NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

ENERGY OPTIMAL TRAJECTORY PLANNING OF MULTICOPTER UAVS IN SUPPORT OF MARINE CORPS EXPEDITIONARY ADVANCED BASE OPERATIONS

by

Matthew Modelo

June 2021

Thesis Advisor: Vladimir N. Dobrokhodov
Co-Advisor: Kevin D. Jones

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

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 instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the 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 Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2021	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE
ENERGY OPTIMAL TRAJECTORY PLANNING OF MULTICOPTER UAVS IN SUPPORT OF MARINE CORPS EXPEDITIONARY ADVANCED BASE OPERATIONS

5. FUNDING NUMBERS

6. AUTHOR(S) Matthew Modelo

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

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

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

Unmanned Aerial Vehicles (UAV) are becoming increasingly important in military operations, and tactical resupply for the United States Marine Corps (USMC) is one such example of an area that is calling for the benefits offered by UAVs as resupply vehicles. This study considers a generic multicopter as a delivery system designed to travel to multiple USMC units in one trip by minimizing energy consumption in the presence of severe time-varying wind. Pontrayagin's Maximum Principle and other known optimal control theories are used to formulate the trajectory optimization task as a boundary value problem. The goal for the algorithm is to take in a set of boundary conditions as well as the weather forecast (in the form of three-dimensional wind, valid for the duration of the mission) as necessary information for determining the optimal airspeed and bank angle that should be commanded to the system in order to send it along an energy-optimal route. The objective is achieved by applying an energy-performance model of a multicopter to a Boundary Value Problem (BVP) solver developed in MATLAB and analyzing the energy-optimality of the result. Multiple principles of optimal control theory are used throughout the solving process to predict certain conditions for optimality and determine the level of success of the results.

14. SUBJECT TERMS optimal control, energy expense, multicopter, boundary value problem, vehicle routing problem, Pontryagin's Maximum Principle, multiple traveling salesman problem, unmanned aerial vehicle	**15. NUMBER OF PAGES** 87
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

ENERGY OPTIMAL TRAJECTORY PLANNING OF MULTICOPTER UAVS IN SUPPORT OF MARINE CORPS EXPEDITIONARY ADVANCED BASE OPERATIONS

Matthew Modelo
Ensign, United States Navy
BS, United States Naval Academy, 2020

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ASTRONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 2021

Approved by: Vladimir N. Dobrokhodov
Advisor

Kevin D. Jones
Co-Advisor

Garth V. Hobson
Chair, Department of Mechanical and Aerospace Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Unmanned Aerial Vehicles (UAV) are becoming increasingly important in military operations, and tactical resupply for the United States Marine Corps (USMC) is one such example of an area that is calling for the benefits offered by UAVs as resupply vehicles. This study considers a generic multicopter as a delivery system designed to travel to multiple USMC units in one trip by minimizing energy consumption in the presence of severe time-varying wind. Pontrayagin's Maximum Principle and other known optimal control theories are used to formulate the trajectory optimization task as a boundary value problem. The goal for the algorithm is to take in a set of boundary conditions as well as the weather forecast (in the form of three-dimensional wind, valid for the duration of the mission) as necessary information for determining the optimal airspeed and bank angle that should be commanded to the system in order to send it along an energy-optimal route. The objective is achieved by applying an energy-performance model of a multicopter to a Boundary Value Problem (BVP) solver developed in MATLAB and analyzing the energy-optimality of the result. Multiple principles of optimal control theory are used throughout the solving process to predict certain conditions for optimality and determine the level of success of the results.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

List of Acronyms and Abbreviations

BVP	Boundary Value Problem
COAMPS	Coupled Ocean/Atmosphere Mesoscale Prediction System
HEC	Hamiltonian Evolution Condition
HMC	Hamiltonian Minimization Condition
HVC	Hamiltonian Value Condition
NED	North-East-Down
ODEs	ordinary differential equations
PMP	Pontryagin's Maximum Principle
UAVs	Unmanned Aerial Vehicles
USMC	United States Marine Corps
VRP	Vehicle Routing Problem

THIS PAGE INTENTIONALLY LEFT BLANK

Nomenclature

Variables

x	Multicopter position in x-direction	m/s
y	Multicopter position in y-direction	m/s
ψ	Multicopter heading angle	$^\circ$
χ	Ground heading angle	$^\circ$
χ_C	Crab angle	$^\circ$
θ	Pitch angle	$^\circ$
V_a	Airspeed	m/s
V_g	Ground speed	m/s
V_w	Wind Velocity	m/s
ϕ	Bank angle	$^\circ$
P_{req}	Required power for constant altitude, forward flight	W
T	Rotor thrust	N
t	Time	sec

UAV Parameters

n	Number of rotors on multicopter	
W	Multicopter weight	N
ρ	Air density	kg/m^3
R	Rotor radius	m

A	Rotor disc area	m^2
Ω	Blade angular velocity	rad/sec
s	Rotor solidity ratio	
d_0	Fuselage drag ratio	
k	Incremental correction factor to induced power	
K	Thrust to weight ratio	
t_{cD}	Thrust coefficient	
v_{i0}	Mean induced velocity of rotor in forward flight	m/s
v_0	Mean induced velocity of rotor in hover	m/s
λ_i	Mean induced velocity normalized by tip speed	m/s
δ	Profile drag coefficient	
μ	Advance ratio	
$\hat{V}$	Forward speed normalized by tip speed	

Physics Constants

g	Gravitational constant	m/s^2

Acknowledgments

This study would not have been possible without the immense help of my thesis advisor, Dr. Dobrokhodov, who has supported me in overcoming obstacles throughout the process. I would also like to thank Dr. Ross and Dr. Karpenko for their guidance in navigating optimal control theory.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

At multiple stages of global conflict, it will be increasingly necessary for United States Marine Corps (USMC) troops to deploy in an expeditionary fashion in different types of terrain, thus requiring a higher level of stealth and faster resupplying logistics. Unmanned Aerial Vehicles (UAVs) help support this mission by leaving behind a smaller footprint and offering the advantage of intelligent autonomy. The fact that many of these drones, some of which will be multicopter systems, will have to prioritize between multiple drop-off sites as well as battle against environmental conditions (time-varying three dimensional weather and terrain) presents many challenges. While optimal control models for fixed-wing UAVs have been well-developed, achieving the same goal for a multicopter system requires additional work due to the fact that its flight dynamics and energy expenditure are quite different. In the envisioned battle-space logistics, the goal of the UAV is to successfully travel to multiple required destinations while utilizing constrained energy. In addition to this task, the system should also be able to calculate optimal routes to and from each destination that harvest environmental weather energy, predominantly wind, in such a way to minimize fuel consumption whilst staying within the dynamic constraints of the aircraft. The specific goal of this thesis is to create an energy expense model of the multicopter and use it as a basis to formulate and solve the corresponding Boundary Value Problem (BVP) for its energy-optimal routing. The BVP should be robust to the variations of the boundary conditions and the parameters of a generic multicopter in diverse wind conditions.

In this thesis study, two unique algorithms are applied in solving the BVP of a typical delivery profile in order to produce a diverse solution set. The solutions will then be compared to promote a better understanding of the overall problem and offer insight into the determination of the most efficient solving method.

1.1 Background Information

The application of dynamic optimization theory can greatly benefit the task of optimal guidance of the multicopter UAV in various military applications. The vehicles to be used

for re-supply operations are relatively small, and thus have limited fuel and payload capacity. Considering a finite energy density of various sources (liquid fuels, compressed hydrogen, modern Li-Po materials) immediately constraints the envelop of operation. Therefore, not only the choice of the fuel, but also the actual mission control become interdependent and important factors of real life operation of UAVs. Additionally, the UAV must travel to multiple destinations and shape its path depending on which drop-off sites claim the highest priority. Considering these constraints, route determination for this mission is paramount, and the use of dynamic optimization theory helps to affirm that each flight will burn the least amount of energy possible. If a re-supply UAV gets slotted for a route that contains seven destinations, but it runs out of fuel after passing through the sixth drop-off site, the personnel at that last location must wait an entire delivery cycle before receiving their supplies. A truly optimal route that takes advantage of wind and minimizes energy-use can significantly affect the framework of USMC logistics. Additionally, a minimal-energy algorithm benefits companies who use multi-rotor UAVs for commercial purposes, such as agriculture, transportation, and long-endurance flights.

1.2 Literature Review

A fundamental aspect of the multicopter's optimal control problem is the energy expenditure model for a generic rotary UAV. In order for the algorithm to be able to solve for an optimal route, it must have an idea of how much power will be consumed during flight as a function of relevant control inputs. The technical paper "Energy Minimization for Wireless Communication with Rotary-Wing UAV" by Yong Zeng, Jie Xu, and Rui Zhang [1] derives such an expression that can be directly applied to the problem formulation for this study. The paper details what features of a rotary UAV contribute to its power consumption during level flight as well as the significant effect of two control variables, airspeed and pitch angle. Appendix B also expands on the need to consider the profile, induced, and parasite components of energy consumption and how they can be properly modeled in the overall expression.

The paper "Smooth Trajectory Optimization in Wind: First Results on a Full-Scale Helicopter" by Vishal Dugar, Sanjiban Cloudhury, and Sebastian Scherer [2] develops an algorithm to solve for an optimal path of an autonomous helicopter in the presence of wind, similar to the problem at hand. It also puts forth a set of dynamic equations to describe the

motion of the rotary vehicles and takes into account real-life constraints, such as a flight path corridor to be followed (in order to keep the aircraft out of restricted airspace). The authors also discuss how their solution is further refined by applying velocity optimization to enhance the airspeed control at specific points and spline fitting to smooth sections that introduce jerk limits. While the current task does not necessarily have to produce a solution that is as polished, the work in [2] can still be referenced in the steps of developing the dynamic model of the system and analyzing the effect of wind on the optimal path.

Lastly, the thesis written by Capt Adam Jatho, USMC [3] directly deals with the optimization of UAV delivery flight paths in support of Marine Corps logistic operations. While it is oriented towards the multiple vehicle routing problem of directing the UAVs along a multi-node path while prioritizing amongst the given nodes, the paper also touches on the formulation of the mission's BVP. Capt Jatho's work serves as the basis for further optimization by considering a new class of multicopter UAVs. The paper provides a skeleton for how the BVP should be structured — it lays out the dynamic equations, minimal energy cost function, and Hamiltonian for the flight of a UAV in the presence of wind. However, this implementation is specific to a fixed-wing UAV. Multicopters possess a different set of dynamic equations and energy profile, presenting a new challenge for the logistics task. Nevertheless, [3] provides strong contextual background on what ultimately needs to be accomplished for this thesis.

1.3 Boundary Value Problems

A boundary value problem models the motion of any system through a finite state space from an initial state to a final state. The goal of the BVP is to optimize the dynamical motion of the given system with respect to any constraints that pertain to the problem. Some common features of a simulation that might be subject to minimization in a BVP are time spent, energy expended, distance travelled, thrust applied, and force exerted. In addition to constraints, boundary conditions, and knowledge of the optimization cost, a BVP must also contain a set of differential equations to describe the dynamic and kinematic motion of the system. For this thesis study, the only parameter subject to optimization for the BVP is the energy expenditure of the multicopter.

1.4 System Analysis

While a BVP solves for optimal trajectories between each unit during flight, this algorithm is only one part of the UAV's logistics mission planning. Figure 1.1 shows the functions of the system as it engages in its mission profile.

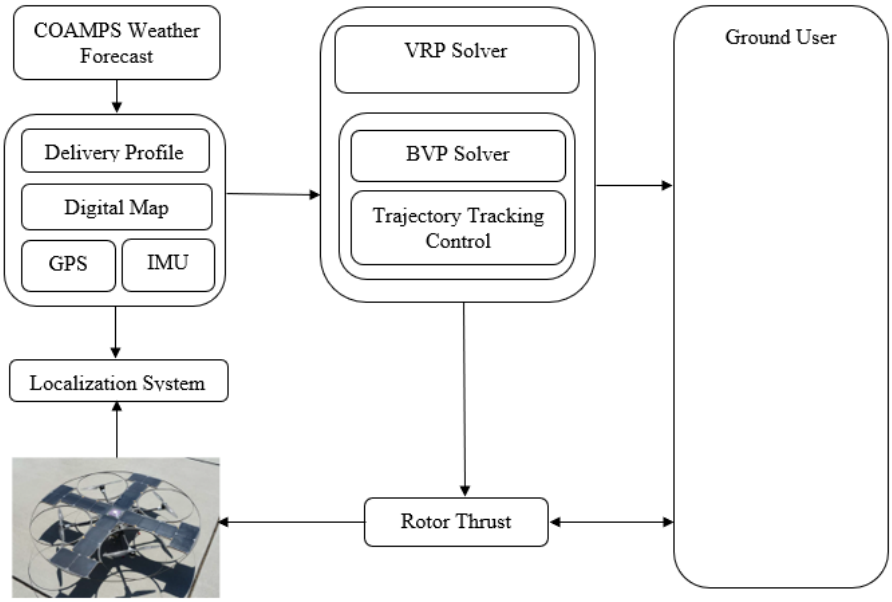

Figure 1.1. Multicopter UAV System Block Diagram.

The picture of a quad-rotor on the bottom left area of the figure represents an example of what a generic multicopter for this mission might look like and refers to the "Dynamic Modeling" aspect of the system. The dynamics equations for a generic multicopter UAV are known and included in this picture. The Coupled Ocean/Atmosphere Mesoscale Prediction System (COAMPS) Weather Forecast [4] is the model that provides real-time weather data to ultimately be fed to the BVP solver. It contains wind vectors that will serve as inputs to the optimal control problem. The delivery profile refers to the "Higher-Level Tasking" of the multicopter's mission - to deliver supplies to units at different locations along an optimal route. For each individual mission, the delivery profile will tell the UAV which units it must deliver supplies to and which claim the highest priority. This profile is crucial for the UAV to determine the optimal path, and which drop-off sites should be visited first. While developing an algorithm for this "Multiple Traveling Salesman Problem" is beyond

the scope of this thesis, it is still important to recognize that it is part of the greater mission of the UAV. The digital map provides coordinates of the delivery sites in order for the UAV to be able to "check them off" as it arrives at each one. The multicopter also has a GPS to provide a footprint of the multicopter's ground track and an IMU on board to provide heading with respect to due North. This information is fed to the localization system, which gives a full estimation of the UAV's position and heading in the inertial frame. The "Trajectory Tracking Control" essentially serves as a correcting mechanism to bring the UAV back on its desired path, should it veer off for any reason. The rotor thrust represents the "Low-Level Actuator Control" of the system. A multicopter must alter its path by varying the amount of thrust provided to each of its motors, thus commencing a roll maneuver. Ultimately, this entire mission will be initiated by a Marine on the ground, or even multiple units within reasonable distance of each other, requesting any type of resupply.

