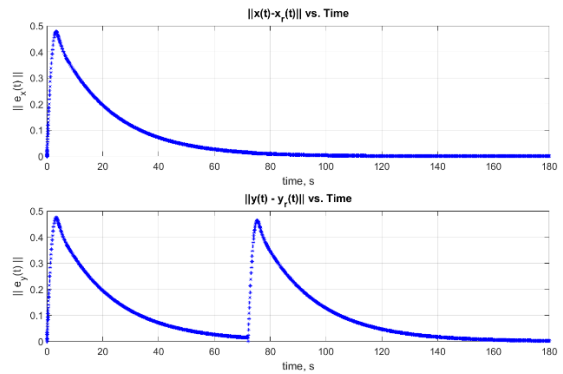
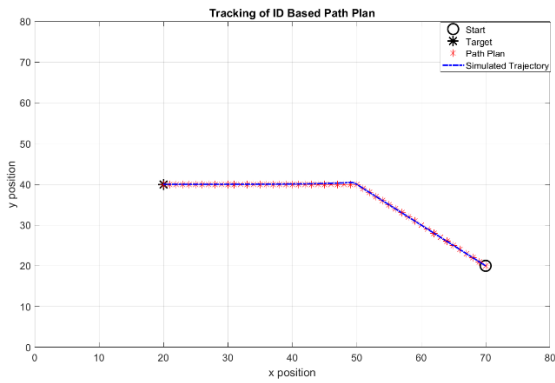
The results of the trajectory tracking controller with the rover model are displayed in Figures 16 to 19. The control gains for each of the examples was set as:  $\lambda_x = 0.05$ ,  $\lambda_y = 0.05$ ,  $\lambda_{\psi} = 5$ ,  $\lambda_v = 5$ ,  $\alpha_1 = \alpha_2 = 5$ . And the time to reach the goal for the trajectory design is set at 3 minutes for each case given.

The results for each of the examples considered demonstrate the effectiveness of the backstepping control design in following the desired trajectory. There are spikes in the position error plots. These spikes correlate to the times when the vehicle reaches a point where it must turn to continue on the trajectory. Even as the spikes occur, the errors quickly decay to zero as expected based on the stability analysis of the controller. A cause of this phenomena could be due to the fact that the trajectory design does not consider a desired heading profile for the rover to track.

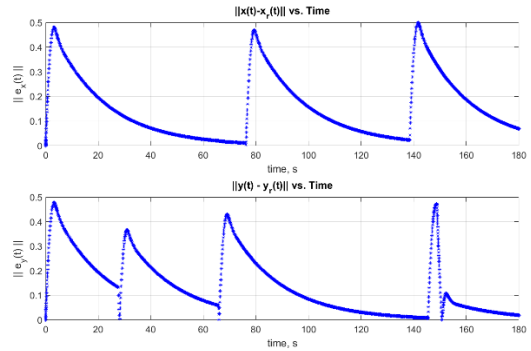
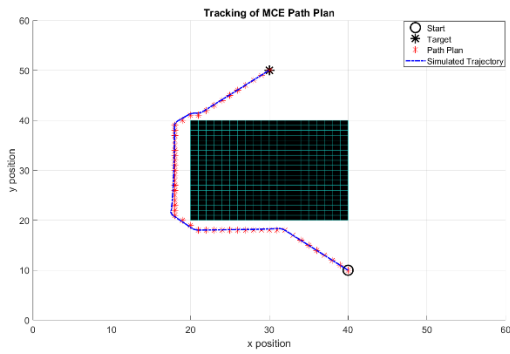
Additionally, there is some steady state error present in the position tracking towards the end of the simulations, which is evident in Figures 16 to 19. The source of this error can be attributed to the singularity avoidance algorithm. In these instances, the heading angle is locked to the reference value due to the desired velocity being too small, i.e.  $v_d \leq \dot{0}$ , as the rover approaches its objective and is therefore not able to adjust accordingly.



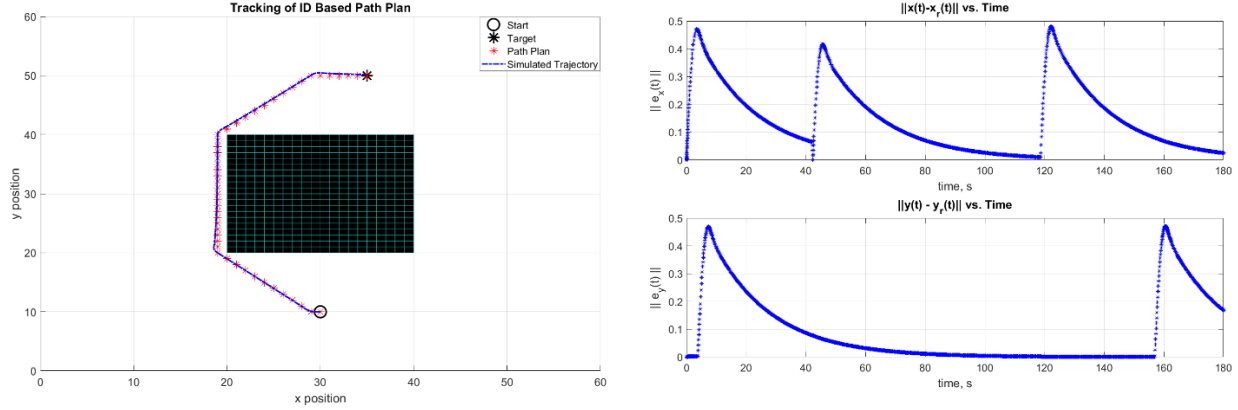
**Figure 16: No Obstacle: Tracking performance of MCE Plan**



**Figure 17: No Obstacle: Tracking Performance Inverse Dynamics**



**Figure 18: With Obstacle: MCE Path Plan**



**Figure 19: With Obstacle: Tracking performance of the Inverse Dynamics**

## 5.2 Numerical Simulations: Nonlinear MPC vs Nonlinear Backstepping like guidance law

For all the trajectory tracking results presented, the same path plan results are used. And the reference trajectory is generated along the path plan with a desired time to reach the goal set at 180 seconds (or 3 minutes). For the NMPC guidance law, the sampling time used to discretize the SDC system is chosen as  $\Delta t = 0.1$  seconds and the prediction horizon length is chosen as  $N = 20$  time steps. Also, the input and state weighting matrices are chosen as:

$$\mathbf{R} = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

The input constraints chosen for the simulation are chosen as:

$$\begin{aligned} -15 \text{ (deg / s)} &\leq \dot{\psi} \leq 15 \text{ (deg / s)} \\ -0.15 \text{ (m / s}^2\text{)} &\leq \dot{v} \leq 0.15 \text{ (m / s}^2\text{)} \end{aligned}$$

for the heading angle turn rate and forward acceleration respectively. The state, or output, constraints are placed on the heading angle and velocity respectively as:

$$\begin{aligned} -180^\circ &\leq \psi \leq 180^\circ \\ 0.001 \text{ (m / s)} &\leq v \leq 1 \text{ (m / s)} \end{aligned}$$

These constraints are chosen based on experiments conducted with the experimental mobile robot platform in the Aerospace Systems Laboratory at the University of Texas at Arlington. However, the lower bounds of 0.001 m/s for the velocity is chosen due to the lack of controllability with the SDC form of the equations when the velocity is equal to zero.

Approved for public release; distribution is unlimited.

Then, for the backstepping guidance law results, the control gains are chosen as constants where  $\lambda_x = 2$ ,  $\lambda_y = 2$ ,  $\lambda_\psi = 5$ ,  $\lambda_v = 5$

### Case 1: Framework Simulation Results

The simulation results for case 1 are presented in Figures 20 through 25. The results illustrate the reference path tracking performance in Figure 20 and Figure 21. Then, the simulated heading angle and velocity signals are depicted in Figure 22 and Figure 23 respectively. Last, the guidance command time histories are shown in Figure 24 and Figure 25 for  $u1$  and  $u2$  respectively. For the simulation results presented, the subscript NL denotes the set of results from the backstepping-based nonlinear guidance law and the subscript NMPC denotes the set of results from the nonlinear model predictive control guidance law.

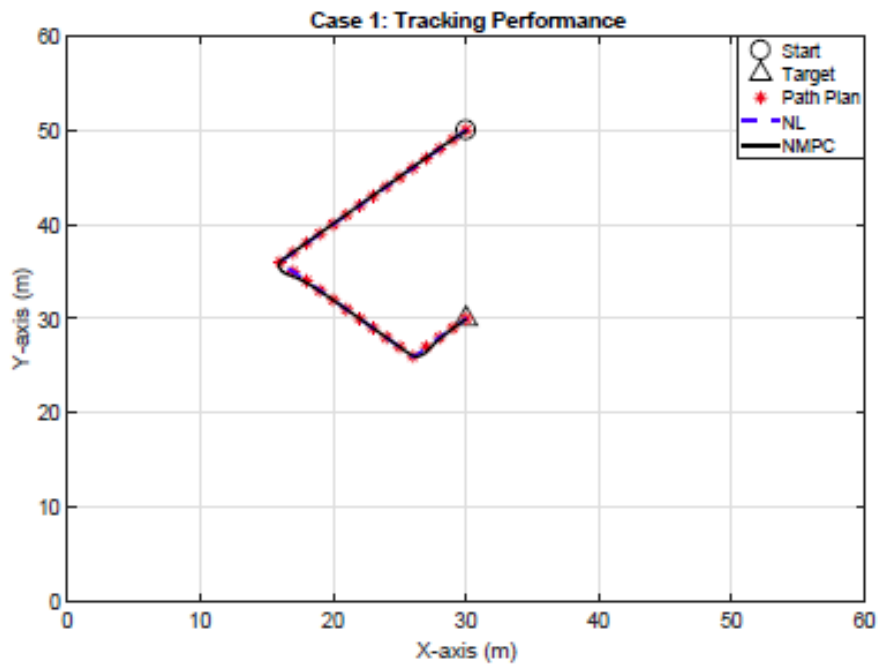
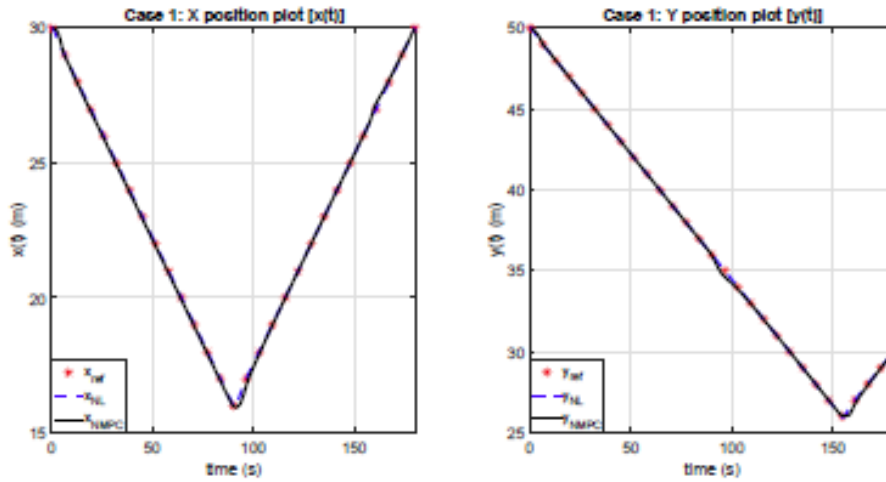
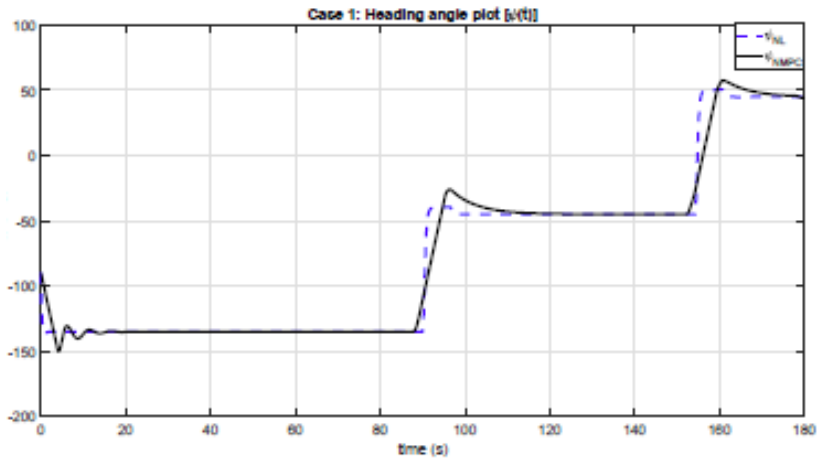


Figure 20: Reference path tracking for case 1



**Figure 21: x and y position tracking for case 1**



**Figure 22: Heading angle plot for case 1**

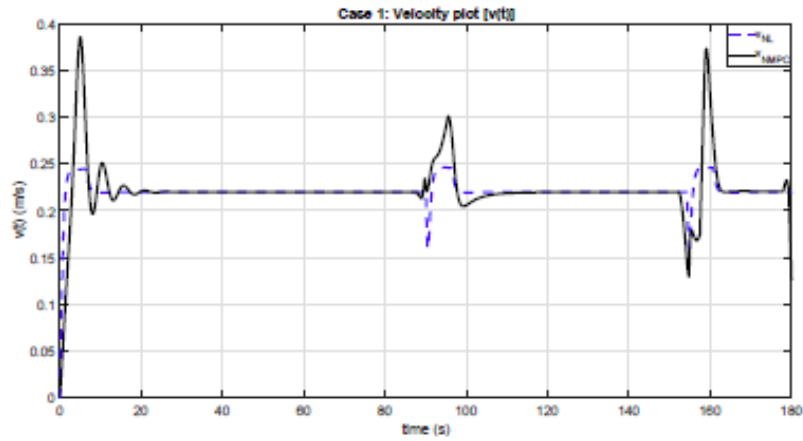


Figure 23: Velocity plot for case 1

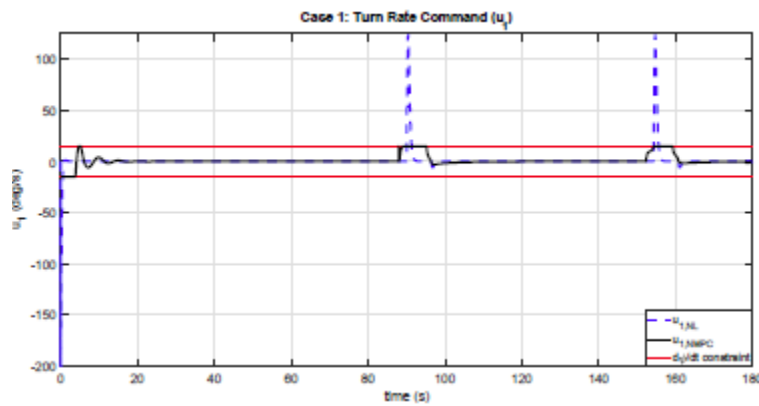


Figure 24: Guidance command  $u_1$  for case 1

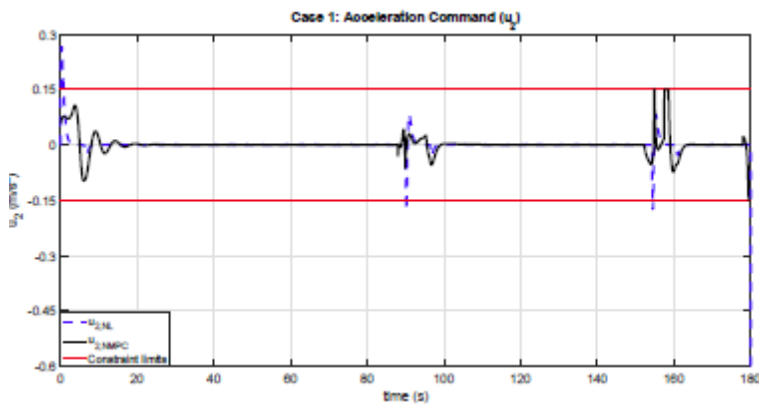


Figure 25: Guidance command  $u_2$  for case 1



From the reference path tracking results in Figure 20 and Figure 21, the performance of the two guidance laws are nearly identical. Both guidance laws tightly track the reference path given with each simulation result showing that the vehicle arrives at its objective location.

The different behavior of the two guidance designs is evident in the plots of the robot's heading angle and velocity, Figure 22 and Figure 23 respectively. In these plots, it can be seen that there are spikes in the signals which correlate to the corner points in the reference path where a sharp turn occurs. The NMPC signals spike higher than the backstepping signals at these times due to the constraints being considered. The constraints enforced (essentially the vehicle's acceleration and the turn rate, see Figure 24 and Figure 25) within the NMPC framework imply that it cannot settle to a steady value as suddenly as the backstepping designs, hence the overshoot in the signals.

As highlighted previously, another important contrast between the two guidance laws is evident in the plots of the guidance command history in Figure 24 and Figure 25. In these plots, the constraints enforced within the NMPC design are overlaid on the plot of the two signals. The results in Figure 24 and Figure 25 illustrate that the guidance commands for the NMPC design obey the constraints of the system. In contrast, the backstepping guidance commands have sudden sharp changes over the course of the simulation which would violate these constraints.

Also, notice the behavior of the acceleration guidance command,  $u_2$ , at the end of the simulation. At the end of the simulation the guidance command drops suddenly to a low negative value. This occurs because the guidance law is commanding the vehicle to decelerate and stop at the end of the trajectory. The NMPC commands the vehicle's acceleration to the lower bounds of its capability whereas the backstepping command tends to a value that violates this constraint.

### Case 2: Framework Simulation Results

The simulation results for case 2 are presented in Figure 26 through Figure 31. As with the previous case, the results illustrate the reference path tracking performance first in Figure 26 and Figure 27. Then, the simulated heading angle and velocity signals are depicted in Figure 28 and Figure 29 respectively. Last, the guidance command time histories for case 2 are shown in Figure 30 and Figure 31 for  $u_1$  and  $u_2$  respectively.