1.5 Implementation of the VRP Solver

The Vehicle Routing Problem (VRP) solver analyzes the delivery profile holistically and is responsible for carrying out the task of creating an optimal delivery route with multiple nodes. The onboard algorithm then develops a cost matrix of all possible combinations for the order of units to visit during its mission. This cost matrix assigns a certain penalty to each combination of flight paths depending on their energy-cost of travel and level of priority. For each of these multi-node journeys generated by the VRP solver, the BVP solver is responsible for providing the energy cost and the corresponding energy-optimal trajectory between the nodes. Due to the fact that the VRP solver requires a large amount of calculations for all possible optimal trajectories that must be continuously updated as the weather changes, it is crucial that the BVP solver executes its task with fast computation time. By working in conjunction with the VRP algorithm, the BVP solver then deals with the actual "Trajectory Planning" portion of the problem. It takes the wind vectors from the COAMPS forecast and the current dynamical state as inputs and solves for a minimal-energy route between *each* node in the delivery profile. The approach to developing an algorithm that solves the BVP in a timely manner is the main focus of this thesis and is discussed in the next chapter.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Methodology

The approach to solving the BVP involves determining the dynamics of the vehicle states and power expenditure of a generic multicopter and incorporating this information into a numerical solver. The BVP is formulated and solved using two distinct numerical approaches in MATLAB. The first method uses a subprogram called DIDO, which is able to iterate through the feasible solution space without the need to define the optimal controls in analytical form. The second involves the application of MATLAB's internal BVP-solving functions, $bvp4c$ and $bvp5c$, which is a more complex procedure that requires the analytical expressions of the optimal speed to fly and bank angle controls. The benefits of both approaches are analyzed and the optimality of the solutions are compared in the following chapters.

2.1 Introduction of System Dynamics and Coordinate Frames

The system dynamics are derived by analyzing the relationship among the UAV's Euler angles and the velocity vectors in the presence of wind. The following kinematic diagram, Figure 2.1, describes the steady state flight of a multicopter in the horizontal plane at constant altitude. Two coordinate frames, body and inertial, are in place to help connect the wind influence with the optimal guidance that is solved in an inertial North-East-Down (NED) frame.

Figure 2.1 shows how the heading of the multicopter relates to both the body frame and inertial frame. The inertial frame is defined as the NED frame, where the x-axis points due North, the y-axis points due East, and the z-axis points toward nadir, completing the right-handed orthogonal triad. The body frame is oriented about the multicopter and its instantaneous heading - the x-axis points out of the "nose" (most multicopters do not have physical noses but the term is used for visualization purposes), the y-axis points out of the "right wing", and the z-axis points out of the "belly" of the multicopter, completing the orthogonal triad.

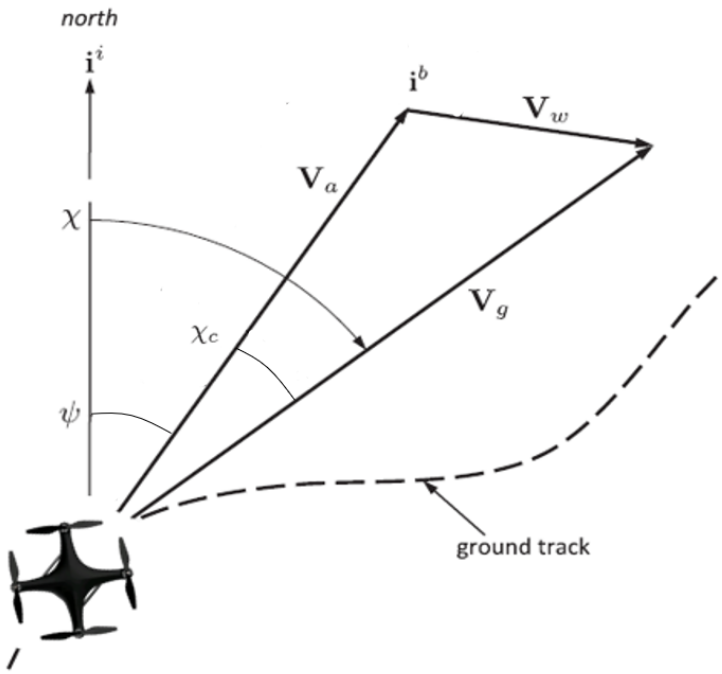

Figure 2.1. Kinematics of a Generic Multicopter Flight in the 2D Plane. Adapted from [5]

Coordinate Frames

F_b = body frame
F_i = inertial frame, defined by North-East-Down triad

The angle ψ is the body heading of the multicopter with respect to due North; it is assumed to be driven by the onboard autopilot according to the provided reference and the measurements of onboard IMU and GPS. It can be observed in Figure 2.1 that this angle also represents a yawing Euler angle between the inertial and body frames. In turn, the angle χ is the difference between the ground speed vector V_g and due North; it is assumed to be given by the onboard GPS. The angle χ_C is the crab angle - the difference between the $\bar{V}_a$ and the $\bar{V}_g$ directions.

In the case of fixed-wing UAV it represents the aircraft maneuver required to compensate the

lateral components of wind during flight. This is due to the fact that the wind causes fixed-wing aircraft, which are not symmetrical about their lateral axes, to drift in an unwanted direction. Pointing the nose slightly in the direction from which the wind is blowing allows the fixed-wing aircraft to maintain its airspeed vector.

Multicopter UAVs, on the other hand, are typically symmetrical about both their longitudinal and lateral axes - this eliminates the need to crab in the presence of wind. Rather, multicopters orient their heading toward the desired direction and command the proper bank to account for the wind. Considering this fundamental difference between fixed-wing and multicopter UAV flight dynamics as well as the fact that the onboard GPS can provide a footprint of the multicopter's ground track in the inertial frame, the ground heading angle does not need to be modeled for this problem. Figure 2.2 helps to visualize the three Euler angles of a multicopter and how they relate to the optimal control problem.

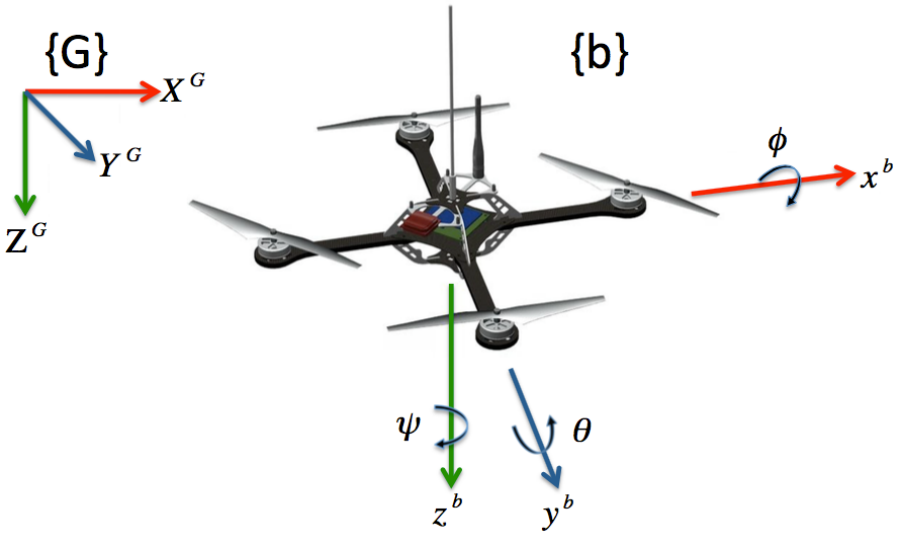

Figure 2.2. Euler Angles for a Generic Multicopter. Source: [6]

In Figure 2.2, the pitch angle θ controls the forward motion of the multicopter and the bank angle ϕ controls the lateral motion of the system. Within the scope of the this project we assume that the lifting force from all rotors acts parallel to the z-axis of the multicopter (in the body frame), and the yaw control is managed by the autopilot altering the rotational and directional speed of the individual rotors, typically using the differential control. Since the

typical pitch angles of a heavy-lifting multicopter are likely to be within 10-15 degrees, the $cos()$ of this angle will be close to 1 and the $sin()$ approximated by the angle itself. The lateral bank angle ϕ follows the desired control reference as the multicopter is required to bank in order to change the lateral direction of flight.

2.2 Problem Assumptions

As stated in Section 2.1, the value of the total forward inclination angle that combines the longitudinal "pitch" and the lateral "bank" angles is small, therefore allowing its longitudinal power-loss effect to be controlled by the autopilot while keeping the lateral effect accounted for by the optimal control task. For multicopters that are designed to fly with great speeds and minimize time-of-flight, a relatively large forward pitch angle is required to attain the proper airspeed. However, the UAV in this mission is not prioritizing speed, but rather energy consumption, and will only use small pitch angles to reach its desired airspeed, which is not to exceed 20 m/s. Additionally, the power requirements of the multicopter during flight are broken up into different components, but with the assumption that there is an autopilot on board, the multicopter should be able to conduct the constant altitude turns without losing energy efficiency to attitude maintenance. While multicopters rarely have airspeed sensors onboard, we assume that it is always possible to calibrate the forward flying speed vs the pitch angle in a no-wind environment that naturally represents the longitudinal airspeed control of the autopilot. It is also assumed that the autopilot has the ability to maintain desired airspeed and align the nose of the aircraft in the direction of the flight path while conducting a maneuver. As touched on in Section 2.1, the autopilot is also capable of inducing a yawing maneuver in either direction by the altering the direction and speed of any of the motors. Essentially, the assumption of the onboard autopilot allows the goal of minimal energy consumption to be achieved by controlling the airspeed and bank angle throughout the trajectory. For this problem formulation, it is assumed that the multicopter only moves in 2-D space (x-y for body frame and N-E for inertial frame), meaning that the effects of wind in the z-direction are not considered. The assumption was also stated in Section 2.1 that the onboard GPS has the capability to provide a footprint of the multicopter's ground track with respect to the inertial frame. Therefore, the ground heading angle χ does not need to be modeled for this problem.

Overall, the practical objective of the energy optimal control problem is to find the path that

requires minimum energy expenditure for the route when it is followed with the optimal airspeed. The trajectory at the hardware implementation level will be discretized by a set of way-points, and along with the optimal airspeed they will be followed by a commercial autopilot.

2.3 Definition of State and Control Variables

Three state variables are used to describe the movement of the multicopter throughout its delivery mission. Displacement in the x and y direction provide exact coordinates of the UAV, and the multicopter heading angle ψ specifies the direction the vehicle is travelling with respect to due North. The trajectories of these three state variables offer an accurate determination of the system's behavior during flight.

State Variables

x = distance in the direction of the "nose" of the multicopter (meters, F_i)
y = distance in the direction for the "right wing" of the multicopter (meters, F_i)
ψ = multicopter heading, a.k.a. one of the Euler angles (radians, F_i)

The state variables are examined in the inertial frame in order for the system dynamics to remain compatible with the onboard GPS readings. The control variables are the physical, aerodynamic contributions to the motion of the multicopter and are referred to in the body frame.

Control Variables

ϕ = bank angle (deg, F_b)
V_a = airspeed (m/s, F_b)

Design Limits

$-30^\circ < \phi < 30^\circ$
$4\ m/s < V_a < 20\ m/s$

The UAV's bank angle, ϕ and commanded airspeed, V_a, are co-dependent on each other and significantly affect the trajectories of the state variables. The onboard autopilot sets a forward pitch angle (and thus a commanded airspeed calibrated to that maneuver) and the processing unit follows the optimal bank angle reference to minimize energy consumption in the given conditions. For this thesis study and to model the specifications of a generic multicopter, design limits of 30 degrees and 20 m/s are applied to the model; these numbers are certainly notional and merely represent an envisioned operational range. The procedure of how the system shall determine the optimal values of these controls is one of the primary challenges of this boundary value problem.

2.4 State Dynamics Equations

The textbook *Small Unmanned Aircraft* by Randal Beard and Timothy McLain [5] provides multiple high-fidelity design models for UAVs with autopilot capabilities. While the book is oriented towards fixed-wing UAVs, its kinematic and design models can be applied to the flight of a generic multicopter with appropriate modifications.

A high-fidelity flight performance model intended for the flight behavior of a generic multicopter was obtained from [5] for the application of this study. By applying the problem assumptions listed in Section 2.2 to these equations, which can be referred to in Appendix C, a proper medium-fidelity model was developed to fit the task. Equations (2.1) and (2.2) demonstrate that the velocities in the x and y-direction of the inertial frame are written in terms of the airspeed and heading, and the turn rate is a function of both controls, airspeed and bank angle:

$$\dot{x} = V_a \cdot \cos\psi + W_x(x, y, t) \tag{2.1}$$

$$\dot{y} = V_a \cdot \sin\psi + W_y(x, y, t) \tag{2.2}$$

$$\dot{\psi} = \frac{g}{V_a} \cdot \tan\phi \tag{2.3}$$

Equations (2.1) and (2.2) show that the wind vector may be a function of the position of the multicopter in 2-D space as well as time. In the operational environment, the COAMPS

forecast [4] will provide a tabulated data set of wind vectors for the UAV's given flight corridor. The directions and magnitudes of the wind vectors in this profile of data will vary greatly as wind envelopes may change with geographic location. Due to this unpredictability of the weather forecast in an effort to verify the solution to the optimal control problem, a simple wind gradient pattern will be chosen, shown by Equations (2.4) and (2.5):

$$W_x(x, y, t) = -h_y \cdot y \tag{2.4}$$

$$W_y(x, y, t) = 0 \tag{2.5}$$

where h_y is a constant parameter. Equations (2.4) and (2.5) state that the wind vector at any point in time only blows in the x-direction, and that it is only a function of the position in the y-direction. A visual representation of this wind gradient is provided in Figure 2.3.

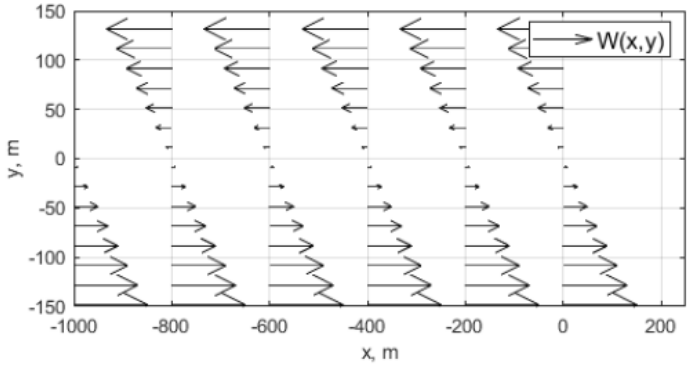

Figure 2.3. Wind Gradient To Be Used for Dynamical Equations. Source: [7]

The specific choice of this wind gradient allows the solution to be further verified by the fact that the multicopter should choose a path that takes advantage of tailwind and avoids headwind, making the solution-validation process straightforward and convenient. Thus, Equations (2.1) and (2.2) can be viewed as such:

$$\dot{x} = V_a \cdot \cos\psi - h_y \cdot y \tag{2.6}$$

$$\dot{y} = V_a \cdot \sin\psi \tag{2.7}$$

The ground speed of the multicopter can also be written as Equation (2.8) in terms of these

two velocities:

$$V_g = \sqrt{\dot{x}^2 + \dot{y}^2} \tag{2.8}$$

Knowing the effect of wind on airspeed helps to understand the behavior of another state variable, multicopter heading (ψ). The relationship between wind velocity, airspeed, and ground speed vectors can be written as such in Equation (2.9):

$$\bar{V}_a = \bar{V}_g - \bar{V}_w \tag{2.9}$$

The turn rate modeled by Equation (2.3) assumes that the UAV is equipped with an autopilot that implements a feedback loop that completely eliminates the non-zero crab angle during the flight in cross-wind conditions. The application of this assumption in deriving Equation (2.3) can be found in Appendix A.

2.5 Co-state Variables

With the 3 degree-of-freedom dynamic model established in Section 2.4, the following co-state vector is established for the problem and displayed by Equation (2.10).

$$\lambda = \begin{bmatrix} \lambda_x \\ \lambda_y \\ \lambda_\psi \end{bmatrix} \tag{2.10}$$

A co-state vector is useful in modeling the system in that it connects all of the state variables to common units of interest. In this boundary value problem, the co-states serve to balance the state variables as the total cost to fly, or the multicopter's power loss, varies in different wind conditions.

2.6 Cost Functional Modeling

An equation for the running power consumption of the multicopter during its 2-D flight must be integrated into the cost function of the BVP. Only scenarios where resupply is

not an emergency are being considered, so time is not a primary concern in the problem. Therefore, an endpoint cost, or a parameter for the final time, is not included. The power consumption of the system serves as the running cost and shall be integrated with respect to time in the problem. The following derivation of the energy expenditure model is based on the actuator disk and blade element theory that is provided in the Rotary-Wing UAV research paper by Zeng, et al [1]. Figure 2.4 helps to visualize the longitudinal forces acting on the UAV point mass as the power equation is derived with respect to these forces. The full-length derivation of the required power for flight can also be referred to in Appendix B.

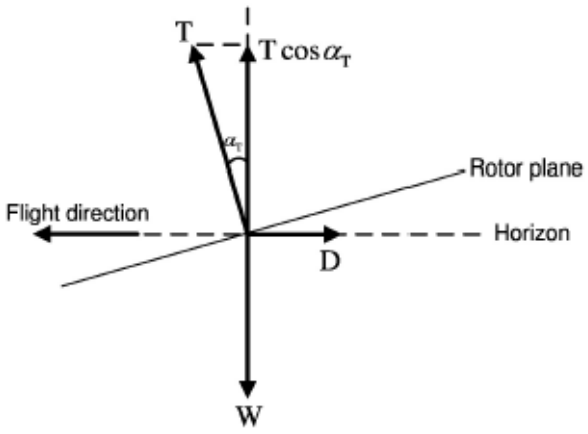

Figure 2.4. Longitudinal Forces Acting on the Multicopter. Source: [1]

By drawing relationships between the thrust, weight, and bank angle, the final equation for the required power in horizontal flight is written as such in Equation (2.11):

$$P_{req}(V_a, \phi) = \underbrace{P_0(1 + \frac{3V_a^2}{\Omega^2 R^2})}_{\text{profile}} + \underbrace{P_i \sec\phi(\sqrt{\sec^2\phi + \frac{V_a^4}{4v_0^4}} - \frac{V_a^2}{2v_0^2})^{\frac{1}{2}}}_{\text{induced}} + \underbrace{\frac{1}{2}d_0\rho s A V_a^3}_{\text{parasite}} \qquad (2.11)$$

This equation for required power serves as the cost functional for the task formulation and

is parameterized by V_a and ϕ. Further analysis of the Equation (2.11) suggests that the required power is comprised of the sum of three individual factors - profile power, induced power, and parasite power. When the multicopter is drifting with wind in a hovering state, or when $V_a = 0$ and $\phi = 0$, the equation for required power is simplified to Equation (2.12):

$$P_{req}^{hover} = P_0 + P_i \tag{2.12}$$

It makes sense that there should be no losses due to parasite power when the multicopter is hovering and that the required power to maintain this state is only dependent on the profile and induced elements.

CHAPTER 3:
Problem Formulations

Before solving the trajectory optimization problem in the MATLAB environment, Pontryagin's Maximum Principle (PMP) is applied to the multicopter's flight dynamics and power consumption model in a typical mission profile to reveal certain conditions that must be met in order to achieve an optimal solution. These conditions for optimality help to determine the general behavior of the state and co-state variables that characterize an energy-optimal flight path.

3.1 Necessary Conditions for Optimality

The mission profile given to the multicopter is a 20-kilometer flight in the East direction in the presence of the wind environment discussed in Figure 2.3. Considering that there are no time bounds to the problem, the boundary value problem is formulated as following:

$$
\begin{cases}
\text{State:} & \mathbf{x} \in \mathbb{R}^3 \\
\text{Control:} & \mathbf{u} \in \mathbb{R}^2 \\
\text{Minimize:} & J[\mathbf{x}(\cdot), \mathbf{u}(\cdot)] = \int_{t_0}^{t_f} P_{req}(V_a, \phi) dt \\
\text{Subject To:} & \dot{x} = V_a \cdot \cos\psi + W_x \\
& \dot{y} = V_a \cdot \sin\psi + W_y \\
& \dot{\psi} = \frac{g}{V_a} \cdot \tan\phi \\
& t_0 = 0 \\
& t_f = free \\
& (x_0, y_0, \psi_0) = (0, 0, 0) \\
& (x_f, y_f, \psi_f) = (20km, 0, 0) \\
& -30^\circ < \phi < 30^\circ \\
& 4\ m/s < V_a < 20\ m/s
\end{cases}
$$

In the state space, the UAV begins its flight at the origin and is slated to terminate when it

has reached its target exactly 20 kilometers in the East direction. Its task is to choose the optimal commanded airspeed V_a and bank angle ϕ, both within the system's physical limits and along the path that will minimize energy consumption throughout the trajectory.

3.1.1 Construction of the Hamiltonian

The construction of a Hamiltonian is a method in optimal control theory for expressing the cost function in terms of the control variables and its connection to the dynamics of the system. PMP states that the Hamiltonian can be examined to help anticipate trends in the optimal solution. As seen in Equation 3.1, the Hamiltonian connects each co-state to its corresponding state dynamic equation.

$$\begin{aligned} H(\lambda, x, y, \psi) = P_0(1 + \frac{3V_a^2}{\Omega^2 R^2}) + P_i \sec\phi(\sqrt{\sec^2\phi + \frac{V_a^4}{4v_0^4}} - \frac{V_a^2}{2v_0^2})^{\frac{1}{2}} + \frac{1}{2} d_0 \rho s A V_a^3 + \\ \lambda_x(V_a \cos\psi + W_x) + \lambda_y(V_a \sin\psi + W_y) + \lambda_\psi(\frac{g}{V_a}\tan\phi) \end{aligned} \tag{3.1}$$

3.1.2 Analysis of the Co-States

Optimal Control theory also states that the Hamiltonian can be used to analyze the behavior of the co-state variables, as shown through the Adjoint Equations and Transversality Condition.

The Adjoint Equations, shown by Equation (3.2) through (3.4) provide expressions for the time rate-of-change of each of the co-states.

$$\dot{\lambda}_x(t) = -\frac{\partial H}{\partial x} = -\lambda_x \cdot J_{1,1} - \lambda_y \cdot J_{2,1} \tag{3.2}$$

$$\dot{\lambda}_y(t) = -\frac{\partial H}{\partial y} = -\lambda_x \cdot J_{1,2} - \lambda_y \cdot J_{2,2} \tag{3.3}$$

$$\dot{\lambda}_{\psi}(t) = -\frac{\partial H}{\partial \psi} = \lambda_x \cdot V_a \cdot \sin\psi - \lambda_y \cdot V_a \cdot \cos\psi \tag{3.4}$$

The variable J represents a Jacobian that is formed by taking the partial derivatives of the wind vector with respect to the three components that determine the wind strength - position in the x-direction, position in the y-direction, and time. Its elements are depicted in Equation (3.5).

$$J = \begin{bmatrix} \frac{\partial W_x}{\partial x} & \frac{\partial W_x}{\partial y} & \frac{\partial W_x}{\partial t} \\ \frac{\partial W_y}{\partial x} & \frac{\partial W_y}{\partial y} & \frac{\partial W_y}{\partial t} \end{bmatrix} \tag{3.5}$$

This generalized form of representing the Jacobian matrix allows different wind environments to be packaged into the same problem formulation. Since the wind case used for this mission is time invariant, the values of $J_{1,3}$ and $J_{2,3}$ are zero. Therefore, the specific wind case shown in Equations (2.4) and (2.5) allows the Jacobian to be simplified to the following form as shown in Equation (3.6):

$$J = \begin{bmatrix} 0 & -h_y & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.6}$$

By applying the elements of this Jacobian to Equations (3.2) through (3.4), the Adjoint Equations can be written as such:

$$\dot{\lambda}_x(t) = -\frac{\partial H}{\partial x} = 0 \tag{3.7}$$

$$\dot{\lambda}_y(t) = -\frac{\partial H}{\partial y} = -h_y \cdot \lambda_x \tag{3.8}$$

$$\dot{\lambda}_{\psi}(t) = -\frac{\partial H}{\partial \psi} = \lambda_x \cdot V_a \cdot \sin\psi - \lambda_y \cdot V_a \cdot \cos\psi \tag{3.9}$$

The application of the Adjoint Equations to this task formulation provides a comprehensive expectation of what the solution co-state trajectories should look like in order to satisfy the goals of the problem.

The Transversality condition may also reveal the general positioning of the co-state variables at the end of the simulation. It is formed by first considering the Endpoint Lagrangian, an equation that accounts for all known boundary conditions at the final time. The Endpoint Lagrangian, shown by Equation (3.10) is used in the formula to arrive at Equations (3.11) through (3.13) that provide the final values for the co-states.

$$\vec{E} = \nu_1(x_f - x^f) + \nu_2(y_f - y^f) + \nu_3(\psi_f - \psi^f) \tag{3.10}$$

$$\lambda_x(t_f) = \frac{\partial \vec{E}}{\partial x_f} = \nu_1 \tag{3.11}$$

$$\lambda_y(t_f) = \frac{\partial \vec{E}}{\partial y_f} = \nu_2 \tag{3.12}$$

$$\lambda_{\psi}(t_f) = \frac{\partial \vec{E}}{\partial \psi_f} = \nu_3 \tag{3.13}$$

The final values of all three co-state variables should be three unique constants.

3.1.3 First Order Necessary Conditions of Optimality

The first order necessary condition, or Hamiltonian Minimization Condition (HMC), might be the most important piece of information developed by applying PMP. Expressions for the optimal airspeed and bank angle that minimize energy consumption can be established

by taking the first order derivative of the Hamiltonian with respect to each control, as shown in Equations (3.14) and (3.15).

$$\frac{\partial H}{\partial V_a} = \frac{6P_0V_a}{\Omega^2R^2} + \frac{3}{2}d_0\rho sA{V_a}^2 + \frac{P_i \sec\phi\left(\frac{{V_a}^3}{2{v_0}^4\sqrt{\frac{{V_a}^4}{4{v_0}^4}+\sec^2\phi}} - \frac{V_a}{{v_0}^2}\right)}{2(\sqrt{\frac{({V_a}^4}{4{v_0}^4} + \sec^2\phi} - \frac{{V_a}^2}{2{v_0}^2})^{\frac{1}{2}}} + \lambda_x\cos\psi + \lambda_y\sin\psi - \frac{\lambda_\psi g\tan\phi}{{V_a}^2} = 0 \tag{3.14}$$

$$\frac{\partial H}{\partial \phi} = P_i \sec\phi(\sqrt{\sec^2\phi + \frac{{V_a}^4}{4{v_0}^4}} - \frac{{V_a}^2}{2{v_0}^2})^{\frac{1}{2}}\tan\phi + \frac{P_i\sec^3\phi\tan\phi}{2\sqrt{\sec^2\phi + \frac{{V_a}^4}{4{v_0}^4}}(\sqrt{\sec^2\phi + \frac{{V_a}^4}{4{v_0}^4}} - \frac{{V_a}^2}{2{v_0}^2})^{\frac{1}{2}}} + \lambda_\psi(\frac{g}{V_a}\sec^2\phi) = 0 \tag{3.15}$$

Both of the controls are coupled in these equations, meaning that they depend on each other throughout the trajectory. For a software package such as DIDO that is able to iterate through possible values of the controls during the solving process, defining an explicit expression for each of the controls is not necessary. MATLAB's internal BVP solver, however, does not allow as much flexibility with defining these conditions in the problem space. The relationship between the optimal airspeed and the power required to fly, as well as the effect of wind direction on the commanded airspeed, are analyzed in Equation (3.14) to provide an accurate expression for V_a. By assuming that the airspeed will always be much greater than the induced velocity during hover, v_0, a first-order Taylor Series approximation is applied to the third term in Equation (3.14) to eliminate the presence of bank angle in that term. Additionally, it is assumed that the power required for horizontal flight does not depend on the rotor inclination angle. These steps help to simplify Equation (3.14) and create an expression for optimal airspeed that is dependent on the power required for level flight and the effects of wind,

$$\frac{\partial H}{\partial V_a} = \frac{6P_0V_a}{\Omega^2R^2} + \frac{3}{2}d_0\rho sA{V_a}^2 + \frac{P_{in}v_0}{V_a} + \Lambda = 0 \tag{3.16}$$

where Λ is equal to the term $\lambda_x \cos\psi + \lambda_y \sin\psi$. It can therefore be observed that the airspeed to minimize energy consumption is dependent on the power required for forward flight (in zero-wind conditions) and the magnitude and direction of wind, respectively. When the multicopter flies in conditions where there is no wind present or it is constant, the term Λ becomes equal to 0 and the energy-optimal airspeed is characterized solely by the modified terms for required power. The relationship between the optimal airspeed and wind effects, $V_a^*(\Lambda)$, is analyzed over a wide range to gain a better understanding of how the multicopter might alter its commanded airspeed in varying wind conditions. It was discovered that a positive Λ depicted a headwind and a higher commanded airspeed while a negative Λ illustrated a tailwind, calling for a lower airspeed. This relationship confirmed t hat the process of simplifying the expression for optimal airspeed in the HMC conditions was valid. The fact that optimal bank angle ϕ was coincidentally removed from the equation did not decrease the fidelity of the control. The polynomial curve was fitted and implemented in the BVP-solving algorithm. It can also be observed in Equation (3.16) that the solution for optimal airspeed is of third order, which is an accurate characterization of how the airspeed of the vehicle should react in a varying wind environment.