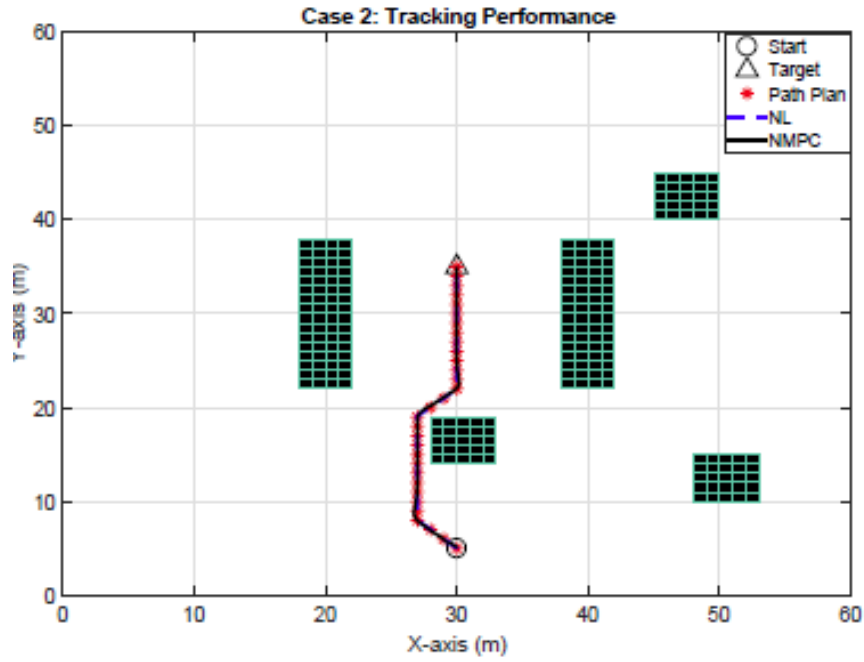


Figure 26: Reference path tracking for case 2

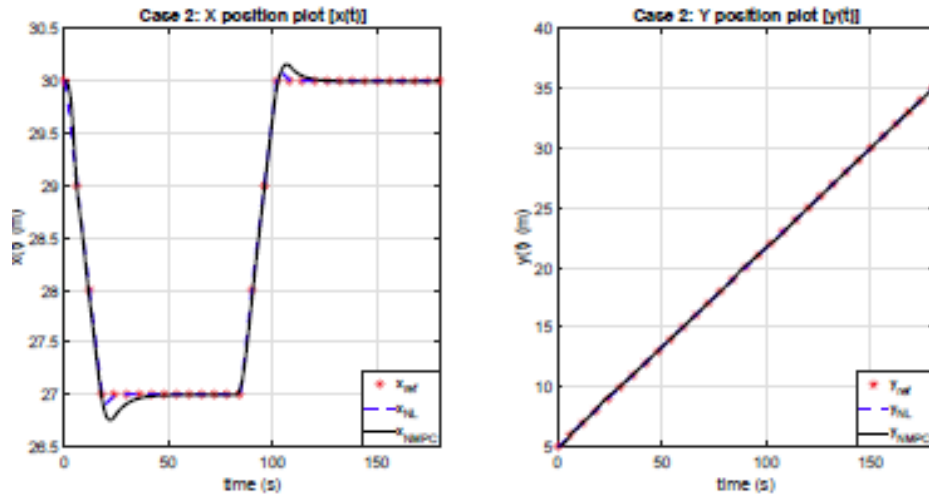


Figure 27: x and y position tracking for case 2

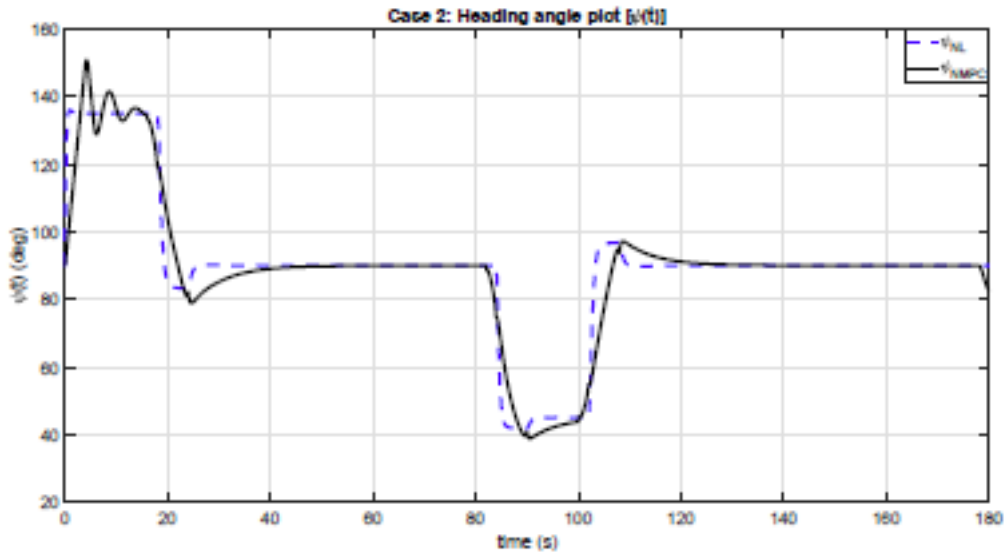


Figure 28: Heading angle plot for case 2

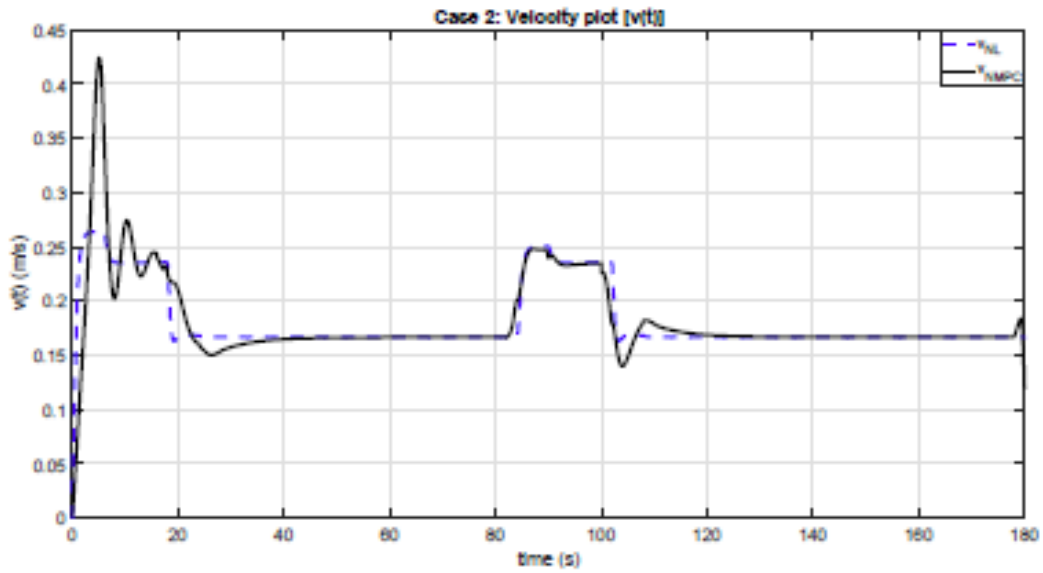
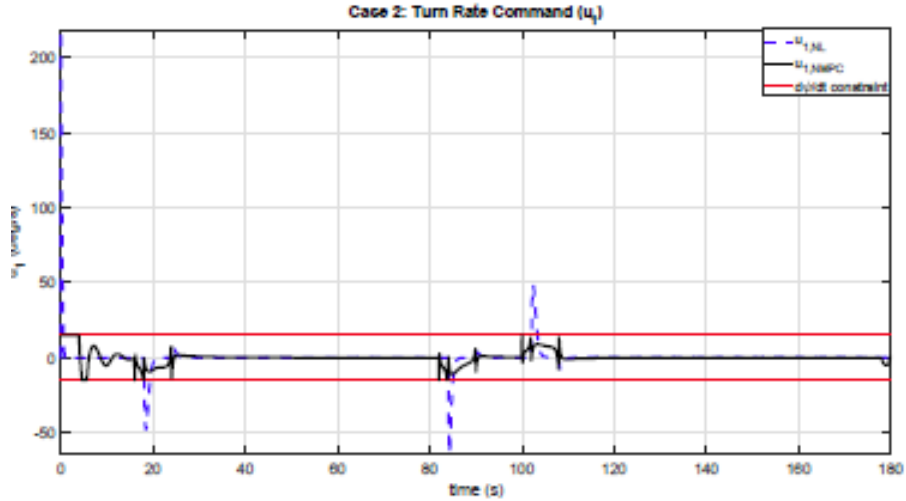
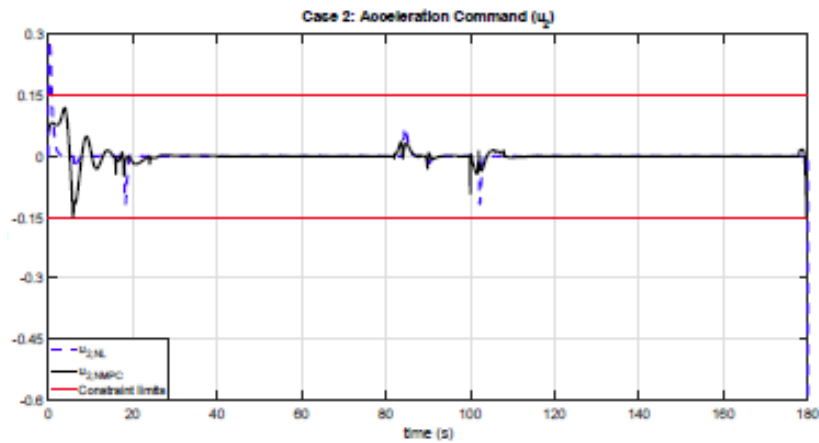


Figure 29: Velocity plot for case 2



**Figure 30: Guidance command  $u_1$  for case 2**



**Figure 31: Guidance command  $u_2$  for case 2**

Similar to the simulation results in case 1, the reference path tracking performance with both guidance laws are indistinguishable, as illustrated in Figure 26 and Figure 27. The differences of the two methods are most evident in the simulation results of the heading angle, velocity and guidance command signals, depicted in Figure 28 through Figure 31 respectively.

As with the results for case 1, the heading angle and velocity signals of the two simulations behave similarly with the signals spiking at time instants that correlate to when the robot is making sudden turns along the reference path. The difference lies in the time it takes for the signals to settle. The backstepping signals settle rather quickly whereas the NMPC signals do so gradually. This behavior with the NMPC design is attributed to the constraints being enforced within its design.

Also, the guidance command history plots, shown in Figure 30 and Figure 31, display similar results to what was presented with case 1. The guidance commands originating from the NMPC design are able to adhere to the constraints on the system over the duration of the simulation. In comparison, the backstepping commands again provide sharp and sudden changes which at times exceed the capabilities of the vehicle.

Overall, the guidance commands from the backstepping approach adhere to the constraints of the system for most of the simulation. However, the presence of sudden sharp changes in its guidance commands, as well as heading angle and velocity signals, could present complications in applying the backstepping approach to a mobile robot platform.

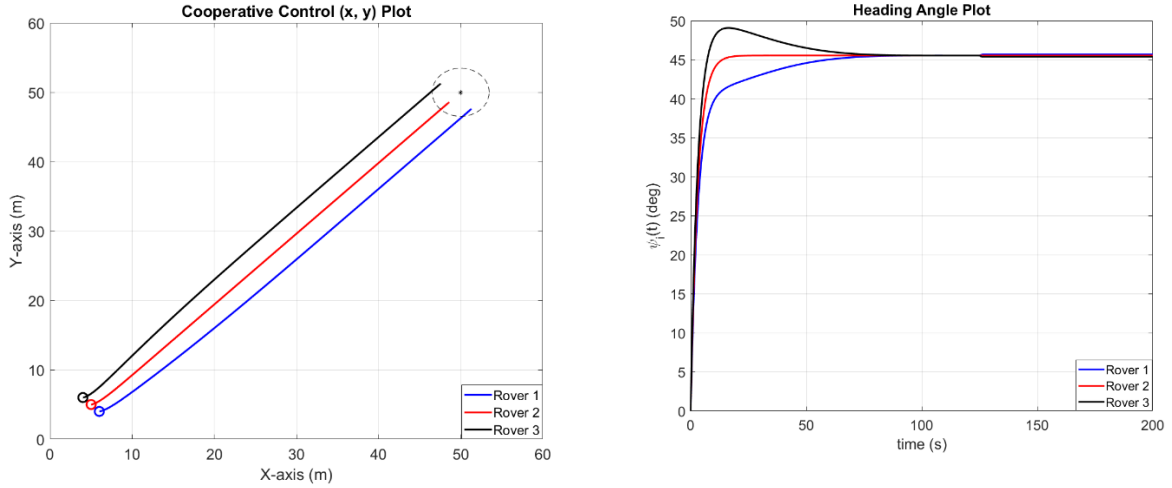
Conversely, all the state and input signals corresponding to the NMPC guidance law operate within the defined capabilities of the vehicle as intended. These signals do have some sudden changes but are less amplified in comparison with the backstepping approach over the course of the simulation. Thus, the capability of the NMPC design to ensure that the vehicle can operate within its physical limitations could prove beneficial for the kinds of tasks it may face.

### 5.3 Numerical Simulations: Cooperative Control of Multiple Vehicles (No Time Delays)

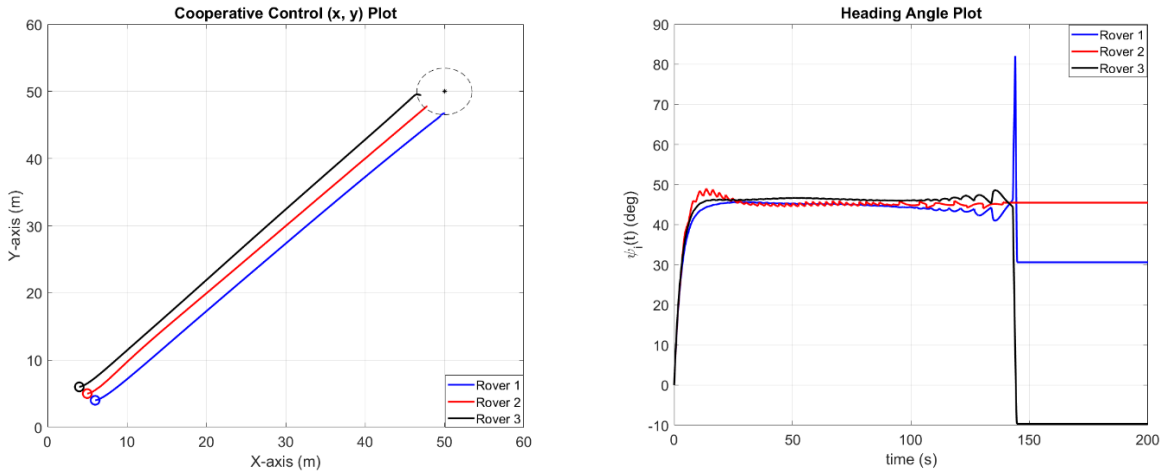
The simulation results in this chapter consider scenarios involving three vehicles in a given environment. The cooperative APF parameters for the steering command for the simulation are chosen as  $a = 0.3$ ,  $b = 1$ ,  $c = 10$  and the parameter for the steering command to the goal as  $k_a = 0.5$ . The collision avoidance parameters for the steering command are set as  $b_r = 1$ ,  $c_r = 2$ . The control gains for the guidance command inputs are set as  $\alpha_1 = 0.5$  and  $\alpha_2 = 0.5$ . The maximum velocity for each robot is set at  $v_{max} = 0.75$  ~m/s. The navigation function algorithm method used for the results in this section is the minimum control effort approach.

The first scenario, shown in Figure 32 and Figure 33 is for an environment with no obstacles present. This example represents a case where the swarm aggregation policy can be tested and compared with the social foraging policy to the same goal. The swarm aggregation policy is enacted by setting  $\delta = 1$  and the social foraging policy is enacted by setting  $\delta = 0$ .

The results in Figure 32 and Figure 33 show the positions of the robots, starting in close proximity in the bottom corner and traveling towards the opposite corner of the environment while avoiding collisions within the group. The robots position components look similar, but their heading angles fluctuate largely in Figure 33 with the social foraging policy. This behavior begins as the robots are approaching the safety zone around the goal position. When the vehicles enter the safety zone, their velocities are driven to zero so they stop, but the heading angles are directing the robots away from each other. In contrast, the heading angle plot in Figure 32 shows the robots smoothly achieving consensus as they enter the safety zone around the goal.



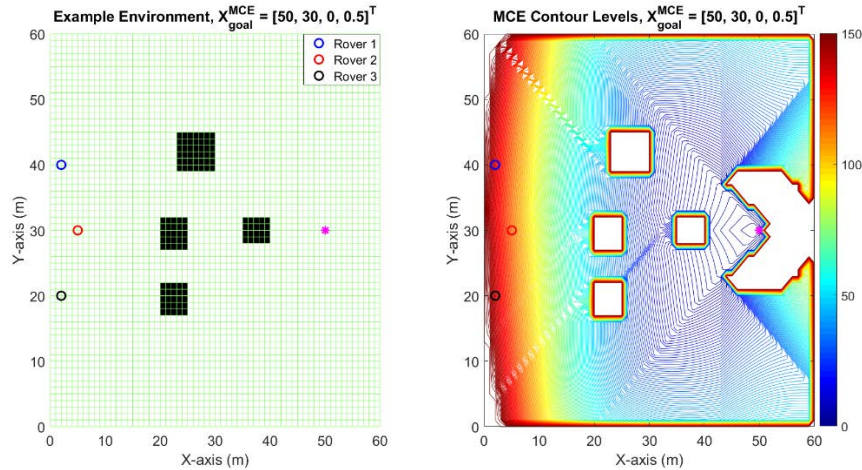
**Figure 32: Example 1 with swarm aggregation policy - Position, Heading**



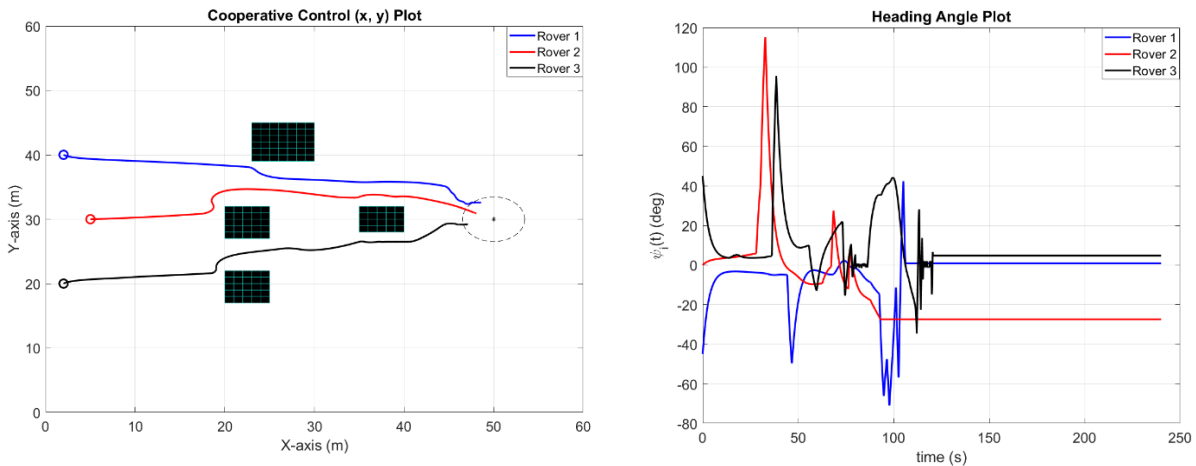
**Figure 33: Example 1 with social foraging - Position, Heading**

The next two examples are considered only with social foraging behavior, by setting  $\delta = 0$ . The steering commands are generated by the navigation function's potential field combined with the collision avoidance potential. The first set of plots show the given obstacle laden environment and its associated navigation function potential field. The second set of plots show the simulation results with the position plots and the heading angle plots.

The results illustrated in Figure 34 and Figure 35 are for an environment where multiple scattered hazards lie between the robots and their objective. The simulation results in Figure 35 show that the robot's steering commands allow the vehicles to avoid colliding with the obstacles and each other while reaching the safety zone near the objective.



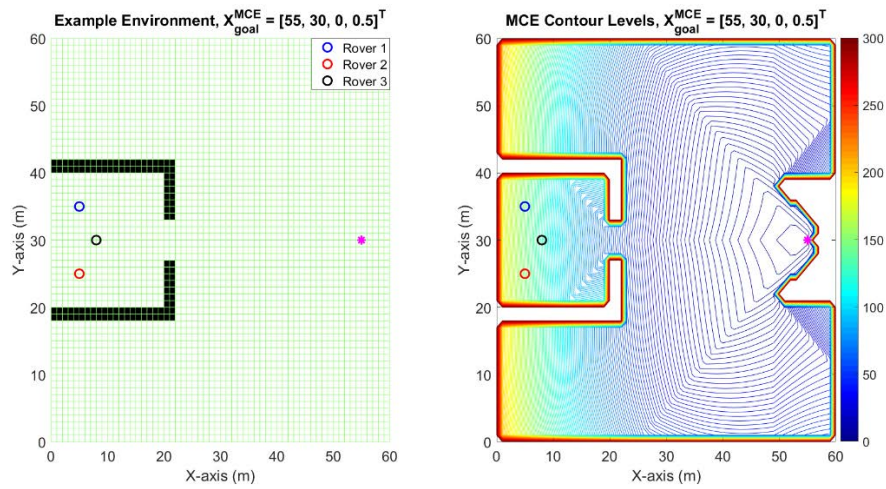
**Figure 34: Example 2 environment and navigation function potential field**



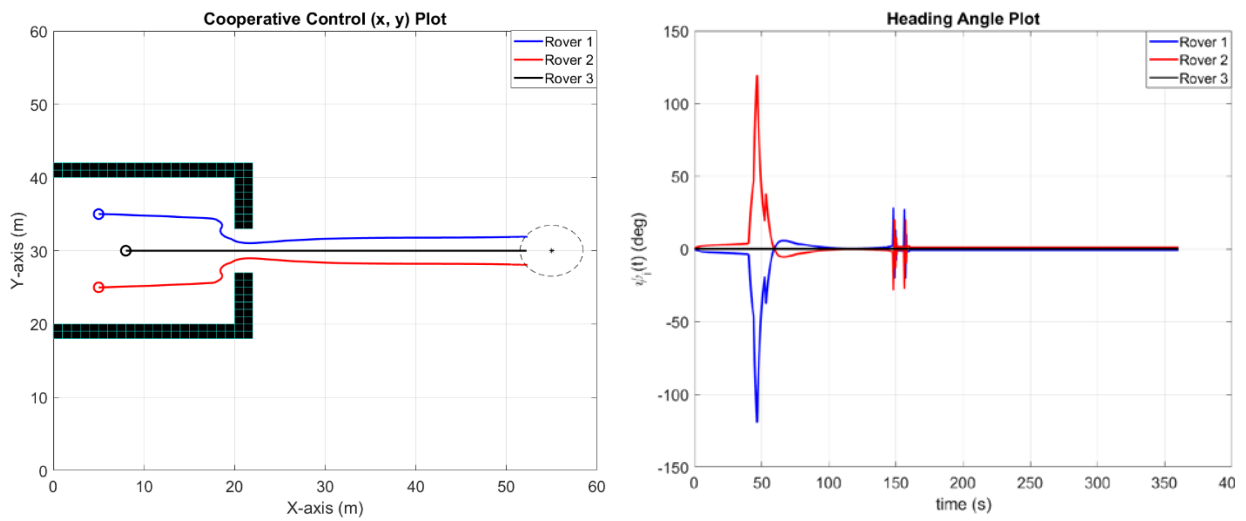
**Figure 35: Example 2 with social foraging policy - Position, Heading**

The next set of results illustrated in Figure 36 and Figure 37 are for an environment where the vehicles must come together through a tight opening before spreading apart and reaching the goal. The simulation results in Figure 37 demonstrate that the robots achieve their task without conflict as they come together to go through the opening then spread apart in the free space on the other side and ultimately arrive at the safety zone near the objective.

Approved for public release; distribution is unlimited.



**Figure 36: Example 3 environment and navigation function potential field**

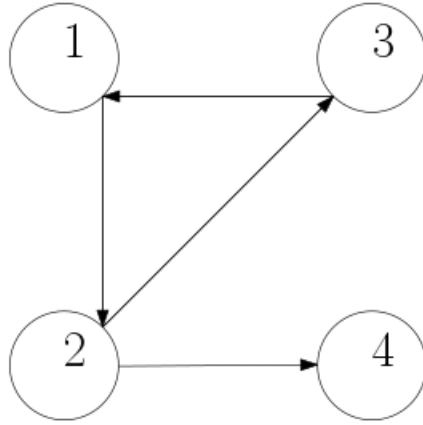


**Figure 37: Example 3 with social foraging policy - Position, Heading**

#### 5.4 Numerical Simulations: Cooperative Control of Multiple Vehicles (With Time Delays)

In this section, the proposed control strategy is implemented for a formation of 4 spacecraft communicating with each other via the graph shown in Figure 38.





**Figure 38: Communication Graph**

The simulation conditions are as specified in the tables below.

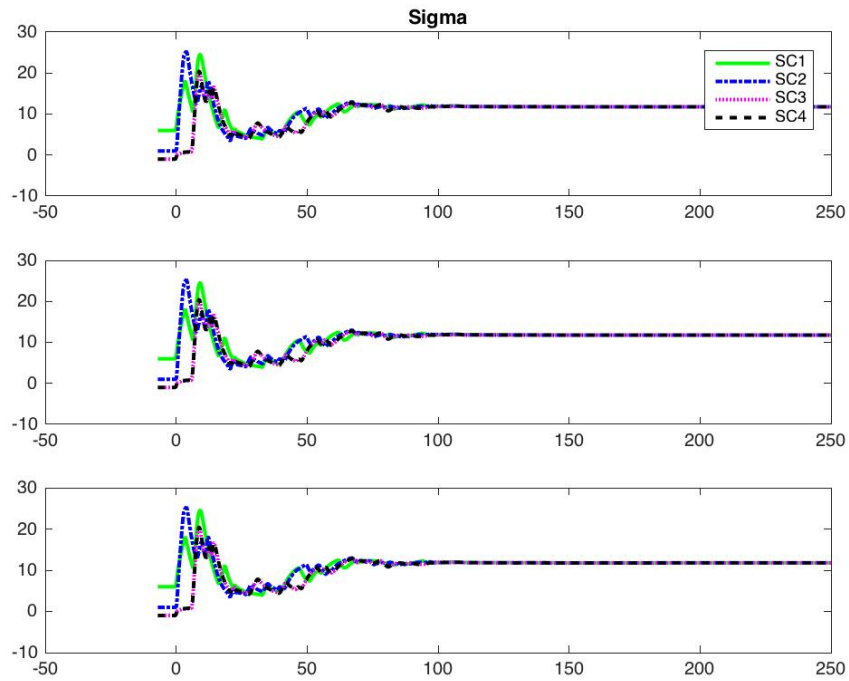
**Table 1: Spacecraft Initial Conditions and Parameters**

	Spacecraft 1	Spacecraft 2	Spacecraft 3	Spacecraft 4
Inertia Matrix	$20 \times \mathbb{I}_{3 \times 3}$	$30 \times \mathbb{I}_{3 \times 3}$	$40 \times \mathbb{I}_{3 \times 3}$	$50 \times \mathbb{I}_{3 \times 3}$
Initial attitude (MRP)	$\begin{bmatrix} 0.8 \\ 0.8 \\ 0.8 \end{bmatrix}$	$\begin{bmatrix} 0.4 \\ 0.4 \\ 0.4 \end{bmatrix}$	$\begin{bmatrix} -0.6 \\ -0.6 \\ -0.6 \end{bmatrix}$	$\begin{bmatrix} -0.8 \\ -0.8 \\ -0.8 \end{bmatrix}$
Initial Angular Velocity	$\begin{bmatrix} 0.06849 \\ 0.06849 \\ 0.06849 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} -0.09615 \\ -0.09615 \\ -0.09615 \end{bmatrix}$	$\begin{bmatrix} 0.06849 \\ 0.06849 \\ 0.06849 \end{bmatrix}$

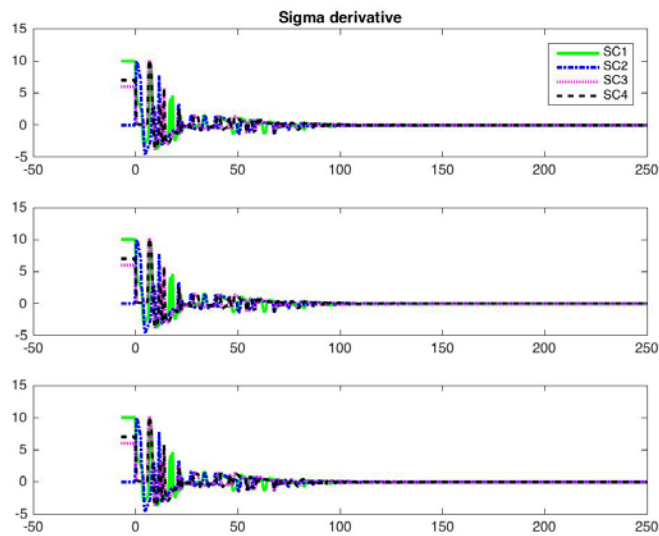
**Table 2: Time Delay Parameters**

	1 $\rightarrow$ 2	2 $\rightarrow$ 3	3 $\rightarrow$ 1	2 $\rightarrow$ 4
Bound on delay (h)	5 seconds	6 seconds	7 seconds	5 seconds
Bound on delay derivative (d)	1	2	0.5	1

For the given communication topology, we calculate  $\gamma > 1.414$ . Thus, the damping gain  $\gamma$  is set to 5. The Theorem then yields a conservative upper bound on the delay at 9.6346 seconds. The simulation results are presented below. All vectors are expressed in body frames of the respective spacecraft.