Now that the expression for optimal airspeed has been made explicit, Equation (3.15) can be modified accordingly so that MATLAB's BVP solver can read it without any numerical issues. The current expression for the optimal bank angle is too complex (the order of the polynomial is too high to provide a sensible array of values) and must be adjusted without sacrificing accuracy. The same assumption of the required power's independence from rotor inclination angle was applied to the pitch angle of the multicopter. The rotary power consumption theory laid out in [1] was also used to include thrust as a variable in the induced power portion of the cost function, allowing Equation (2.11) to be rewritten as such:

$$P_{req}(V_a^*, \phi, T) = \underbrace{P_0\left(1 + \frac{3V_a^{*2}}{\Omega^2 R^2}\right)}_{\text{profile}} + \underbrace{\left(1 + \frac{1}{\cos\phi}\right) T v_{i0}}_{\text{induced}} + \underbrace{\frac{1}{2} d_0 \rho s A V_a^{*3}}_{\text{parasite}} \tag{3.17}$$

In Equation (3.17), v_{i0} is the mean induced velocity of the multicopter that also accounts for the rotor thrust, T. By referencing the free body diagram of the multicopter in Figure

2.4, recall that the rotor thrust is equal to the weight of the vehicle divided by the cosine component of the bank angle, $T = \frac{mg}{\cos\phi}$. However, with the assumption that thrust is not sensitive to rotor inclination angle, the total thrust in this equation can be calculated from the optimal speed to fly along the longitudinal axis. Essentially, this allows thrust to be written as a function of the optimal airspeed that has already been calculated, $T(V_a^*)$ and further simplify the required power in Equation (3.17). This equation is used to derive the corresponding Hamiltonian, resulting in the following first-order necessary condition for the optimal bank angle:

$$\frac{\partial H}{\partial \phi} = \frac{g\lambda_\psi(\tan^2\phi + 1)}{{V_a}^*} + \frac{v_{i0}mg\sin\phi}{\cos^2\phi} = 0 \tag{3.18}$$

Resolving equation (3.18) with respect to ϕ shows that the bank angle depends mainly on the commanded airspeed, mean induced velocity, and heading co-state.

$$\phi^* = \sin^{-1}\left(\frac{\lambda_\psi}{V_a \cdot v_{i0} \cdot m}\right) \tag{3.19}$$

A scaling term h_{cont} will be added to the equation at the "continuation" stage to improve the convergence speed of the BVP algorithm. The details of the "continuation approach" and its numerical implementation are discussed in section 4.2.

$$\phi^* = \sin^{-1}\left(\frac{h_{cont} \cdot \lambda_\psi}{V_a \cdot v_{i0} \cdot m}\right) \tag{3.20}$$

3.1.4 Hamiltonian Value Condition

The Hamiltonian Value Condition (HVC) provides that the Hamiltonian at the final time of the simulation is equal to the partial derivative of the Endpoint Lagrangian with respect to any time-constricted terms, as shown in Equation (3.21).

$$H(t_f) = \frac{-\partial \vec{E}}{\partial t_f} = 0 \tag{3.21}$$

Recall from Equation (3.10) that the Endpoint Lagrangian depends on the final state conditions and is absent of a final time parameter. Since time minimization is not a parameter in the trajectory optimization problem, it is expected that the Hamiltonian at the final time, $H(t_f)$ should be equal to zero.

3.1.5 Hamiltonian Evolution Condition

The Hamiltonian Evolution Condition (HEC) is constructed by taking the partial derivative of the *minimized* Hamiltonian, H, with respect to time. Equation (3.22) shows that when the Hamiltonian is truly minimized, its partial derivative should be equal to that of the standard Hamiltonian.

$$\frac{\partial H}{\partial t} = \frac{\mathrm{dH}}{\mathrm{dt}} = 0 \tag{3.22}$$

This procedure proves that the minimized Hamiltonian, H, is constant as a function of time. These two pieces of information reported in Sections 3.1.4 and 3.1.5 provide the expected trajectory of the Hamiltonian during the simulation. Since the minimized Hamiltonian should stay constant with time and terminate with a value of zero, its trajectory should display as a straight line along the x-axis. By applying Pontryagin's Maximum Principle to the boundary value problem before attempting to reach a solution, the process of exposing red flags and sub-optimal features becomes more apparent. The necessary conditions outlined by PMP serve as a compass when implementing the boundary value problem into a software program and also help to authenticate the optimality of the solution.

3.2 Problem Scaling

Once given the delivery profile, the UAV must process the optimal trajectory algorithm onboard before flying its course, and must continuously do so in order to search for routes that offer a higher amount of potential energy savings. Computation time can be significantly decreased by scaling the problem space down to a much more manageable size for MATLAB or any software program to search through. The process of applying scaling to an optimal control problem should not skew the trajectories of the state dynamics equations, but rather map them to a smaller space to shorten the time to reach the solution. The solution should

then be "deconstructed" by applying the same scaling factors that were used to initialize the problem formulation. For this problem formulation, a canonical-based approach is taken in scaling the multicopter's system dynamics. Arbitrary factors for velocity and acceleration are chosen and used as a foundation for scaling the dynamical equations.

$$V_{scale} = \frac{DU}{TU} = arbitrary \tag{3.23}$$

$$A_{scale} = \frac{DU}{TU^2} = arbitrary \tag{3.24}$$

The terms DU and TU are used to represent "distance units" and "time units", respectively, in order to distinguish between variables that are calculated in the engineering and canonical form. These scaling factors are then applied to the velocity triad in order to arrive at values for mapping the time and distance variables.

$$T_{scale} = \frac{V_{scale}}{A_{scale}} = TU \tag{3.25}$$

$$D_{scale} = V_{scale} T_{scale} = DU \tag{3.26}$$

Since angles are depicted in radians for this problem formulation, it does not make sense to subject ψ and ϕ to scaling because the domain for these variables is already small. Feeding scaled values to trigonometric operations would also further skew the data. Thus, all relevant variables are able to be compressed by the four scaling factors shown in Equations (3.23) - (3.26) to create a new set of fully-scaled state and co-state dynamics equations, displayed by Equations (3.27) to (3.32).

$$\dot{\tilde{x}} = \tilde{V}_a \cos\psi + \tilde{W}_x \tag{3.27}$$

$$\dot{\tilde{y}} = \tilde{V}_a \sin\psi + \tilde{W}_y \tag{3.28}$$

$$\dot{\tilde{\psi}} = \frac{\tilde{g}}{\tilde{V}_a}\tan\phi \tag{3.29}$$

$$\dot{\tilde{\lambda}}_x(t) = -\frac{\partial \tilde{H}}{\partial \tilde{x}} = T_{scale}(-\lambda_x \frac{\partial W_x}{\partial x} - \lambda_y \frac{\partial W_y}{\partial x}) \tag{3.30}$$

$$\dot{\tilde{\lambda}}_y(t) = -\frac{\partial \tilde{H}}{\partial \tilde{y}} = T_{scale}(-\lambda_x \frac{\partial W_x}{\partial y} - \lambda_y \frac{\partial W_y}{\partial y}) \tag{3.31}$$

$$\dot{\tilde{\lambda}}_\psi(t) = -\frac{\partial \tilde{H}}{\partial \psi} = \lambda_x \tilde{V}_a \sin\psi - \lambda_y \tilde{V}_a \cos\psi \tag{3.32}$$

Any variable that is marked with a tilde (~) symbol signifies that it has been scaled properly. The full derivation of the scaled dynamics equations can be referred to in Appendix D.

CHAPTER 4:
Practical Results

After the problem formulation was fully analyzed and the conditions for optimality in the solution were defined, the mission was simulated using the DIDO software and BVP-solving environment in MATLAB. The solutions produced from both methods are verified by the requirements laid out in Section 3.1 and are also compared with each other in order to strengthen their validity.

4.1 Mechanics of the Software Algorithms

The DIDO software package is used to solve the BVP by defining a valid search space for the state and control variables. DIDO has the ability to examine time-discrete solutions within the search space that both satisfy the boundary conditions and minimize the cost function. The BVP-solving functions in MATLAB apply a method known as collocation to iterate through potential solutions. Instead of building a search space for the state and control variables, the problem formulation is fed into the algorithm by defining a time horizon for the simulation that is divided into a certain number of nodes. The multicopter's system dynamics equations, mission-specific boundary conditions, and an initial guess for the state trajectories must also be provided to the solver. The algorithm then processes this information as a set of ordinary differential equations (ODEs) and solves them simultaneously through each discrete step in the time horizon. Since there are no time bounds in the mission and the final time of the simulation is unknown, this value is also solved for by first creating an initial guess, then deconstructing the guess with solution-extracted parameters. In addition to the BVP solving technique, a method known as continuation is used to increase the effectiveness of the BVP solver and verify the expected behavior of the solution.

4.2 Continuation Algorithm

Unlike DIDO, the BVP solver approaches the problem with a predefined search space. The initial guess, therefore, is prone to inaccuracy and can inhibit the algorithm from converging on a solution. A method known as continuation helps to overcome the difficulty of producing

a feasible guess at the solution and increases the overall effectiveness of the route-finding task [8]. The process involves running the algorithm a single time and saving the solution set to be used as the initial guess for each subsequent run. For the multicopter's flight task, the first run of the simulation is conducted in a non-wind environment, and the solution set from that first run is applied to the initial guess of the second. This process is repeated until the solver runs in an environment where the wind matches the full magnitude of that which is provided in the mission profile. This incremental adjustment of the wind magnitude is also applied in the optimal control for the bank angle, Equation (3.20). As the strength of wind increases, the banking input to the system should also increase proportionally. Employing continuation in this manner ensures a positive trend of initial guesses and provides a proper sense of direction for the algorithm in wind environments that range from mild and predictable to turbulent and erratic. The technique also leads to a faster computation time in that it removes unnecessary guesses at the solution that may be far off.

4.3 Overview of the Optimal Control Solution

This section provides an overview of the key properties of the optimal trajectory obtained by both DIDO and the BVP solver in MATLAB. The following figures connect the set of assumptions made towards the task formulation with the physical interpretation of the optimal trajectory and two control laws of the airspeed and bank angle. The discussion highlights the functions performed by the autopilot when it is given the trajectory and the airspeed as two fundamental references to follow. It should not be confusing that the optimal bank angle is omitted as the knowledge of trajectory and the speed uniquely identifies the "missing" bank control.

The scenario describes an optimal flight between the boundary conditions which are distant at 100 m. The wind is time invariant and follows the definition given in Equation (2.4) and (2.5). The resulting trajectory is presented in Figure 4.1. The scale of axes is chosen equal to highlight the minimal curvature and symmetric shape of the resulting path. Despite the insignificant magnitude of wind, the trajectory goes through the area of tailwind with higher intensity thus demonstrating the energy harvesting effect.

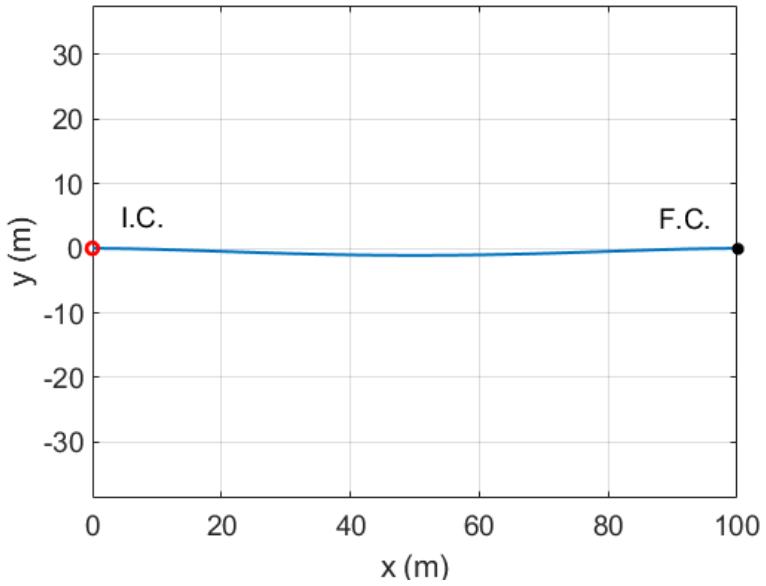

Figure 4.1. Multicopter Optimal Trajectory for 100 Meter Flight.

It can also be observed in Figure 4.1 that the heading of the multicopter points primarily east throughout its trajectory; an exclusive tailwind is the dominating vectoral wind component during flight. While the vehicle does turn south to take advantage of the higher magnitude of tailwind in that region, it does not fly through a crosswind and the optimal trajectory is not much longer in distance than a path that follows the shortest Great Circle route. Figure 4.2 illustrates the nature of the optimal airspeed and the bank angle.

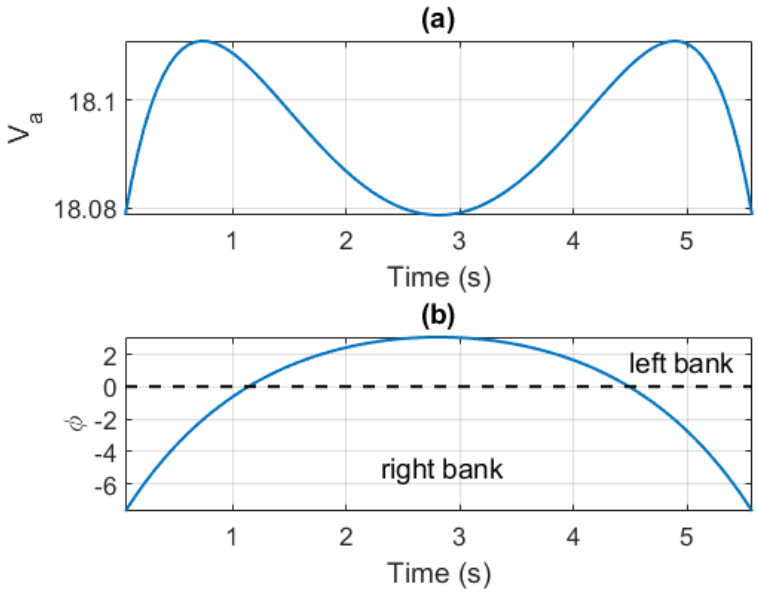

Figure 4.2. Control Variable Trajectories for 100 Meter Flight.

In Figure 4.2, both the airspeed and bank angle during the 100 *m* flight behave as expected and adhere to aerodynamic laws. While the change in airspeed is minimal, the shape of the time history represents the act of harvesting wind energy by decreasing the commanded airspeed in stronger tailwind regions. When the multicopter does encounter minimal crosswinds when turning in and out of its most southerly flight path, the autopilot increases the airspeed accordingly. The UAV also flies faster through crosswind regions in order to arrive at the maximum tailwind section in a shorter amount of time. Similarly, the time history of the bank angle commanded by the autopilot follows a practical trend. For this short flight, the vehicle initiates its trajectory by banking to the right, which is shown by the negative value. As the multicopter begins to enter the complete tailwind region, its right bank decreases until it must turn the other direction to reach the final coordinates. Although a left banking maneuver properly sends the UAV back north to the target, another right bank is required to satisfy the requirement of 0° for the final heading angle. The switching of the bank angle sign from negative to positive and then positive back to negative delineates two coordinated turn maneuvers during the trajectory.

4.4 Optimal Solution Produced by DIDO

The DIDO software solved the BVP with the conditions set forth in Section 3.1 and provided an array of the solution states, co-states, and controls throughout time. In Figure 4.3, the axes are not scaled because the deflection in the North-South direction is small compared to the distance traveled. Representing the trajectory in this manner allows clear visualization of the turning points along the flight path.

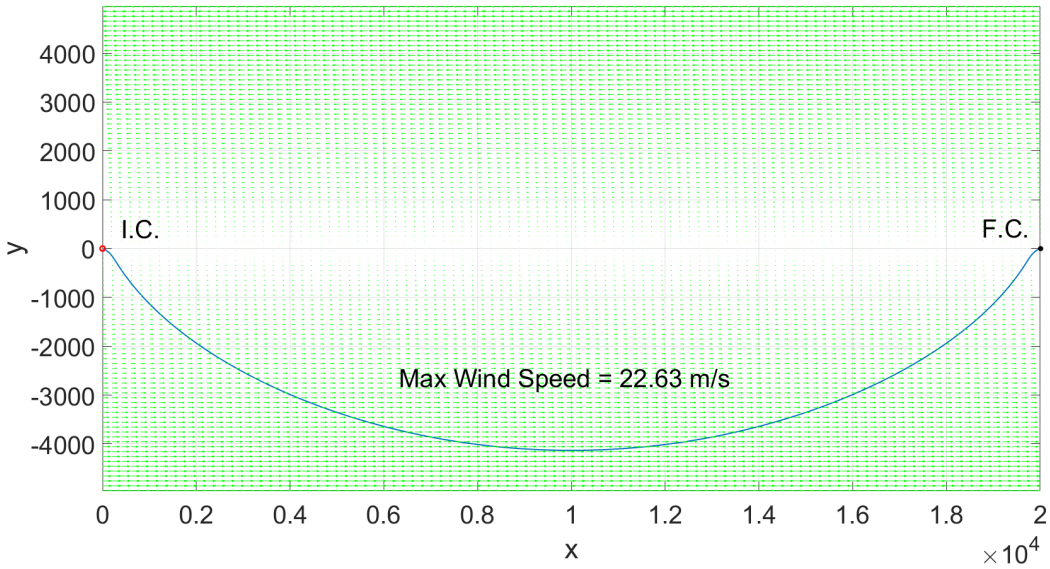

Figure 4.3. Multicopter Optimal Trajectory Produced by DIDO Software Package.

Figure 4.3 displays the multicopter's trajectory that minimizes energy consumption during its flight from the origin to 20 kilometers due East. The green arrows depict the strength and direction of the wind field in the scenario. The UAV immediately flies "south" to take advantage of energy savings in the tailwind region, as the wind in the easterly direction increases in magnitude the more south the UAV flies. It is expected that the flight path is symmetrical in nature, given that the wind field is mirrored about the x-axis. It should also be noted that the onboard autopilot points the nose of the multicopter in the direction of the airspeed vector, thus eliminating the necessity to crab in crosswind. This wind gradient offers multiple levels of verification to the solution by analyzing the shape of the trajectory

in conjunction with that of the controls. Figure 4.4 shows the inputs to the airspeed and bank angle from the DIDO solution.

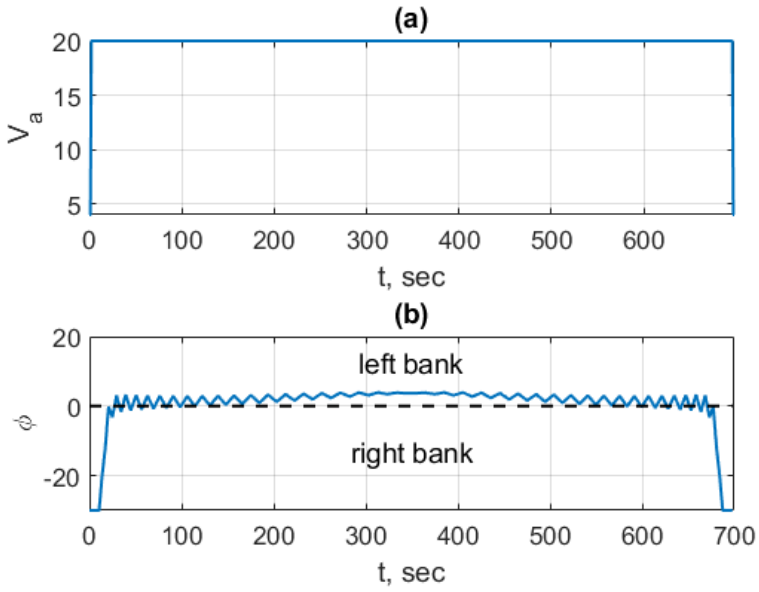

Figure 4.4. Control Variable Trajectories from DIDO Solution. (a) Airspeed, m/s (b) Bank Angle, deg.

In Figure 4.4, both the airspeed and bank angle display trends that match the multicopter's trajectory. The commanded airspeed does not surpass the system's 20 m/s ceiling and the bank angle never exceeds the physical design limit of 30 degrees on each side. The autopilot can be seen to increase the airspeed once entering crosswind regions and holds it until required to turn back to the final position. This shows that even in sections where the speed of the tailwind is around 10 m/s, DIDO still found it necessary to maintain airspeed in order to minimize fuel consumption. Additionally, the bank angle shown in the figure is the exclusive lateral roll commanded by the autopilot. The aerodynamic concept that describes the physical relationship between bank angle and airspeed also holds in Figure 4.4. When an aircraft flies with a relatively slower airspeed, a greater bank angle is required to conduct the same magnitude turning maneuver than if it was flying faster. This banking in response to airspeed follows the same trend displayed in Figure 4.2. At the start of the route, the multicopter recognizes the advantage of flying south and immediately turns that

way, initiating a large bank angle to the right. Once the vehicle is heading in the proper direction, the magnitude of the bank input starts to relax. While the behavior of the flight path and control trajectories make visual sense, it is difficult to determine true optimality of the solution from these two figures. The goal of the task is to minimize the cost function over time while meeting the necessary conditions discussed in Section 3.1, so a further analysis of the results is required in the solution-validation process. A plot of the so-states is shown in Figure 4.5 to help support this verification procedure.

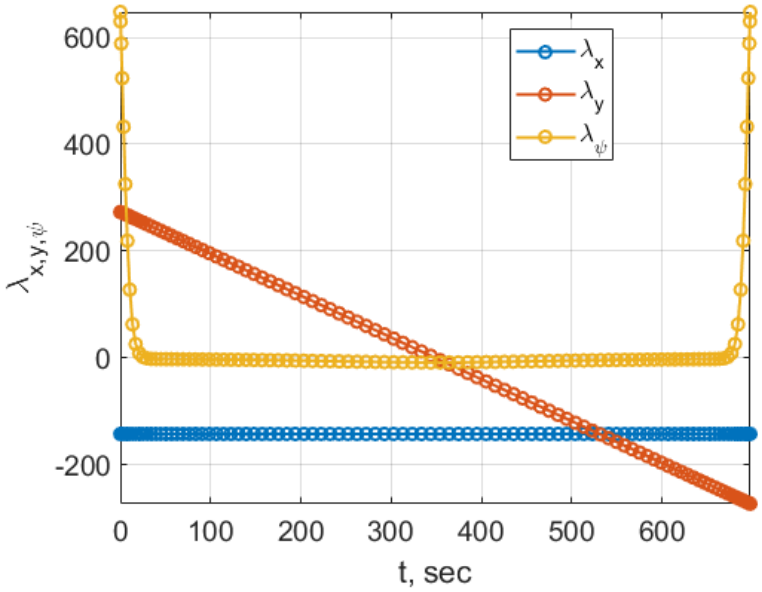

Figure 4.5. Co-state Trajectories from DIDO Solution.

Figure 4.5 illustrates the solved co-state trajectories from DIDO and strengthens the validity of the algorithm by satisfying the conditions outlined in Section 3.1.The slope of λ_x is zero, which complies to the information provided in Equation (3.1.2). Because this characteristic of λ_x is met, Equation (3.1.2) states that the slope of λ_y should be some negative constant, meaning that its trajectory should display a downward, linear trend. As Equation (3.1.2) suggests, the co-state λ_ψ is a function of both λ_x and λ_y and should therefore be second order in nature. In addition to meeting the conditions of the Adjoint Equations, the co-state trajectories satisfy the Transversality Conditions in that all three of their final values are equal to unique constants. In order to further analyze the efficiency of the DIDO solution,

Figure 4.6 compares the optimal solution with a second route that has the potential to offer increased energy savings.

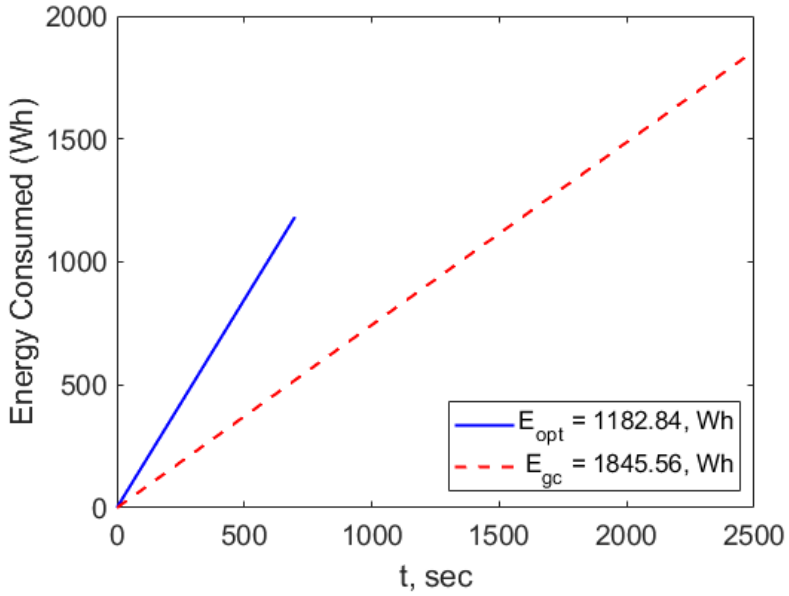

Figure 4.6. Comparison of Optimal and Sub-Optimal Energy Consumption from DIDO Solution.

Figure 4.6 shows the UAV's cumulative energy consumption over time for two different scenarios. The first, represented by the solid curve, is the energy corresponding to the optimal trajectory shown in Figure 4.3. The second case, illustrated by the dashed line, describes a vehicle that ignores the presence of wind and takes the shortest Great Circle route to the given target, also flying with an airspeed designed to minimize power loss in no-wind conditions. For this "wind-ignorant" route, the bank angle is set to zero (since there is no need to turn when flying in a straight line without wind) and the airspeed is set to a constant value that minimizes the required power. Not only does the optimal route burn less energy than the wind-ignorant route, but it also requires a shorter travel time. The benefits of increased ground speed obtained in the tailwind region explain the significantly shorter time-of-flight provided by the optimal trajectory. While the multicopter does consume energy at a faster rate throughout its flight, the airspeed of the vehicle is supplemented by a substantial amount of wind speed in the same, vectoral direction. By analyzing the

relationship between time, distance traveled, and ground speed in Equation (4.1), it is clear that a substantial increase in speed over ground leads to a shorter time-of-flight.

$$Time\ of\ Flight = \frac{\sqrt{x^2 + y^2}}{V_g} \tag{4.1}$$

The strength of the wind as implemented in the task formulation by Equations (2.4) and (2.5) is controlled by the size of the constant parameter. A specific value of this parameter is applied to the algorithm in order to ensure that the speed of the wind does not exceed 30 m/s, which is somewhat unrealistic for multicopter flight conditions but allows convenient visual analysis of the influence of wind. This environment is used to clearly demonstrate that the consideration of wind leads to consistent energy savings throughout time.

4.5 Optimal Solution Produced by the BVP Solver Method in MATLAB

MATLAB's BVP solver is just as efficient in producing a solution for the same optimal control task of flying 20 km in the presence of wind. The opportunity to compare the results generated via the BVP solving method with those from DIDO add an extra layer of verification to both techniques. Although the mechanisms that each of the algorithms apply are quite different, the BVP solver method should produce a similar solution set in order to achieve optimality. Figure 4.7 shows the optimal trajectory in the 2-D frame that is produced from the problem formulation in Section 3.1. The axes are once again not set to scale in order to allow easy interpretation of the trajectory.

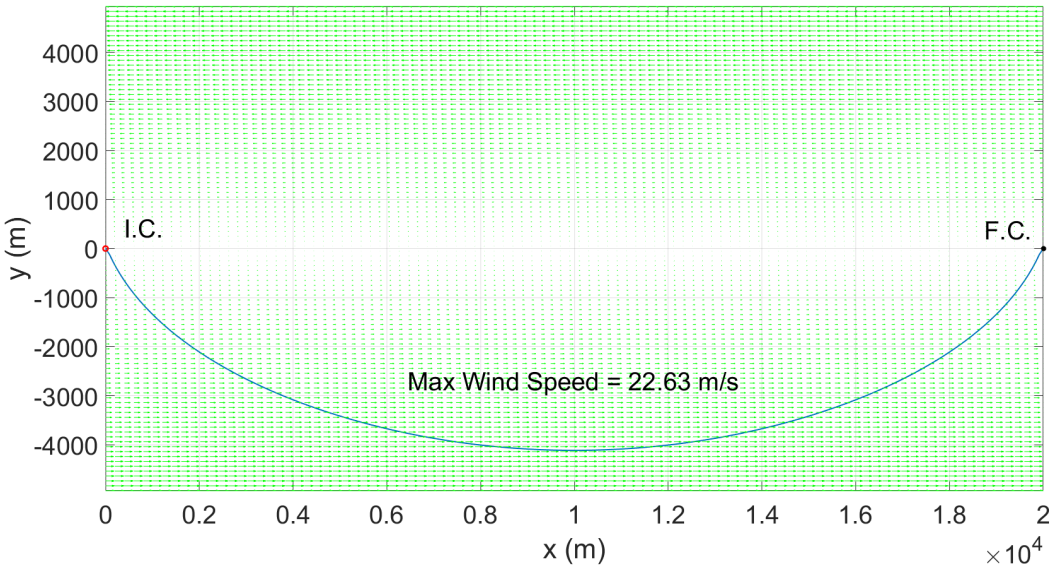

Figure 4.7. Multicopter Optimal Trajectory produced by MATLAB's BVP Solver.

In Figure 4.7, the multicopter also flies south to gain access to stronger tailwinds and follows a symmetrical trajectory, thus adhering to the wind gradient. While the UAV's path exhibits a slightly larger amount of deflection to the south than that from the DIDO solution, the two routes are still similar in nature. The difference between these two trajectories is explained by the BVP solver's strict protocol for formulating the control trajectories. Recall from Section 4.1 that DIDO has the power to iterate through possible control trajectories that minimize the cost function without having to provide an initial structure or set of equations for these variables. Adversely, the BVP solver requires an explicit representation of the control variables in the algorithm that is derived from the minimization condition of PMP. Therefore, a difference in the controls among the DIDO and BVP solver methods is inevitable. The appropriate procedure for comparing the solution sets of the two methods should be to ensure that the state, co-state, and control trajectories follow the same expected trends and satisfy the necessary conditions for optimality. Figure 4.8 shows the airspeed and bank angle time histories during flight.

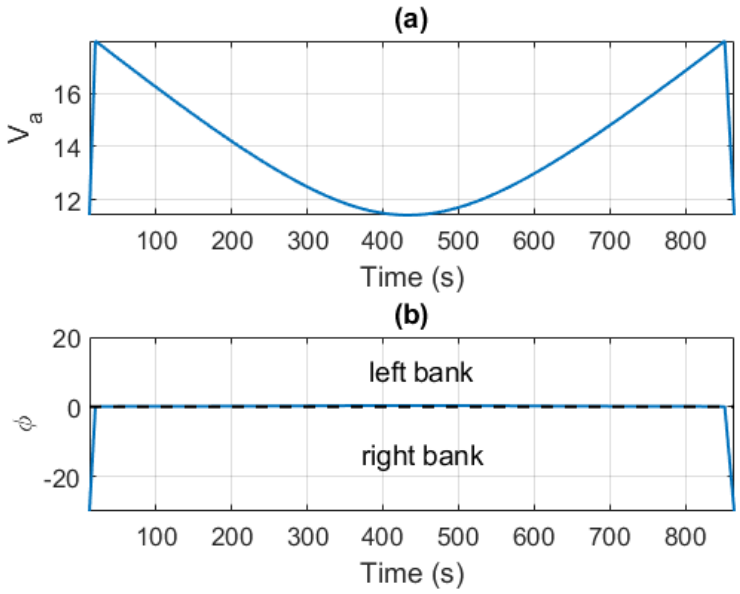

Figure 4.8. Control Variable Trajectories from BVP Solver. (a) Airspeed, m/s (b) Bank Angle, deg.