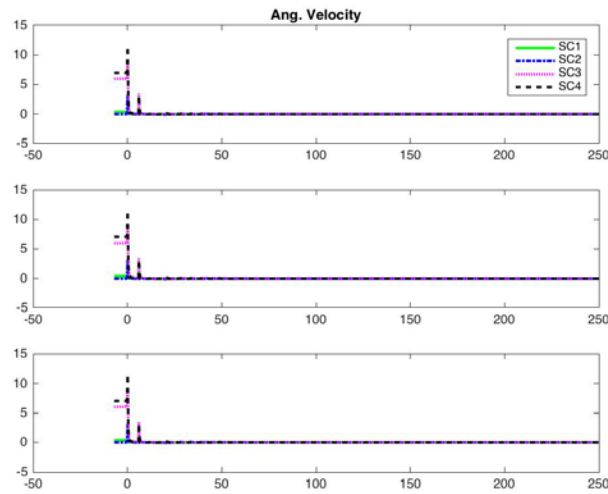


**Figure 39: Spacecraft Attitude vs Time**



**Figure 40: Spacecraft Attitude Rates vs Time**

Clearly, the spacecraft achieve consensus despite the delays.



**Figure 41: Spacecraft Angular Velocities vs Time**

It can be seen that when the gain  $\gamma$  is set to 0.1, we observe that consensus is not achieved and the states of the spacecraft diverge from each other, which is consistent with the bound calculated.

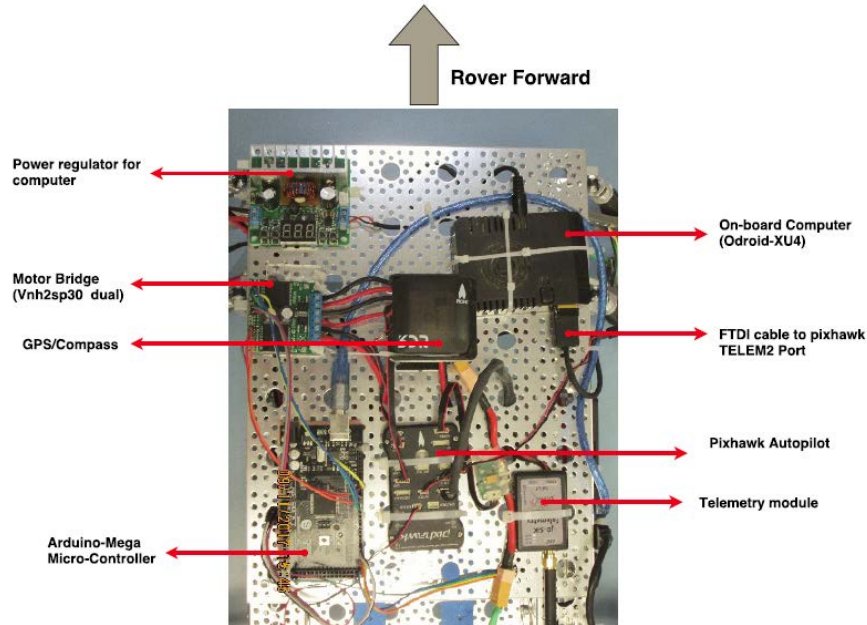
### 5.5 Experimental Results: Cooperative Control of Multiple Vehicles (With Time Delays)

To verify the approached developed thus far, an experimental test bed involving two wheeled mobile robots was developed and the path planning and control algorithms were implemented in a cyber physical systems (CPS) set up.



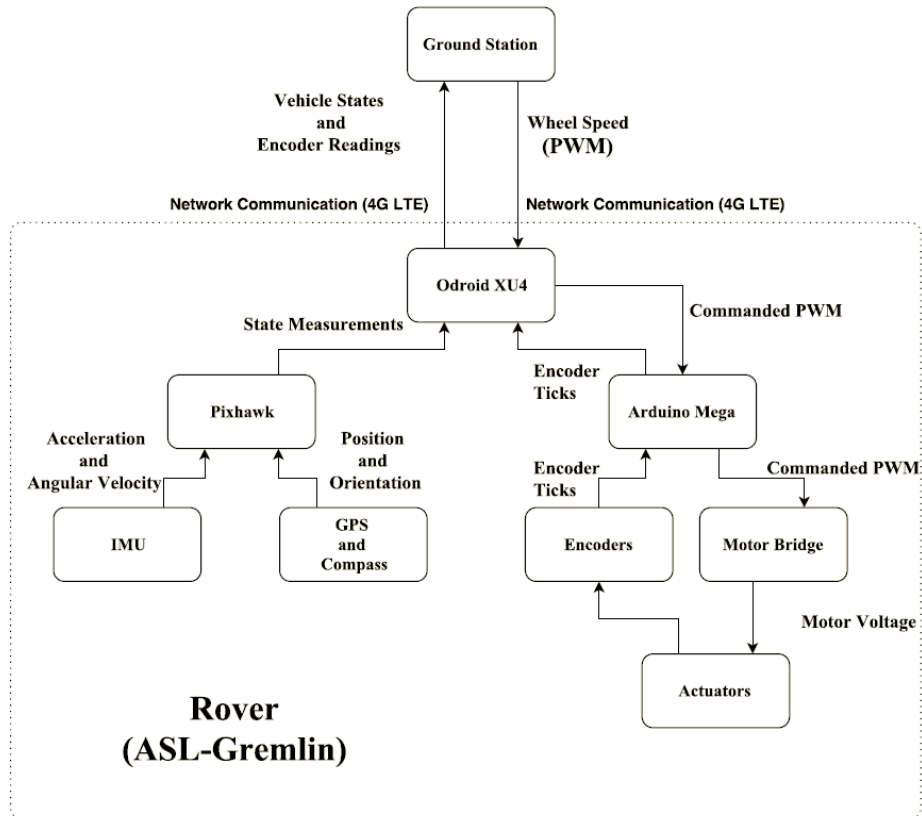
**Figure 42: Rover built by the Aerospace Systems Lab (ASL) at UT Arlington**

The vehicle used to conduct the experiments presented within this paper was designed as a modified version of the Mantis™ off road rover chassis. The design has a motor actuator attached to each wheel making it four wheel drive and is made with aluminum shock absorbing suspension making it able to travel through different types of terrain. An illustration of the finished vehicle is given in Figure 42.



**Figure 43: Top-View of the ASL-Gremlin-Rover with all the components**

The main components onboard the rover consists of a single-board computer (Odroid XU4), a Pixhawk autopilot and an Arduino Mega micro-controller board. The sensors used by the system are an inertial measurement unit (IMU), GPS receiver and compass which are connected through the Pixhawk autopilot as well as Quadrature encoders mounted on the wheels that are connected through the Arduino Mega. There are also four actuated motors that are controlled through the Arduino interface. The information on-board the rover itself like pixhawk sensor data, encoder data is passed through the Odroid computer which is then transmitted to a ground station computer where it is processed. The ground station computer is connected to the rover system through a wireless network and is used to handle the computations needed for localization and control of the rover. The information flow chart for the rover system is displayed in Figure 44. The cyber-physical system (CPS) shown in Figure 44 depicts how the system on the rover manages the physical processes involved and communicates with a network connected ground station to process the data. The communication between the devices included in the system architecture is handled through the Robot Operating System (ROS). ROS framework facilitates the communication of information between different components through network connections. Therefore, it renders the CPS architecture incorporated in this design possible.

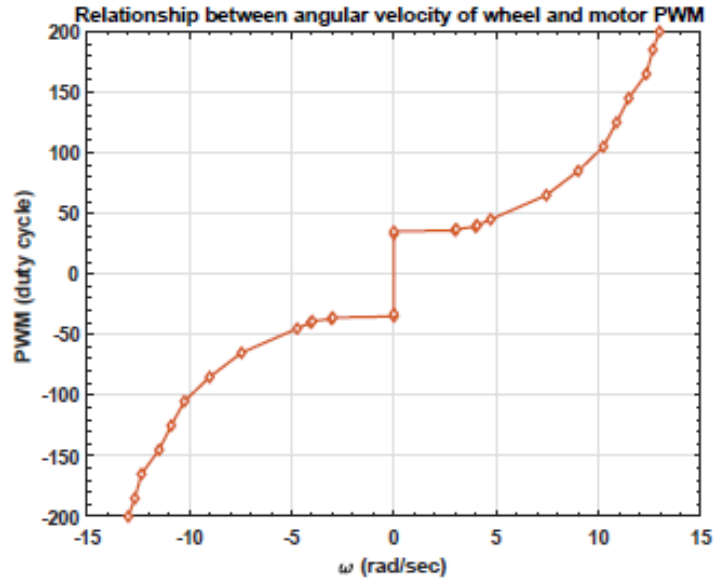


**Figure 44: Illustration of the system architecture making up the CPS of the experimental setup**

In-order to command the motors, an appropriate relationship between the angular velocity and motor PWM voltage signals is required. To get this relationship, a series of PWM values are selected. Then these selected PWM values are applied to motors with the rover on the ground for a time span of  $\delta t$  and the traveled distance by rover is measured. By using PWM and the distance traveled, the angular velocity of the wheel ( $\omega$ ) is calculated using equation 82 and the obtained relationship between the angular velocity of the wheels to motor PWM signal is illustrated in Figure 45.

$$\omega = \frac{\text{distance travelled}}{r\delta t} \quad (82)$$

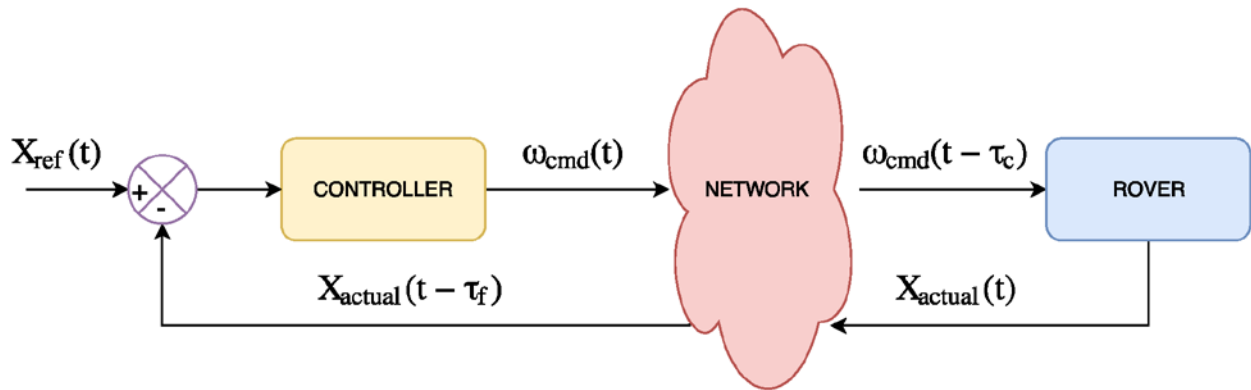
Figure 45 shows the relationship between the angular velocity of the wheels to motor PWM signal.



**Figure 45: Relationship between angular velocity of wheel and motor PWM**

There will be few pitfalls to overcome with the CPS based architecture employed on the rover. Chief among them is the presence of network communication delays. These kinds of difficulties arise due to the strength of the network being used and the capabilities of the hardware involved. The results here will demonstrate the effects of these delays and any other issues that can arise.

In-order to measure the network communication delay between the rover and the ground station, GPS pseudo-range based technique is used. All the data before being sent to the ground station is time-stamped which says the time-of-sent for a signal. And when a signal reaches the ground station, a time-of-arrival of signal is recorded. So the delay between the rover and the ground station is the difference between the time-of-sent and time-of-arrival of a particular signal. This method requires that both the clocks (on the ground station and on the rover) need to be in sync, which is not possible because of the time-drift in systems. So, at a fixed time-interval (5 sec) the local clock offset between the ground station and the rover is measured. Then, adding or subtracting the local clock offset to the time-of-arrival of signal, an approximation of the communication delay is obtained.



**Figure 46: Closed loop control system with communication delay**

Since the overall implementation is a CPS based architecture, network communication delays and their effects on the closed loop system were explicitly characterized in simulation before final implementation. The experimental results clearly demonstrate the effects of these delays. To measure the network communication delay between the rover and the ground station, the time-stamps on the sent and received data were compared. Together with the knowledge of the local clock offset an approximation of the communication delay is obtained. To observe the effect of communication delays on the trajectory tracking performance, an artificial communication delay is generated between the controller and the rover model in simulation. A communication delay in range 0.1 sec to 0.8 sec is selected for both the control signal and feedback signal. Figure 47 shows the results obtained from simulation, where 1 in the table represents rover reached the goal and 0 represents rover didn't reach the goal. As shown in Figure 47, when the delay in the signal is more than 0.6 sec the stability of the controller degrades which results in the system instability.