In Figure 4.8, both the airspeed and bank angle stay within their design limits and follow the same flow as that which is displayed in Figure 4.4. The UAV commands a higher airspeed in the beginning of the route and slowly decreases it as the vehicle moves into stronger tailwinds. At the very beginning and end of the route, a larger bank angle to the right (negative) is required to match the vehicle's low airspeed and turn in the presence of crosswinds. For a majority of the trajectory the bank angle remains near to zero, except for the middle section of the route where the UAV must turn in and out of the straight tailwind region. This specific behavior of the control trajectories shows that the algorithm immediately recognizes where to fly in order to reduce its immediate energy consumption, and wastes no time orienting and flying to the appropriate area. In order to further verify that the BVP solver method using continuation offers a consistent, energy-optimal solution, more analysis of the result and its satisfaction of Pontryagin's Maximum Principle is conducted. An examination of the co-states in Figure 4.9 is conducted to reference PMP.

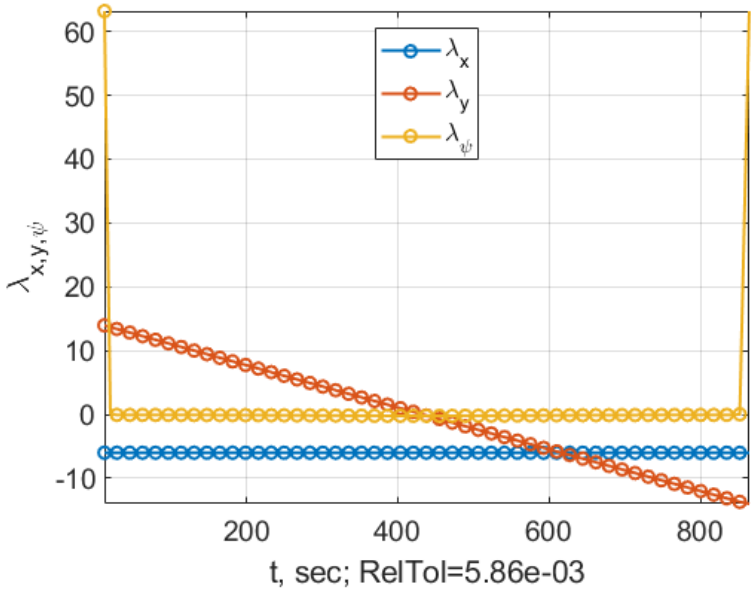

Figure 4.9. Co-state Trajectories from BVP Solver.

The co-state solution displayed in Figure 4.9 also matches the behavior in Figure 4.5 and meets both the standards of the Adjoint Equations and Transversality Condition. The sharper nature of the λ_ψ time history concurs with the faster response of the control input behavior. The same comparison of energy consumption of the optimal trajectory and the shortest Great Circle route is produced in Figure 4.10.

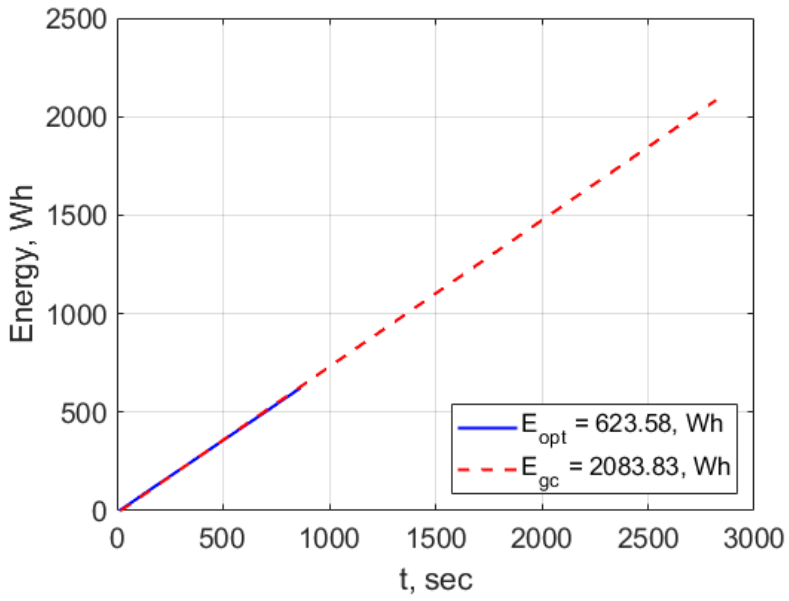

Figure 4.10. Comparison of Optimal and Sub-Optimal Energy Consumption from BVP Solver.

Figure 4.10 shows that the required energy for the solved trajectory is much lower than that of a Great Circle route. The lower values of commanded airspeed throughout the flight also help to explain the fact that the vehicle consumes energy at a slower rate than it does in the simulation run by DIDO. Additionally, it can be seen that the energy consumption required by the BVP solver is much less than that of DIDO. This lower energy requirement is attributed to the fact that the airspeed is throttled back in the tailwind region to take advantage of the increased ground speed. The same strength in wind gradient is applied to this solving method and contributes to a significant decrease in time-of-flight.

Providing energy-optimal solution sets that satisfy the necessary conditions established by PMP with two unique methods opens the door for trade studies in regards to onboard compatibility. While it took DIDO an average of 30-60 seconds to run the algorithm on an Intel core i7 processing unit, the BVP solver required approximately 2-4 seconds to run the entire continuation scheme on the same system. DIDO also requires a predefined search space for the state and control trajectories which must be adjusted manually as boundary conditions and wind environments will vary with each delivery package. In

MATLAB's BVP-solving environment, the boundary conditions are read in conjunction with the system dynamics equations to define the problem space. This protocol supplemented by the continuation method provides the capability to solve different BVPs without depending on manual, user adjustments. In consideration of the given trade studies, it is much more practical to run the BVP solver onboard the multicopter, as the algorithm is expected to frequently update the system with new wind conditions and potential route adjustments.

4.6 Numerical Analysis of Energy Efficiency

With the BVP solver as the algorithm of choice to be implemented onboard the system, it is important to further verify its ability to produce truly optimal results. One such way to analyze the energy efficiency of the continuation method is by running the same task formulation with a variance of control inputs to search for an altered trajectory that might provide additional energy savings. As the applied model describes a wind strength that increases in magnitude with displacement away from the x-axis, a neighboring trajectory that turns farther south to fly in a stronger tailwind region has the potential to minimize energy consumption to a higher degree. Similarly, a route that flies closer to the x-axis may save more energy by requiring smaller turning maneuvers. These two neighboring routes are compared with the solution generated in Figure 4.10 to allow visual analysis in Figure 4.11.

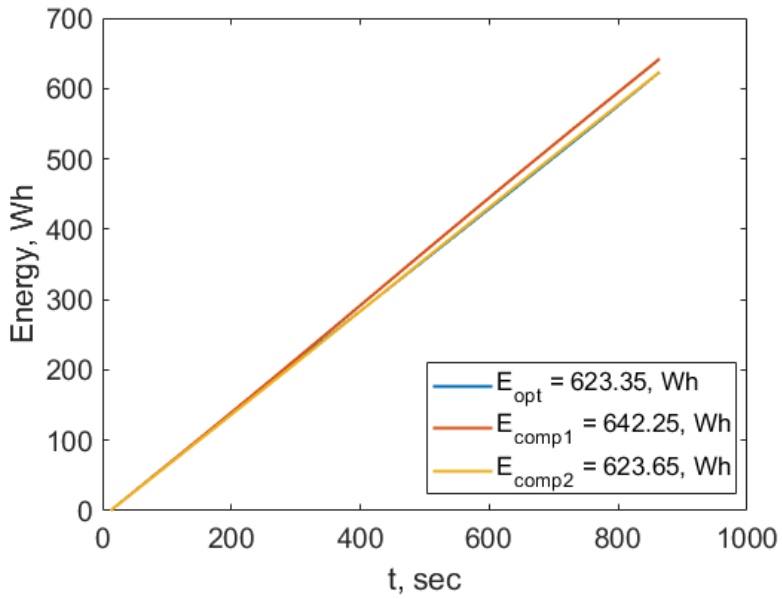

Figure 4.11. Comparison of Optimal Energy Consumption from BVP Solver with Alternate Solutions.

In Figure 4.11, it is evident that the trajectory of the optimal solution requires less energy usage than both of the alternate routes. The curve depicted as E_{comp1} represents a trajectory that stays relatively closer to the x-axis with smaller control inputs. Conversely, the route labeled as E_{comp2} employs larger control inputs and flies farther south than the solution trajectory. A more detailed numerical analysis of comparable trajectories and their respective energy requirements is shown in Figure 4.12.

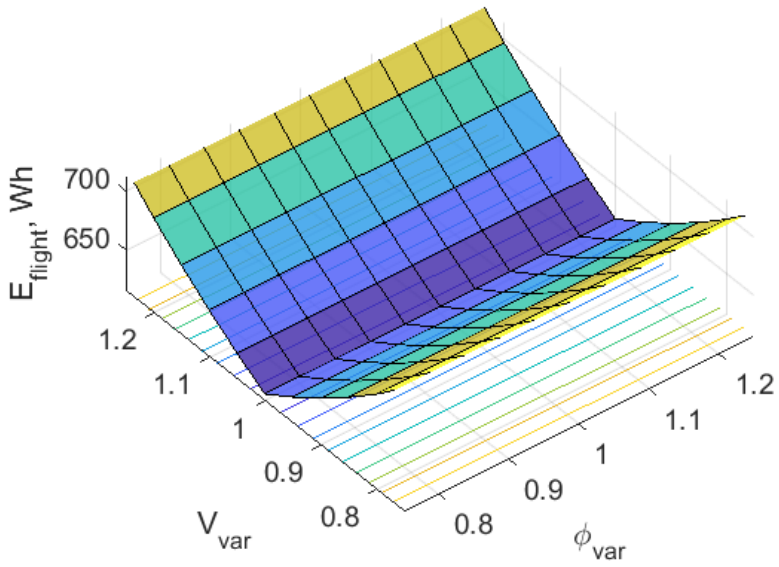

Figure 4.12. Variance of Optimal Energy Consumption from BVP Solver.

Figure 4.12 displays the resulting energy consumption of running the algorithm with all possible combinations of control inputs within a 25% variance from the optimal airspeed and bank angle solutions. Any variance away from the solution provided by the BVP solver leads to an instant increase in energy expended throughout the flight, proving that the trajectory is indeed the most efficient option. While the penalty for altering the airspeed from the solution is transparent, the effects of varying the bank angle is difficult to observe. This is due to the fact that the commanded bank angle is near to zero during a majority of the trajectory; thus, a variance in such a small number leads to smaller penalties in energy consumption.

4.7 Introduction of Time-Variance to the Wind Environment

The effectiveness of the BVP solver algorithm is further tested by employing it in a time-varying wind environment. Nearly the same problem formulation is given to the software with the addition of time-dependency in the wind model,

$$W_x(x, y, t) = -(at + b)y \tag{4.2}$$

$$W_y(x, y, t) = 0 \tag{4.3}$$

where a and b are both user-defined coefficients to modify the strength of wind with respect to time and displacement from the x-axis. Figure 4.13 shows the resulting BVP solver trajectory to the problem formulation discussed in Section 3.1.

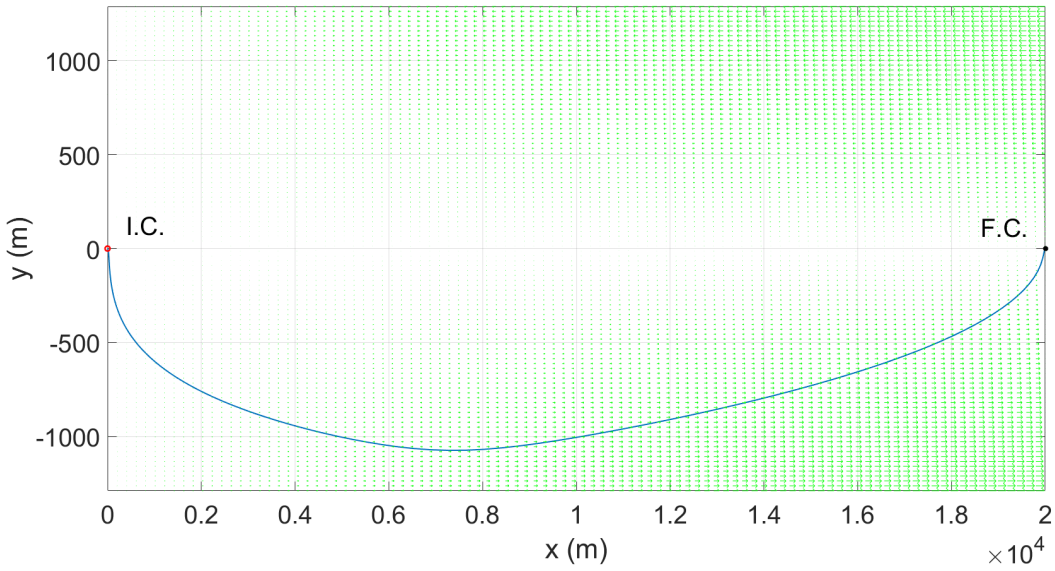

Figure 4.13. Multicopter Optimal Trajectory Produced by MATLAB's BVP Solver in Time-Varying Wind.

The flight path in Figure 4.13 shows that the algorithm also possesses the capability to process a time-varying wind in the delivery package. The green arrows represent the magnitude of the wind speed to change locally; that is, as time increases in the simulation the larger effect of wind is applied to the equation based on the multicopter's exact location. The vehicle turns south to fly in the tailwind region as expected, but the trajectory is asymmetrical, unlike that which is displayed in Figure 4.7. The system realizes that the strength of wind increases with flight time, and therefore makes a greater effort to enter the tailwind region in the beginning of the route in order to take advantage of this specific wind pattern. The

time histories of the control inputs are also shown in Figure 4.14.

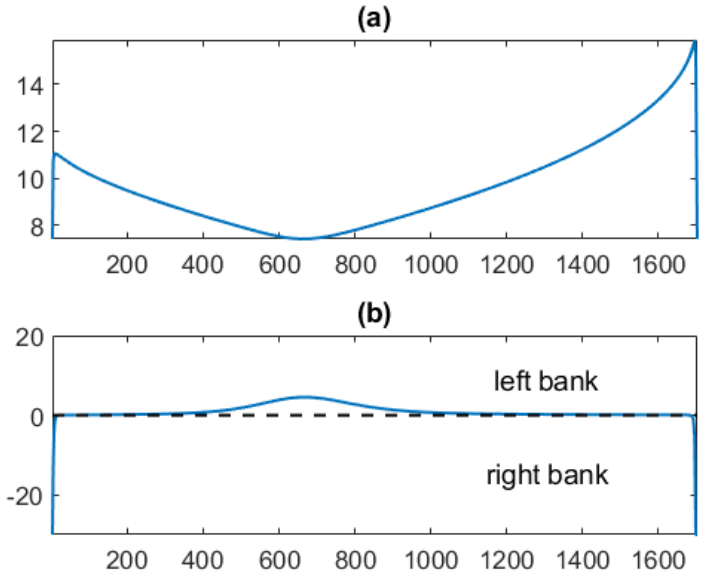

Figure 4.14. Control Variable Trajectories from BVP Solver in Time-Varying Wind. (a) Airspeed, m/s (b) Bank Angle, deg.

The commanded airspeed and bank angle in the time-varying wind environment correspond to the turning maneuvers in the flight path. The airspeed decreases and reaches its low point at an earlier point in the trajectory, due to the multicopter experiencing stronger tailwinds with time. The same energy-oriented analysis of the time-varying wind solution is employed to prove true optimality, shown in Figure 4.15.

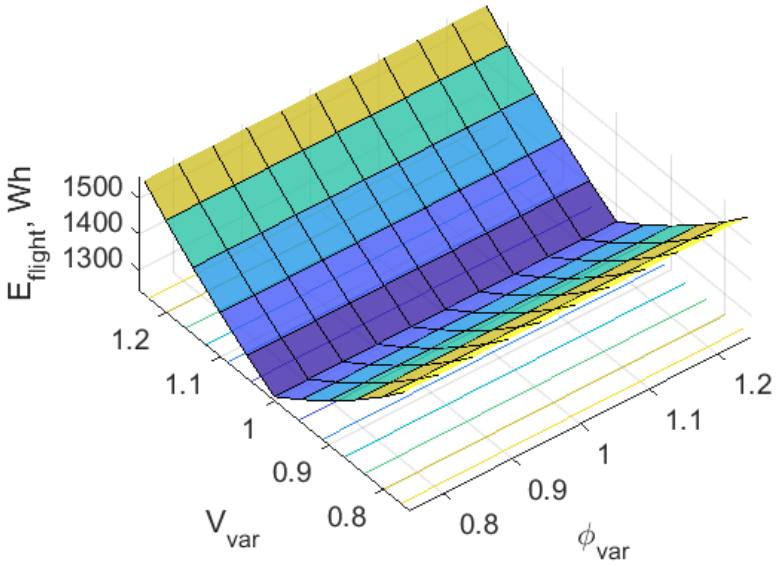

Figure 4.15. Variance of Optimal Energy Consumption from BVP Solver in Time-Varying Wind.

Although increasing the airspeed in the beginning of the trajectory to enter the tailwind region earlier presents a possible decrease of overall energy usage, doing so has the opposite effect. While a simple, time-independent wind field creates transparency in the solution-verification process, an environment that changes throughout time provides a more realistic scenario for the system. The ability of the BVP solver to produce an exact trajectory that minimizes energy expenditure in the presence of time-varying wind speaks to the accuracy of the algorithm's process in determining the proper control inputs.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5:
Conclusion and Recommendations

5.1 Conclusions

This study proves that the application of dynamic optimization theory to the objective of minimizing a multicopter UAV's energy expenditure during flight significantly changes the approach to tactical resupply for the USMC. By integrating an algorithm with the capability of harvesting energy from wind onboard the system, the UAV optimizes its use of fuel throughout the entire flight.

By packaging critical information from the delivery profile and an accurate model of the multicopter system into a logical and readable structure, the algorithm solves for the exact optimal trajectory for flight. The solver ultimately interprets the mission as a boundary value problem with the goal of minimizing the cost function (required energy), throughout the flight. While multicopter UAVs share some similarities with those of the fixed-wing type, their flight dynamics and power consumption contain many technical differences. The energy expenditure model for a generic multicopter UAV is dependent on the instantaneous airspeed and the total pitch angle of the system. This power model, along with a set of dynamic equations to describe the state of the system, are used in conjunction to formulate the boundary value problem read by the onboard algorithm. Two different methods of solving this problem in the software environment, DIDO and MATLAB's BVP solving functions, were analyzed and their results compared. While both techniques were successful in developing optimal solutions, the BVP solver method offers a much faster computation time and is more compatible with the autonomous system. An algorithm that applies continuation to the BVP solver produces a consistent optimal solution in different environments and can be confidently implemented onboard a multicopter UAV in that it satisfies the necessary conditions set forth in dynamic optimization theory.

5.2 Future Work

While the algorithm developed in this study that utilizes both collocation and continuation is fully verified in the discussed problem space, its overall robustness can be improved in order to harden the system in more extreme and unpredictable environments. The next steps are to work towards full operational readiness of the solver by increasing the complexity of the wind environment and range of the boundary conditions without sacrificing computation time. The state space should also be expanded into the 3rd dimension and include wind blowing in all directions as processed from the updated weather forecast file. In addition to improvements in the software element of the system, the hardware components onboard should be prepared in a manner to provide readable data to the BVP solver algorithm.

APPENDIX A:
Derivation of Multicopter Heading Angle

Collaboration with ENS Shawn Lee, USN, was done in order to understand the derivation and justification of this dynamical equation. Recall the equation for the wind triangle, Equation 2.9. It can also be written with the consideration of the ground heading angle and flight path angles:

$$V_g \begin{bmatrix} \cos\chi\cos\gamma \\ \sin\chi\cos\gamma \\ -\sin\gamma \end{bmatrix} - \begin{bmatrix} W_N \\ W_E \\ W_D \end{bmatrix} = V_a \begin{bmatrix} \cos\psi\cos\gamma_a \\ \sin\psi\cos\gamma_a \\ -\sin\gamma_a \end{bmatrix} \tag{A.1}$$

where γ is the flight path angle, the angle from the horizontal plane to the ground speed vector. γ_a is the angle from the horizontal plane to the airspeed vector. W_N, W_E, and W_D are the wind velocity components in the North, East, and Down direction, respectively. Considering the 2-D task where the altitude is constant, the flight path angle γ as well as wind W_D in the Down direction are assumed zero. By taking the time derivative of both sides of this equation and cancelling out terms that go to 1 or 0, the following expression is yielded:

$$\dot{V}_g \sin\chi + \dot{\chi} V_g \cos\chi = \dot{V}_a \sin\psi + \dot{\psi} V_a \cos\psi \tag{A.2}$$

Isolating the equation in terms of both $\dot{V}_g$ and $\dot{\psi}$ then yields:

$$\dot{V}_g = \frac{\dot{V}_a \cos\psi - V_a \dot{\psi} \sin\chi}{\cos\chi} + V_g \dot{\chi} \tan\chi \tag{A.3}$$

$$\dot{\psi} = \frac{\dot{V}_g \sin\chi}{V_a \cos\chi} + \frac{V_g \dot{\chi} \cos\chi}{V_a \cos\chi} - \frac{\dot{V}_a}{V_a} \dot{\chi} \tan\chi \tag{A.4}$$

Plugging the expression for $\dot{V}_g$ into $\dot{\psi}$ then yields:

$$\dot{\psi} = \frac{\dot{V}_a}{V_a} \tan(\chi - \psi) + \frac{V_g \dot{\chi}}{V_a \cos(\chi - \psi)} \tag{A.5}$$

The known equation for $\dot{\chi}$ is given by Beard and McLain in [5]:

$$\dot{\chi} = \frac{g}{V_g} \tan\phi \cos(\chi - \psi) \tag{A.6}$$

By plugging Equation A.6 into the previous equation and assuming that airspeed is constant (recall the 'fast autopilot' assumption in sec.2.2), the final equation for the change in heading angle is derived:

$$\dot{\psi} = \frac{g}{V_a} \tan\phi \tag{A.7}$$

It is important to realize that this equation holds true in the presence of wind.

APPENDIX B:
Derivation of Required Power

The following derivation of the energy expenditure model is based on the actuator disk and blade element theory that was provided in the Rotary-Wing UAV research paper by Zeng, et al [1]. The source categorizes the power requirement into 3 separate contributions: a blade profile power, induced power, and parasite power. The actuator theory states that the torque coefficient, q_c, is a contributing factor to the required power for cruise flight and can be represented as a function of the following variables.

$$q_c = \frac{\delta}{8}(1 + 3\mu^2) + (1 + k)\lambda_i t_{cD} + \frac{1}{2}\hat{V}^3 d_0 \tag{B.1}$$

where δ is the profile drag coefficient, μ is the advance ratio, k is an incremental correction factor to induced power, and λ_i is the mean induced velocity normalized by tip speed. Ω is the blade angular velocity in rad/s. v_{i0} is the mean rotor induced velocity in forward flight. t_{cD} is the thrust coefficient which can be written as a function of the thrust generated by the multicopter. $\hat{V}$ is the forward speed normalized by tip speed and is equal to μ. d_0 is the fuselage drag ratio. The advance ratio, mean induced velocity normalized by tip speed, and thrust coefficient are expressed as follows:

$$\mu = \frac{V_a}{\Omega R} \tag{B.2}$$

$$\lambda_i = \frac{v_{i0}}{\Omega R} \tag{B.3}$$

$$t_{cD} = \frac{T}{\rho s A \Omega^2 R^2} \tag{B.4}$$

$$\tag{B.5}$$

where V_a is the airspeed, T is the thrust of the multicopter, ρ is the air density, s is the rotor solidity ratio, A is the rotor disc area, R is the rotor radius. By substituting Equations B.3-B.5, a new expression of the thrust coefficient as a function of airspeed and thrust is yielded:

$$q_c(V_a, T) = \frac{\delta}{8}(1 + \frac{3V_a^2}{\Omega^2 R^2}) + \frac{(1 + k)\lambda_i T}{\rho s A \Omega^2 R^2} + \frac{1}{2} d_0 \frac{V^3}{\Omega^3 R^3} \tag{B.6}$$

The power required can then be written as a function of this torque coefficient for cruise flight, where

$$P = q_c \rho s A \Omega^3 R^3 \tag{B.7}$$

The equation for hovering power also defines two constants that are used in this derivation, P_0 and P_i. They can be represented by the following equations:

$$P_0 = \frac{\delta}{8} \rho s A \Omega^3 R^3 \tag{B.8}$$

$$P_i = (1 + k) \frac{W^{\frac{3}{2}}}{\sqrt{2\rho A}} \tag{B.9}$$

Substituting Equations B.8 and B.9, as well as the Equation B.5 into the required power equation yields the following:

$$P(V_a) = P_0 (1 + \frac{3V_a^2}{\Omega^2 R^2}) + P_i K (\sqrt{K^2 + \frac{V_a^4}{4v_0^4}} - \frac{V_a^2}{2v_0^2})^{\frac{1}{2}} + \frac{1}{2} d_0 \rho s A V_a^3 \tag{B.10}$$

It is known that the thrust-to-weight ratio is $K = \frac{T}{W}$. By referencing the balance of these two forces in Figure 2.4, the following relationship must hold in a constant altitude flight:

$$T \cos \phi = W \tag{B.11}$$

where the angle ϕ is the total inclination angle of the multicopter. Since the longitudinal component of ϕ angle is typically small for the 'not racing drones', the thrust in the longitudinal direction will be equal to weight, therefore leading to $K = 1$. However, the multicopter will still experience significant 'bank' angles in the lateral direction (roll maneuver) in order to change the direction of flight. Therefore, the lateral pitch angle, also known as bank angle, will still need to be included in the cost function to minimize the energy expenditure of the multicopter. Using the relationship from the previous equation, the following expression for the parameter K is derived:

$$K = sec(\phi) \tag{B.12}$$

This leads to the final equation for the required power in horizontal flight that can be written as a function of two control variables, airspeed and bank angle:

$$P(V_a, \phi) = P_0(1 + \frac{3V_a^2}{\Omega^2 R^2}) + P_i \sec\phi(\sqrt{\sec^2\phi + \frac{V_a^4}{4v_0^4}} - \frac{V_a^2}{2v_0^2})^{\frac{1}{2}} + \frac{1}{2}d_0\rho s A V_a^3 \tag{B.13}$$

This equation for required power in horizontal maneuvering flight implements the cost functional of the minimum energy trajectory optimization task and is parameterized by two controls V_a and ϕ.

Further analysis of the Equation (2.11) suggests that the required power is comprised of the sum of three individual factors - profile power, induced power, and parasite power. When the multicopter is drifting with wind in a hovering state, or when $V_a = 0$ and $\phi = 0$, the equation for required power is simplified to the following:

$$P_{req}^{hover} = P_0 + P_i \tag{B.14}$$

It makes sense that there should be no losses due to parasite power when the multicopter is hovering and that the required power to maintain this state is only dependent on the profile and induced elements.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C:
Kinematic Guidance Model

The following kinematic guidance model has been adopted from the textbook "Small Unmanned Aircraft" by Randal Beard and Timothy McLain to represent the kinematics of a generic multicopter flying in the presence of wind:

$$\dot{x} = V_a \cdot cos\psi \cdot cos\theta + W_x \tag{C.1}$$

$$\dot{y} = V_a \cdot sin\psi \cdot cos\theta + W_y \tag{C.2}$$

$$\dot{z} = V_g \cdot sin\theta + W_z \tag{C.3}$$

$$\dot{\theta} = \frac{g}{V_g}(n \cdot cos\phi - cos\theta) \tag{C.4}$$

$$\dot{\phi} = b_\phi(\phi^c - \phi) \tag{C.5}$$

$$\dot{\psi} = \frac{g}{V_a} \cdot tan\phi \tag{C.6}$$

$$\dot{\chi} = \frac{g}{V_g} \cdot tan\phi \cdot cos(\chi - \psi) \tag{C.7}$$

This truth model possesses 7 degrees of freedom. The states x, y, and z represent the multicopter's position in the inertial frame. It can be observed that the change in position of the multicopter is dependent on its airspeed, pitch angle, roll angle, and wind vector. The pitch rate, $\dot{\theta}$, is a function of the multicopter's ground speed (V_g), load factor (n), pitch angle and bank angle. The roll rate, $\dot{\phi}$ is controlled by the onboard autopilot and depends on a feedback control loop. The multicopter heading angle, ψ, is measured in the body frame and is a function of the airspeed and bank angle. The ground heading angle, χ, is measured in the inertial frame and therefore dependent on the ground speed and bank angle. It is important to note that the baseline kinematic model in this work excludes the vertical states and controls that makes the task feasible to solve within the constraints of the onboard micro-controller. The vertical position state z and the pitch angle θ are eliminated from the model with the assumption of working in the 2-D plane.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D:
Formulation of Scaling Problem

D.1 Definition of scaling parameters

$$V_{scale} = \frac{DU}{TU} = arbitrary \tag{D.1}$$

$$A_{scale} = \frac{DU}{TU^2} = arbitrary \tag{D.2}$$

$$T_{scale} = \frac{V_{scale}}{A_{scale}} = TU \tag{D.3}$$

$$D_{scale} = V_{scale} T_{scale} = DU \tag{D.4}$$

$$P_{scale} = \frac{D_{scale}^2}{T_{scale}^3} = \frac{DU^2}{TU^3} \tag{D.5}$$

D.2 Application of scales to the dynamics of states and co-states

$$\tilde{V_a} = \frac{V_a}{V_{scale}} \tag{D.6}$$

$$\tilde{W_x} = \frac{W_x}{V_{scale}} \tag{D.7}$$

$$\tilde{W_y} = \frac{W_y}{V_{scale}} \tag{D.8}$$

$$\tilde{g} = \frac{g}{A_{scale}} \tag{D.9}$$

$$\tilde{\psi} = \psi \tag{D.10}$$

$$\tilde{\phi} = \phi \tag{D.11}$$

D.3 Scaling of the state dynamics equations

Original State Equations

$$\dot{x} = V_a \cos\psi + W_x(x, y)$$
$$\dot{y} = V_a \sin\psi + W_y(x, y)$$
$$\dot{\psi} = \frac{g}{V_a} \tan\phi$$