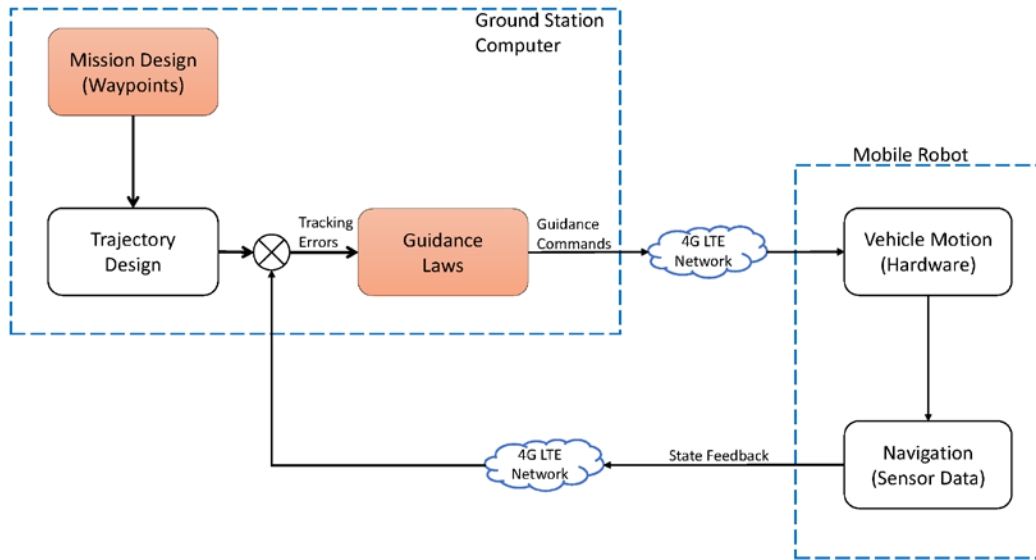
**Stability Analysis with delay**

Control signal delay ( $\tau_c$ )	0.1s	0.15s	0.2s	0.25s	0.3s	0.35s	0.4s	0.45s	0.5s	0.55s	0.6s	0.65s	0.7s	0.75s	0.8s
0.8s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.75s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.7s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.65s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.6s	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.55s	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0.5s	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0.45s	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
0.4s	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
0.35s	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0.3s	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
0.25s	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
0.2s	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
0.15s	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0.1s	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0

Feedback signal delay ( $\tau_f$ )

**Figure 47: System stability analysis with different combination of delays**

The flow of information, for a single vehicle, illustrating how this architecture is used within the GNC framework, is given in Figure 48. The results in this dissertation make use of a cyber-physical system architecture wherein the main data processing and guidance commands are generated on a separate ground station computer which dictates the desired behavior for the physical mobile robot system. The information is transmitted between a ground station CPU and the Odroid on-board CPU through a 4G LTE mobile network hotspot.

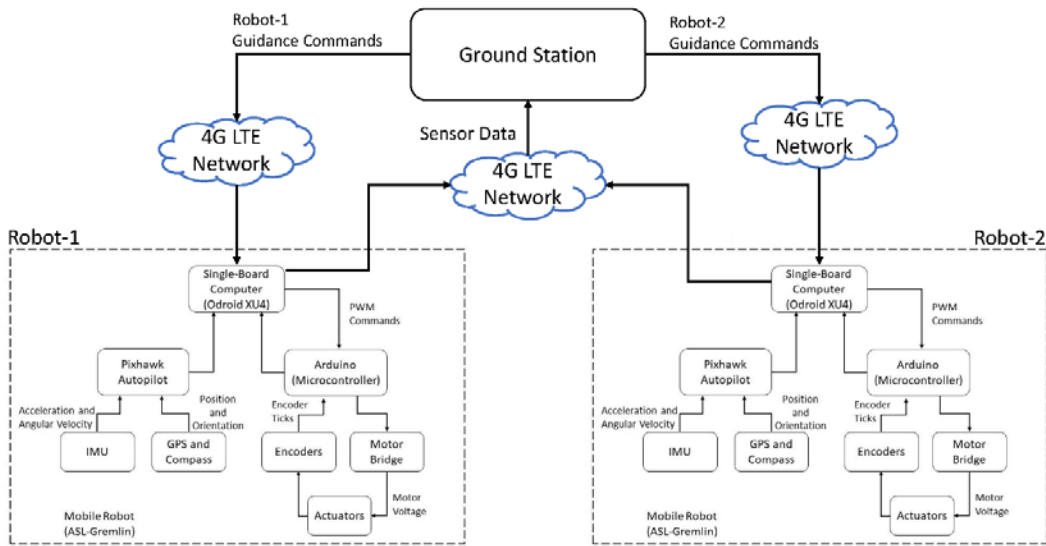


**Figure 48: Flow of information within the GNC framework for the mobile robot platform**

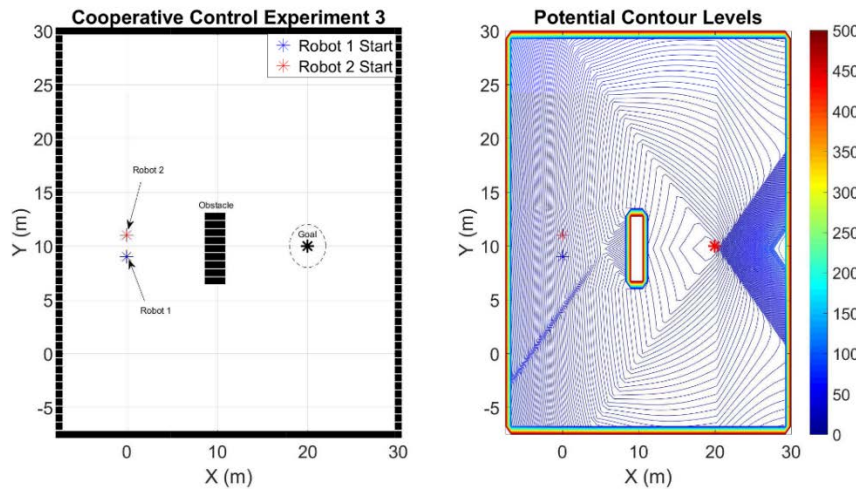
For the cooperative control experiments presented in this project, two mobile robot platforms are used in combination with a ground station computer. Similar to the individual mobile robot tasks, the communication of information will take place over a 4G LTE mobile network and is considered as a cyber-physical system. The flow of information for the cooperative tasks presented, is illustrated in Figure 49.

Several cooperative control experiments were performed. One particular cooperative experiment is illustrated in Figure 50. For this experiment, the goal position is located at (20m, 10m) in the local ENU frame. The maximum velocity constraint on each robot is  $v_{max} = 0.60$  m/s and the safety zone proximity is set at  $\delta_{prox} = 2$  m. In this experiment, there is an obstacle placed between the vehicles and the goal. The obstacle is placed at a distance of 9 m from the robots' starting positions and their objective is to avoid collisions with each other and the obstacle and arrive at the objective location. Also shown in Figure 50 is the potential field generated by the navigation function algorithm to direct the robots to the goal. Notice that the constraints for  $u_{max}$  and  $d_{eff}$  for the navigation function were removed, this was done to enable the vehicles to find their way to the goal from anywhere in the free space.





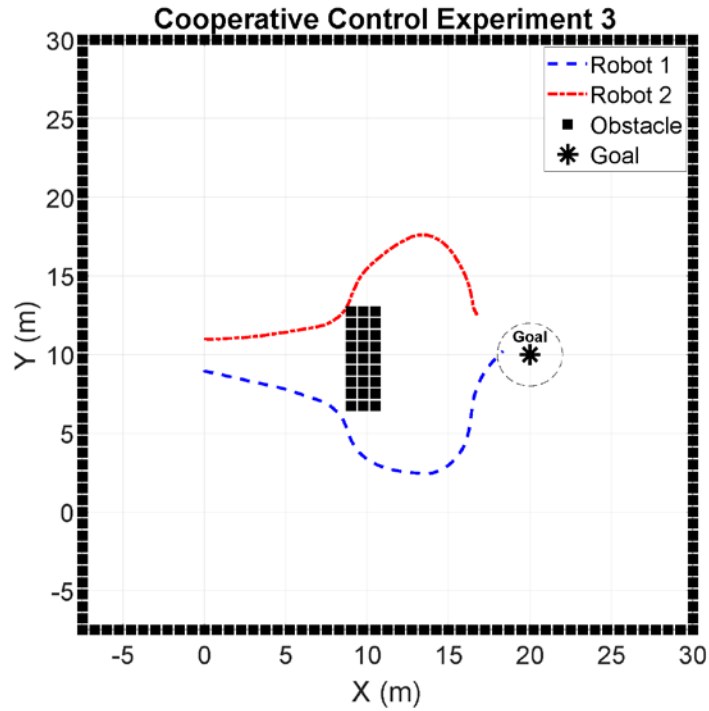
**Figure 49: Flow of information for cooperative control experiments**



**Figure 50: Setup for cooperative experiment**

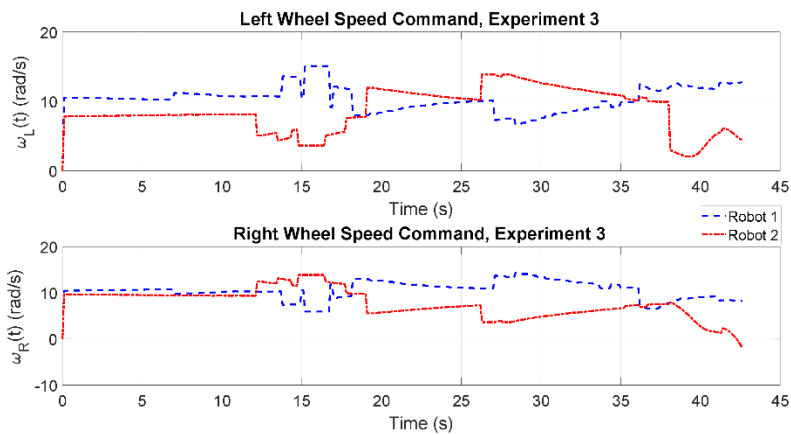
The resulting trajectories taken by the two robots is given in Figure 51. From this result, it is evident that both vehicles are repelled by each other at the start and gradually navigate around the obstacle. Robot 2 comes within close proximity to the corner obstacle. One drawback of this method is due to the cyber-physical system architecture employed, as there are inherent delays in the transmission of information over a 4G LTE mobile network. This leads to the delays in gathering positioning information and applying the guidance commands to the system. This issue also leads to close calls such as what is shown in Figure 51.

Approved for public release; distribution is unlimited.



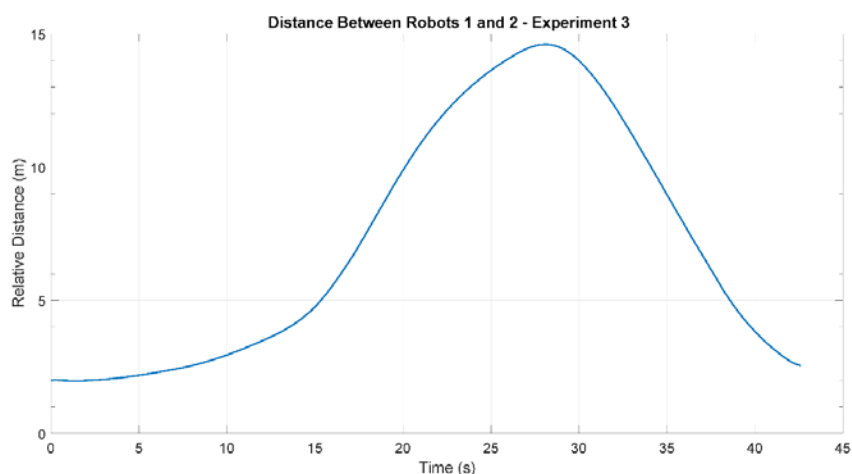
**Figure 51: Robot positions for cooperative experiment**

The resulting wheel speed commands generated by the cooperative APF guidance method is depicted in Figure 52. The behavior of the wheel speed command profiles for each robot suggest they are both maneuvering to both avoid colliding with each other as well as the obstacle present.



**Figure 52: Wheel speed commands for cooperative experiment**

The relative distance between each of the vehicles is shown in Figure 53. This result illustrates how the two robots start within a close proximity to one another and then gradually drift apart to go around the obstacle and eventually come back together and the end of the experiment.



**Figure 53: Relative distance between Robots for cooperative experiment**

## 6.0 CONCLUSIONS

All stated objectives and tasks outlined in the proposal were successfully attained. The project explored cooperation among multiple vehicles with uncertainties in measurements and time delays in the communication. The framework, and algorithms, were verified in several numerical simulations and validated in a CPS framework involving multiple ground rovers.

In relation to the above, the actual accomplishments included the following:

- Development of artificial potential functions-based navigation function applied to path planning for planetary rovers to navigate around obstacles.
- Development of a closed loop control algorithm to guide a planetary rover along a prescribed path.
- Development of a distributed time-delay framework for achieving consensus in a cooperative control scenario.
- Development of a planetary rover platform for conducting verification of algorithms in experiment.
- Development of a model-based navigation function algorithm applied to path planning problems for planetary rovers to navigate around obstacles.
- Development of a stable closed loop control algorithm to guide a planetary rover along a prescribed path.
- Development of a constrained nonlinear model predictive control algorithm to guide a planetary rover along a designed path.
- Development of a distributed potential function-based framework for achieving vehicle aggregation and social foraging in a cooperative scenario.
- Development of planetary rover platforms for conducting verification of algorithms in experiment.

Approved for public release; distribution is unlimited.

## REFERENCES

- [1] Ihle, I. A. F. and Fossen, T. I., "Formation Control of Marine Surface Craft Using Lagrange Multipliers," *Proceedings of the IEEE Conference on Decision and Control and the European Control Conference*, Seville, Spain, 2005.
- [2] Chung, S. J., Ahsun, U., and Slotine, J. J., "Application of Synchronization to Formation flying spacecraft: Lagrangian approach," *AIAA Journal of Guidance, Control and Dynamics*, **32** (2), pp. 512–426, 2009.
- [3] Lavretsky, E., "F-18 Autonomous Formation Flight Control Systems Design," *In Proceedings of AIAA Guidance, Navigation, and Control Conference and Exhibit, Journal of Guidance Control, Control and Dynamic, and Dynamics*, AIAA Paper 4757, 2002.
- [4] Patcher, M., "Tight Formation Flight Control," *AIAA Journal of Guidance, Control and Dynamics*, **24** (2), pp. 246–254, 2001.
- [5] Guibout, V. M. and Scheeres, D. J., "Spacecraft Formation Dynamics and Design," *AIAA, Journal of Guidance, Control and Dynamics*, **29** (1), pp. 121–133, 2006.
- [6] Ren, W. and Beard, R. W., "A Decentralized Scheme for Spacecraft Formation Flying via the Virtual Structure Approach," *Proceedings of the American Control Conference, American Automatic Control Council*, Evanston, IL, pp. 1746–1751, 2003.
- [7] Proud, A., Patcher, M., and D'Azzo, J., "Close Formation Control," *In Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, AIAA, Reston, VA, pp. 1231–1246, 1999.
- [8] Slotine, J. J. E., and Li, W., "Applied NonLinear Control," *Prentice Hall International Inc.*, London, UK, 2005.
- [9] Subbarao, K., "Nonlinear Pid-like Controllers for Rigid-body Attitude Stabilization," *The Journal of the Astronautical Sciences*, **52** (1 and 2), pp. 61–74, 2004.
- [10] Chung, S. J., and Slotine, J. J., "Cooperative Robot Control and Concurrent Synchronization of Lagrangian Systems," *IEEE Transactions on Robotics*, **25** (3), pp. 686–700, 2009.
- [11] Belanger, G. M., Ananyev, S., and Carpenter, J. R., "Decentralized Control of Satellite Clusters Under Limited Communication," *AIAA Journal of Guidance, Control and Dynamics*, **29** (1), pp. 134–145, 2006.
- [12] Ren, W., and Atkins, E. M., "Information Consensus in Multivehicle Cooperative Control," *IEEE Control Systems Magazine*, **27** (2), pp. 71–82, 2007.
- [13] Ren, W., "Formation Keeping and Attitude Alignment for Multiple Spacecraft through Local Interactions," *AIAA, Journal of Guidance, Control and Dynamics*, **30** (2), pp. 633–638, 2000.
- [14] Zou, Y., Pagilla, P. R., and Misawa, E. A., "Formation of a Group of Vehicles with Full Information using Constraint Forces," *ASME Journal of Dynamics Systems, Measurement, and Control*, **129** (5), pp. 654–661, 2007.
- [15] Murray, R. M., "Recent Research in Cooperative Control of Multivehicle Systems," *Journal of Dynamics Systems, Measurement, and Control*, **129** (5), pp. 571–583, 2007.
- [16] Leonard, N. E., and Fiorello, E., "Virtual Leader, Artificial Potentials and Coordinated Control of Groups," *Proceedings of 40th IEEE Conference on Decision and Control, Institute of Electrical and Electronics Engineers*, New York, NY, pp. 2968–2973, 2001.

- [17] Olfati-Saber, R., and Murray, R., “Distributed Cooperative Control of Multiple Vehicle Formations using Structural Potential Functions,” *15th IFAC World Congress, International Federation of Automatic Control*, Kidlington, Oxford, U. K., 2002.
- [18] Ogren, P., and Leonard, N. E., “Cooperative Control of Mobile Sensor Networks: Adaptive Gradient Climbing in a Distributed Environment,” *IEEE Transactions on Automatic Control*, **49** (8), pp. 1292–1302, 2004.
- [19] Romin, E., and Koditschek, D. E., “Exact Robot Navigation using Artificial Potential Function,” *IEEE Transactions on Robotics and Automation*, **8** (5), pp. 501–518, 1992.
- [20] Dimarogonas, D. V., Loizou, S. G., and Zavlanos, M. M., “A Feedback Stabilization and Collision Avoidance Scheme for Multiple Independent Non-point Agents,” *Automatica*, **42** (2), pp. 229–243, 2006.
- [21] Ren, W., “Multi-vehicle Consensus with Time-varying Reference State,” *Systems and Control Letters*, **56** (7-8), pp. 474–483, 2007.
- [22] Olfati-Saber, R., and Murray, R. M., “Consensus Problems in Networks of Agents with Switching Topology and Time Delays,” *IEEE Transactions on Automatic Control*, **49** (9), pp. 1520–1533, 2004.
- [23] Clough, B. T., “Metrics, Schmetrics! How the Heck do you Determine a UAV’s Autonomy Anyway,” URL: <http://www.dtic.mil/dtic/tr/fulltext/u2/a515926.pdf>, last modified Aug 2002. Accessed Jul 2019.
- [24] Huntsberger, T., Aghazarian, H., Cheng, Y., Baumgartner, E. T., Tun-stel, E., Leger, C., Trebi-Ollennu, A., and Schenker, P. S., “Rover Autonomy for Long Range Navigation and Science Data Acquisition on Planetary Surfaces,” in *Proceedings 2002 IEEE International Conference on Robotics and Automation* (Cat. No.02CH37292), **3**, pp. 3161–3168, 2002.
- [25] Ono, M., et al., “Risk-Aware Planetary Rover Operation: Autonomous Terrain Classification and Path Planning,” in *IEEE Aerospace Conference*, pp. 1–10, 2015.
- [26] Quadrelli, M. B., Wood, L. J., Riedel, J. E., McHenry, M. C., Aung, M., Cangahuala, L. A., Volpe, R. A., Beauchamp, P. M., and Cutts, J. A., “Guidance, Navigation, and Control Technology Assessment for Future Planetary Science Missions,” *Journal of Guidance, Control, and Dynamics*, **38** (7), pp. 1165–1186, 2015.
- [27] Howard, T. M. and Kelly, A., “Optimal Rough Terrain Trajectory Generation for Wheeled Mobile Robots,” *The International Journal of Robotics Research*, **26** (2), pp. 141–166, 2007.
- [28] Howard, T., Green, C., and Kelly, A., “Receding Horizon Model-predictive Control for Mobile Robot Navigation of Intricate Paths,” in *Field and Service Robotics: Results of the 7th International Conference*. Springer, Berlin, Heidelberg, pp. 69–78, 2010.
- [29] Lavelle, S. M., “Rapidly-exploring Random Trees,” URL: [http://www-symbiotic.cs.ou.edu/~fagg/classes/mobile\\_manipulation\\_2015/papers/LavKuf01.pdf](http://www-symbiotic.cs.ou.edu/~fagg/classes/mobile_manipulation_2015/papers/LavKuf01.pdf), last modified 1998. Accessed Jul 2019.
- [30] Karaman, S. and Frazzoli, E., “Incremental Sampling-based Algorithms for Optimal Motion Planning,” in *Proceedings of Robotics, Science and Systems*, Zaragoza, Spain, June 2010.
- [31] LaValle, S. M. and Kuffner Jr., J. J., “Randomized kinodynamic planning,” *The International Journal of Robotics Research*, **20** (5), pp. 378–400, 2001.

- [32] Hsu, D., Kindel, R., Latombe, J. C., and Rock, S., “Randomized Kinodynamic Motion Planning with Moving Obstacles,” *The International Journal of Robotics Research*, **21** (3), pp. 233–255, 2002.
- [33] Khatib, O., “Real-Time Obstacle Avoidance for Manipulators and Mobile Robots,” In *Autonomous Robot Vehicles*, Springer, New York, NY, pp. 396–404, 1990.
- [34] Latombe, J. C., “Robot Motion Planning,” *Springer Science & Business Media*, **124**, 2012.
- [35] Barraquand, J., Langlois, B., and Latombe, J. C., “Numerical Potential Field Techniques for Robot Path Planning,” *IEEE Transactions on Systems, Man and Cybernetics*, **22** (2), pp. 224–241, 1992.
- [36] [14] Rimon, E. and Koditschek, D. E., “Exact Robot Navigation Using Artificial Potential Functions,” *IEEE Transactions on Robotics and Automation*, **8** (5), pp. 501–518, 1992.
- [37] Wang, Y., Wang, D., and Zhu, S., “A New Navigation Function Based Decentralized Control of Multi-Vehicle Systems in Unknown Environments,” *Journal of Intelligent & Robotic Systems*, **87** (2), pp. 363–377, 2017.
- [38] Kowalczyk, W., Przybyla, M., and Kozlowski, K., “Set-Point Control of Mobile Robot with Obstacle Detection and Avoidance using Navigation Function-Experimental Verification,” *Journal of Intelligent & Robotic Systems*, **85** (3-4), pp. 539–552, 2017.
- [39] Kowalczyk, W., “Rapidly Converging Navigation Function Control for Differentially Driven Mobile Robots,” in *11th International Workshop on Robot Motion and Control (RoMoCo) IEEE*, Wasowo Palace, Poland, pp. 244–250, 2017.
- [40] Horowitz, M. B. and Burdick, J. W., “Optimal Navigation Functions for Non-Linear Stochastic Systems,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, pp. 224–231, 2014.
- [41] Connolly, C. I., Burns, J. B., and Weiss, R., “Path Planning Using Laplace’s Equation,” in *IEEE International Conference on Robotics and Automation*, **3**, pp. 2102–2106, 1990.
- [42] Masoud, A. A. and Bayoumi, M. M., “Robot Navigation Using the Vector Potential Approach,” in *IEEE International Conference on Robotics and Automation*, **1**, pp. 805–811, 1993.
- [43] Garrido, S., Moreno, L., Blanco, D., and Martin, F., “Smooth Path Planning for Non-Holonomic Robots Using Fast Marching,” in *IEEE International Conference on Mechatronics*, pp. 1–6, 2009.
- [44] Ralli, E. and Hirzinger, G., “Fast Path Planning for Robot Manipulators Using Numerical Potential Fields in the Configuration Space,” in *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems 94*, Advanced Robotic Systems and the Real World, IROS’94, **3**, pp. 1922–1929, 1994.
- [45] Wang, Y. and Cao, W., “A Global Path Planning Method for Mobile Robot Based on a Three-Dimensional-Like Map,” *Robotica*, **32** (4), pp. 611–624, 2014.
- [46] Brock, O., “Generating Robot Motion, The Integration of Planning and Execution,” *Ph.D. dissertation, Stanford University*, Stanford, CA, USA, aAI9961867, 2000.
- [47] Stentz, A., “The D\* Algorithm for Real-Time Planning of Optimal Traverses.” Carnegie-Mellon University Pittsburgh, PA Robotics Institute, 1994.
- [48] Stentz, A., “The Focussed D\* Algorithm for Real-time Replanning,” in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, **2**, San Francisco, CA, pp. 1652–1659, 1995.
- [49] LaValle, S. M., **Planning algorithms**, Cambridge University Press, New York, NY, 2006.

- [50] Nilsson, N. J., **Principles of artificial intelligence**, Morgan Kaufmann, San Francisco, CA, 2014.
- [51] Kwon W. H. and Han S. H., “Receding Horizon Control/Model Predictive Control for State Models,” *Springer Science & Business Media*, 2006.
- [52] [30] Cloutier J. R., “State-dependent Riccati equation techniques: An overview,” in *Proceedings of the 1997 American Control Conference*, **2**, pp. 932–936, 1997.
- [53] Cloutier, J. R. and Stansbery, D. T., “The Capabilities and Art of State-Dependent Riccati Equation-Based Design,” in *Proceedings of the 2002 American Control Conference IEEE*, **1**, pp. 86–91, 2002.
- [54] Cimen, T., “State-dependent Riccati Equation (sdre) Control, A Survey,” *IFAC Proceedings*, **41** (2), pp. 3761–3775, 2008.
- [55] Cimen, T., “Survey of State-Dependent Riccati Equation in Nonlinear Optimal Feedback Control Synthesis,” *Journal of Guidance, Control and Dynamics*, **35** (4), pp. 1025–1047, 2012.
- [56] Heydari, A. and Balakrishnan, S., “Path Planning Using a Novel Finite Horizon Suboptimal Controller,” *Journal of Guidance, Control, and Dynamics*, **36** (4), pp. 1210–1214, 2013.
- [57] Khamis, A. and Naidu, D., “Nonlinear Optimal Tracking Using Finite-Horizon State Dependent Riccati Equation (sdre),” in *Proceedings of the 4th International Conference on Circuits, Systems, Control, Signals (WSEAS)*, pp. 37–42, 2013.
- [58] Rawlings, J. B., “Tutorial Overview of Model Predictive Control,” *IEEE Control Systems*, **20** (3), pp. 38–52, 2000.
- [59] Klančar, G. and Škrjanc, I., “Tracking-error model-based predictive control for mobile robots in real time,” *Robotics and Autonomous Systems*, **55** (6), pp. 460–469, 2007.
- [60] Mayne, D. Q., Rawlings, J. B., Rao, C. V., and P. O. Scokaert, “Constrained Model Predictive Control: Stability and Optimality,” *Automatica*, **36** (6), pp. 789–814, 2000.
- [61] Primbs, J. A., “Nonlinear optimal control: A Receding Horizon Approach,” *Ph.D. Dissertation, California Institute of Technology*, 1999.
- [62] Jadbabaie, A., “Receding Horizon Control of Nonlinear Systems: A Control Lyapunov Function Approach,” *Ph.D. dissertation, California Institute of Technology*, 2001.
- [63] Sznaier, M. and Suarez, R., “Receding Horizon State Dependent Riccati Equations,” in *Proceedings of the 40th IEEE Conference on Decision and Control*, **4**, pp. 3832–3837, 2001.
- [64] Sznaier, M., Sua´rez, R., and Cloutier, J., “Suboptimal Control of Constrained Non-Linear Systems via Receding Horizon Constrained Control Lyapunov Functions,” *International Journal of Robust and Nonlinear Control*, **13** (3-4), pp. 247–259, 2003.
- [65] Gu, D. and Hu, H., “A Stabilizing Receding Horizon Regulator for Nonholonomic Mobile Robots,” *IEEE Transactions on Robotics*, **21** (5), pp. 1022–1028, 2005.
- [66] Gu, D. and Hu, H., “Receding Horizon Tracking Control of Wheeled Mobile Robots,” *IEEE Transactions on Control Systems Technology*, **14** (4), pp. 743–749, 2006.
- [67] Ru, P. and Subbarao, K., “Nonlinear Model Predictive Control for Unmanned Aerial Vehicles,” *Aerospace*, **4** (2), p. 31, 2017.
- [68] Dixon, W., Dawson, D., Zergeroglum, E., and Zhang, F., “Robust Tracking and Regulation Control for mobile robots,” in *Proceedings of the 1999 IEEE International Conference on Control Applications*, **2**, pp. 1015–1020, 1999.

- [69] Huang, J., Wen, C., Wang, W., and Jiang, Z. P., “Adaptive stabilization and tracking Control of a Nonholonomic Mobile Robot with Input Saturation and Disturbance,” *Systems and Control Letters*, **62** (3), pp. 234–241, 2013.
- [70] Wilson, D. G. and Robinett, I., “Robust Adaptive Backstepping Control for a Nonholonomic Mobile Robot,” in *IEEE International Conference on Systems, Man, and Cybernetics*, **5**, pp. 3241–3245, 2001.
- [71] Fierro, R. and Lewis, F. L., “Control of a Nonholonomic Mobile Robot: Back-Stepping Kinematics into Dynamics,” in *Proceedings of the 34th IEEE Conference on Decision and Control*, **4**, pp. 3805–3810, 1995.
- [72] Quillen, P., Subbarao, K., and Muñoz, J., “Guidance and Control of a Mobile Robot via Numerical Navigation Functions and Backstepping for Planetary Exploration Missions,” in *AIAA Space 2016*, no. 2016-5237, 2016.
- [73] Quillen, P., Muñoz, J., and Subbarao, K., “Path Planning to a Reachable State using Inverse Dynamics and Minimum Control Effort Based navigation functions,” in *2017 AAS/AIAA Astrodynamics Specialist Conference in Columbia River Gorge*, Stevenson, WA, paper AAS 17-849, August 2017.
- [74] Godbole, A., Murali, V., Quillen, P., and Subbarao, K., “Optimal Trajectory Design and Control of a Planetary Exploration Rover,” in *Advances in the Astronautical Sciences Spaceflight Mechanics 2017*, **160**, *27th AAS/AIAA Space Flight Mechanics Meeting*, AAS 17-481, Feb 2017.
- [75] Ostafew, C. J., Schoellig, A. P., and Barfoot, T. D., “Learning-based Non-Linear Model Predictive Control to Improve Vision-Based Mobile Robot Path-Tracking in Challenging Outdoor Environments,” in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4029–4036, 2014.
- [76] Martins, F. N., Celeste, W. C., Carelli, R., Sarcinelli-Filho, M., and Bastos-Filho, T. F., “An Adaptive Dynamic Controller for Autonomous Mobile robot trajectory tracking,” *Control Engineering Practice*, **6** (11), pp. 1354–1363, 2008.
- [77] Buccieri, D., Perritaz, D., Mullhaupt, P., Jiang, Z. P., and Bonvin, D., “Velocity- scheduling control for a unicycle mobile robot: Theory and experiments,” *IEEE Transactions on Robotics*, **25** (2), pp. 451–458, 2009.
- [78] Lewis, F. L., Zhang, H., Hengster-Movric, K., and Das, A., “Cooperative Control of Multi-Agent Systems: Optimal and Adaptive Design Approaches,” *Springer Science & Business Media*, 2013.
- [79] Gazi, V. and Passino, K. M., “A Class of Attractions/Repulsion Functions for Stable Swarm Aggregations,” *International Journal of Control*, **77** (18), pp. 1567–1579, 2004.
- [80] Gazi, V., FI dan, B., Hanay, Y. S., and K”oksal, I, “Aggregation, Foraging, and Formation Control of Swarms with Non-Holonomic Agents using Potential Functions and Sliding Mode Techniques,” *Turkish Journal of Electrical Engineering & Computer Sciences*, **15** (2), pp. 149–168, 2007.
- [81] Leonard, N. E., and Fiorelli, E., “Virtual Leaders, Artificial Potentials and Coordinated Control of Groups,” in *Proceedings of the 40th IEEE Conference on Decision and Control*, **3**, pp. 2968–2973, 2001.
- [82] Saber, R. O. and Murray, R. M., “Flocking with Obstacle Avoidance: Cooperation with Limited Communication in Mobile Networks,” in *Proceedings of 42nd IEEE Conference on Decision and Control*, **2**, pp. 2022–2028, 2003.



- [83] Do, K. D., “Formation Tracking Control of Unicycle-Type Mobile Robots with Limited Sensing Ranges,” *IEEE Transactions on Control Systems Technology*, **16** (3), pp. 527–538, 2008.
- [84] Kim, D. H., Wang, H., and Shin, S., “Decentralized Control of Autonomous Swarm Systems Using Artificial Potential Functions: Analytical Design Guidelines,” *Journal of Intelligent and Robotic Systems*, **45** (4), pp. 369–394, 2006.
- [85] De Vries, E. and Subbarao, K., “Cooperative Control of Swarms of Unmanned Aerial Vehicles,” in *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, Orlando, USA, AIAA 2011-78, American Institute of Aeronautics and Astronautics (AIAA), 4-7 January 2011.
- [86] Lewis, F. L., Vrabie, D., and Syrmos, V. L., **Optimal Control**, John Wiley & Sons, Inc., New York, 2012.
- [87] Chen, C. T., **Linear System Theory and Design**, Oxford University Press, Inc., 1995.
- [88] Williams, R. L., Lawrence, D. A., et al., **Linear State-Space Control Systems**, John Wiley & Sons Inc., 2007.
- [89] Slotine, J.-J. E. and Li, W., **Applied Nonlinear Control**, Prentice hall Englewood Cliffs, NJ, **199**, 1991.
- [90] Wright, S. J. and Nocedal, J., “Numerical optimization,” *Springer Science*, 1999.
- [91] Removed for administrative reasons.
- [92] Removed for administrative reasons.
- [93] Removed for administrative reasons.
- [94] Nair, S., Subbarao, K., “Attitude Control of Spacecraft Formations subject to Distributed Communication Delays,” *27th AAS/AIAA Space Flight Mechanics Meeting*, San Antonio, TX, USA, February 2017.
- [95] Quillen, P., Subbarao, K., “Minimum Control Effort Based Path Planning and Nonlinear Guidance for Autonomous Mobile Robots,” *International Journal of Advanced Robotic Systems (Sage Publications)*, **15** (6), November, 2018.

## LIST OF SYMBOLS, ABBREVIATIONS, AND ACRONYMS

<b>Acronym/Abbreviation</b>	<b>Description</b>
CPS	Cyber Physical System
ROS	Robot Operating System
RRT	Rapidly Exploring Randomized Trees
APF	Artificial Potential Field/Function
NF	Navigation Function
PDE	Partial Differential Equation
HJB	Hamilton Jacobi Bellman
MPC	Model Predictive Control
NMPC	Nonlinear Model Predictive Control
RHC	Receding Horizon Control
SDC	State Dependent Coefficient
SDRE	State Dependent Riccati Equation
LQR	Linear Quadratic Regulator
CLF	Control Lyapunov Function
MCE	Minimum Control Effort
ID	Inverse Dynamics
ENU	East North UP
GPS	Global Positioning System
IMU	Inertial Measurement Unit
ZOH	Zero Order Hold
SDDARE	State Dependent Discrete Riccati Equation
LMI	Linear Matrix Inequality (Inequalities)
MRP	Modified Rodrigues Parameters
PWM	Pulse Width Modulation
GNC	Guidance, Navigation, and Control

## DISTRIBUTIONLIST

DTIC/OCP

8725 John J. Kingman Rd, Suite 0944

Ft Belvoir, VA 22060-6218

1 cy

AFRL/RVIL

Kirtland AFB, NM 87117-5776

1 cy

Official Record Copy

AFRL/RVSV/Christopher Petersen

1 cy

(This page intentionally left blank)