Scaled State Equations

$$\dot{\tilde{x}} = \frac{d\tilde{x}}{d\tilde{t}} = \frac{d(x/D_{scale})}{d(t/T_{scale})} = \frac{dx}{dtV_{scale}} = \frac{\dot{x}}{V_{scale}} = \tilde{V}_a \cos\psi + \tilde{W}_x \tag{D.12}$$
$$\dot{\tilde{y}} = \frac{d\tilde{y}}{d\tilde{t}} = \frac{d(y/D_{scale})}{d(t/T_{scale})} = \frac{dy}{dtV_{scale}} = \frac{\dot{y}}{V_{scale}} = \tilde{V}_a \sin\psi + \tilde{W}_y \tag{D.13}$$
$$\dot{\tilde{\psi}} = \frac{d\tilde{\psi}}{d\tilde{t}} = \frac{d\psi}{d(t/T_{scale})} = \dot{\psi} T_{scale} = \frac{\tilde{g}}{\tilde{V}_a} \tan\phi \tag{D.14}$$

D.4 Scaling of the Co-State Dynamics Equations

Original Co-State Equations

$$\dot{\lambda}_x(t) = -\frac{\partial H}{\partial x} = -\lambda_x \frac{\partial W_x}{\partial x} - \lambda_y \frac{\partial W_y}{\partial x}$$
$$\dot{\lambda}_y(t) = -\frac{\partial H}{\partial y} = -\lambda_x \frac{\partial W_x}{\partial y} - \lambda_y \frac{\partial W_y}{\partial y}$$
$$\dot{\lambda}_\psi(t) = -\frac{\partial H}{\partial \psi} = \lambda_x V_a \sin\psi - \lambda_y V_a \cos\psi$$

Scaled Co-State Equations

$$\frac{\partial \tilde{W}_x}{\partial \tilde{x}} = \frac{\partial (W_x / V_{scale})}{\partial (x / D_{scale})} = T_{scale} \frac{\partial W_x}{\partial x}$$

This scaling of the wind is applied to all of the parts of the Jacobian that are included in the equations of the co-states, since they are all partial derivatives of a velocity term with respect to a distance term. It is shown below that a distance unit divided by a velocity unit can also be written as a time unit.

$$\dot{\tilde{\lambda}}_x(t) = -\frac{\partial \tilde{H}}{\partial \tilde{x}} = T_{scale}(-\lambda_x \frac{\partial W_x}{\partial x} - \lambda_y \frac{\partial W_y}{\partial x}) \tag{D.15}$$

$$\dot{\tilde{\lambda}}_y(t) = -\frac{\partial \tilde{H}}{\partial \tilde{y}} = T_{scale}(-\lambda_x \frac{\partial W_x}{\partial y} - \lambda_y \frac{\partial W_y}{\partial y}) \tag{D.16}$$

$$\dot{\tilde{\lambda}}_\psi(t) = -\frac{\partial \tilde{H}}{\partial \psi} = \lambda_x \tilde{V}_a \sin\psi - \lambda_y \tilde{V}_a \cos\psi \tag{D.17}$$

D.5 Scaling of the boundary conditions and the Hamiltonian

Scale Boundary Conditions

$$(\tilde{x}_0, \tilde{y}_0, \tilde{\psi}_0) = (\frac{x_0}{D}, \frac{y_0}{D}, \psi_0) \tag{D.18}$$

$$(\tilde{x}_f, \tilde{y}_f, \tilde{\psi}_f) = (\frac{x_f}{D}, \frac{y_f}{D}, \psi_f) \tag{D.19}$$

The Hamiltonian needs to be scaled by scaling all of its individual variables.

$$\tilde{H}(\lambda, x, y, \psi) = \frac{P_0}{P_{scale}}(1 + \frac{3\tilde{V_a}^2}{(\Omega T_{scale})^2 (R/D_{scale})^2}) +$$
$$\frac{P_i}{P_scale} \sec\phi(\sqrt{\sec^2\phi + \frac{\tilde{V_a}^4}{4(v_0/V_{scale})^4}} - \frac{\tilde{V_a}^2}{2(v_0/V_{scale})^2})^{\frac{1}{2}} + \frac{1}{2} d_0 (\rho * D_{scale}^3) s (A * D_{scale}^2) \tilde{V_a}^3 +$$
$$\tilde{\lambda_x}(\tilde{V_a}\cos\psi + \tilde{W_x}) + \tilde{\lambda_y}(\tilde{V_a}\sin\psi + \tilde{W_y}) + \tilde{\lambda_\psi}(\frac{\tilde{g}}{\tilde{V_a}}\tan\phi)$$

Scaled Hamiltonian

$$\tilde{H}(\lambda, x, y, \psi) = \tilde{P_0}(1 + \frac{3\tilde{V_a}^2}{\tilde{\Omega}^2 \tilde{R}^2}) + \tilde{P_i}\sec\phi(\sqrt{\sec^2\phi + \frac{\tilde{V_a}^4}{4\tilde{v_0}^4}} - \frac{\tilde{V_a}^2}{2\tilde{v_0}^2})^{\frac{1}{2}} +$$
$$\frac{1}{2} d_0 \tilde{\rho}^3 s \tilde{A}^2 \tilde{V_a}^3 + \tilde{\lambda_x}(\tilde{V_a}\cos\psi + \tilde{W_x}) + \tilde{\lambda_y}(\tilde{V_a}\sin\psi + \tilde{W_y}) + \tilde{\lambda_\psi}(\frac{\tilde{g}}{\tilde{V_a}}\tan\phi) \quad \text{(D.20)}$$

D.6 MATLAB code implementation of the scaled state dynamics

The following MATLAB code represents the formulation of the state dynamics in the BVP solver. The provided script is function that creates that set up the dynamical equations in their scaled form to be solved in the algorithm.

```
function dydx = odeFcn(x, y, T)
% This function reads the system dynamics of the multicopter as a
% set of differential equations and integrates them with respect
% to the provided time vector.

% Author: Matthew Modelo
% Date Created: 15 February 2021
% Date Last Edited: 18 May 2021

% Canonical Scaling Factors
```

```
V_scale = 8.058;
A_scale = 9.807;
T_scale = V_scale/A_scale;
D_scale = V_scale*T_scale;
P_scale = (D_scale^2/T_scale^3);

% fuel consumption model
d0 = UAV.d0;   % fuselage drag ratio (unitless)
rho = UAV.rho;  % air density (kg/m^3)
R = UAV.R;  % rotor radius (m)
A = UAV.A;    % rotor disc area (m^2), A = pi*R^2
Omega = UAV.Omega;  % blade angular velocity (rad/s)
∆ = UAV.∆;    % profile drag coefficient (unitless)
k = UAV.k;   % incremental correction factor to induced power ...
    (unitless)
m = UAV.m;  % mass of multicopter (kg)
s = UAV.s; % rotor solidity, ratio of total blade area to total ...
    disc area (unitless)
n = UAV.n;    % number of rotors (unitless)
P_0 = ((∆/8)*(rho*D_scale^3)*s*(A/D_scale^2)*...
      ((Omega*T_scale)^3)*((R/D_scale)^3)); % profile power, W
P_i = ((1 + k)*((m*(g/A_scale))^(3/2))...
      /sqrt(2*rho*A*D_scale)); % induced power, W
v_o = (sqrt((m*(g/A_scale))/(2*rho*A*D_scale))); %induced ...
    velocity, m/s

% Redefine states and co-states for readability
xvec = y(1);
yvec = y(2);
psi = y(3);
lx = y(4);
ly = y(5);
lpsi = y(6);

% Call the wind function using the time scale
[Wx, Wy, J] = Wind_Model( Vw, h, Sc, xvec*D_scale, yvec*D_scale, ...
    x.*abs(T)*T_scale);

% Optimal airspeed:
% This calls an external function that evaluates the optimal
```

```
% airspeed across the range of Lambda
Va = va_opt( UAV.Vopt_coef, lx.*cos(psi) + ly.*sin(psi) );

% Now calculate Vin0 in order to find the optimal bank angle
Tr = m*(g); %thrust in the longitudinal plane
Vin0 = sqrt( sqrt( (Tr/(4*rho*A))^2 + (Va).^4./4 ) - (Va).^2./2); ...
    % eq[65]

% Expressions for optimal bank angle, phi
% Derivation from 'optbank_1stordercond.mlx'
phi = -asin(lpsi./(Va.*Vin0.*m*(h/40))); %eqn solve method

% Maximum bank angle for the system is 10 degrees
phi( phi > phi_lim) = phi_lim;
phi( phi < -phi_lim) = -phi_lim;

% Scale the airspeed for the dynamics
Va = Va./V_scale;

% Dynamcis Equations (SCALED)
x_dot = Va.*cos(psi) + Wx./V_scale;
y_dot = Va.*sin(psi) + Wy./V_scale;
psi_dot = ((g/A_scale)./Va).*tan(phi);
lx_dot = (-lx.*J(1,1) - ly.*J(2,1))*T_scale;
ly_dot = (-lx.*J(1,2) - ly.*J(2,2))*T_scale;
lpsi_dot = lx.*(Va).*sin(psi) - ly.*(Va).*cos(psi);

dydx = T_scale*T*[x_dot
                  y_dot
                  psi_dot
                  lx_dot
                  ly_dot
                  lpsi_dot];


end
```

D.7 MATLAB code implementation of the scaled boundary conditions and Hamiltonian

The following MATLAB code illustrates the implementation of the scaled boundary conditions and Hamiltonian into the algorithm. The provided script is a function that scales the boundary conditions and all terms in the Hamiltonian before providing the information in a final package to be read by the algorithm.

```
function res = BC(ya, yb ,T)
% This function defines the necessary boundary conditions to be met
% in the task.

% Author: Matthew Modelo
% Date Created: 15 February 2021
% Date Last Edited: 18 May 2021

% Canonical Scaling Factors
V_scale = 8.058;
A_scale = 9.807;
T_scale = V_scale/A_scale;
D_scale = V_scale*T_scale;
P_scale = (D_scale^2/T_scale^3);

% fuel consumption model (IN SCALED UNITS)
d0 = UAV.d0;   % fuselage drag ratio (unitless)
rho = UAV.rho;  % air density (kg/m^3)
R = UAV.R;  % rotor radius (m)
A = UAV.A;    % rotor disc area (m^2), A = pi*R^2
Omega = UAV.Omega;  % blade angular velocity (rad/s)
ɑ = UAV.ɑ;    % profile drag coefficient (unitless)
k = UAV.k;   % incremental correction factor to induced power ...
    (unitless)
m = UAV.m;  % mass of multicopter (kg)
s = UAV.s; % rotor solidity, ratio of total blade area to total ...
    disc area (unitless)
n = UAV.n;    % number of rotors (unitless)
P_0 = ((ɑ/8)*(rho*D_scale^3)*s*(A/D_scale^2)*...
      ((Omega*T_scale)^3)*((R/D_scale)^3)); % profile power, W
```

```
29 P_i = ((1 + k)*((m*(g/A_scale))^(3/2))...
30       /sqrt(2*rho*A*D_scale)); % induced power, W
31 v_o = (sqrt((m*(g/A_scale))/(2*rho*A*D_scale))); %induced ...
      velocity, m/s
32
33 Va = UAV.Va; % Nominal airspeed of the vehicle
34
35 % Optimal airspeed and tf
36 Va_f = va_opt( UAV.Vopt_coef, yb(4)*cos(yb(3)) + yb(5)*sin(yb(3)));
37
38 % Initial Conditions
39 x_0 = ya(1);
40 y_0 = ya(2);
41 psi_0 = ya(3);
42 lx0 = ya(4);
43 ly0 = ya(5);
44 lpsi0 = ya(6);
45
46 % Final Conditions
47 x_f = yb(1);
48 y_f = yb(2);
49 psi_f = yb(3);
50 lx_f = yb(4);
51 ly_f = yb(5);
52 lpsi_f = yb(6);
53
54 % Call the wind function to determine the state of the wind at the
55 % final condition
56 [Wx,Wy,¬] = Wind_Model(Vw, h, Sc, x_f*D_scale, y_f*D_scale, ...
      abs(T)*T_scale);
57
58 % Now calculate Vin0 in order to find the optimal bank angle
59 Tr = m*(g); %thrust in the longitudinal plane
60 Vin0 = sqrt( sqrt( (Tr/(4*rho*A))^2 + (Va_f)^4/4 ) - (Va_f)^2/2); ...
      % eq[65]
61
62 % Expressions for optimal bank angle at tf, phi
63 % Derivation from 'optbank_1stordercond.mlx'
64 phi_f = -asin(lpsi_f/(Va_f*Vin0*m*(h/40))); %eqn solve method
65
```

```
% Physical limit for bank angle
phi_f( phi_f > phi_lim) = phi_lim;
phi_f( phi_f < -phi_lim) = -phi_lim;

% Scale the airspeed for the Hamiltonian
Va_f = Va_f/V_scale;

% Build the Hamiltonian at final time, tf
Profile_f = P_0*(1 + ...
    (3*Va_f^2)/(((Omega*T_scale)^2)*((R/D_scale)^2)));
Induced_f = P_i*sec(phi_f)*(sqrt((sec(phi_f)^2) + ...
    (Va_f^4)/(4*(v_o^4)))...
    - (Va_f^2)/(2*(v_o^2)))^(1/2);
Parasite_f = (1/2)*d0*rho*s*A*(Va_f^3)*D_scale;

P_f = n*(Profile_f + Induced_f + Parasite_f);

Hf = P_f + ...
    lx_f*(Va_f*cos(psi_f) + Wx(end)/V_scale) + ...
    ly_f*(Va_f*sin(psi_f) + Wy(end)/V_scale) + ...
    lpsi_f*(((g/A_scale)/Va_f)*tan(phi_f));

% Definition of boundary conditions that must be met
res = [x_0 - IC(1)
       y_0 - IC(2)
       psi_0 - IC(3)
       x_f - FiC(1)
       y_f - FiC(2)
       psi_f - FiC(3)
       Hf ];        % H(tf) = 0

end
```

THIS PAGE INTENTIONALLY LEFT BLANK

List of References

[1] Y. Zeng, J. Xu, , and R. Zhang, "Energy minimization for wireless communication with rotary-wing uav," IEEE, Department of Electrical and Computer Engineering, National University of Singapore, Tech. Rep., 2018.
[Online]. Available: arXiv:1804. 02238v1

[2] V. Dugar, S. Cloudbury, and S. Scherer, "Smooth trajectory optimization in wind: First results on a full-scale helicopter," AHS International, Fort Worth, Texas, USA, Tech. Rep., 2017.

[3] A. Jatho, "Optimizing energy eÿcient uav routing in support of marine corps expeditionary advanced base operations," M.S. thesis, Dept. of Operations Research, NPS, Monterey, CA, USA, 2020. [Online].

[4] R. Beard and T. McLain, *Small Unamanned Aircraft*, 1st ed. 41 William St, Princeton, NJ, USA: Princeton University Press, 2012.

[5] R. Peter, "Coamps-os® 250mb jet stream wind product," NRL, Jan. 30, 2018. [Online]. Available: https://cavu.nrlmry.navy.mil/COAMPSOS/pascal.nrlmry.navy.mil/coamps-os/cgi-bin/COAMPSOS_vis_help_wrapper.tcl3fname=def-250mb-wind-wspd0

[6] W. Shelby. "System Modeling." Accessed Feb. 18, 2021. [Online]. Available: https: //www.wilselby.com/research/arducopter/modeling/

[7] V. N. Dobrokhodov, C. Walton, I. I. Kaminer, and K. Jones, "Energy-optimal trajectory planning of hybrid ultra-long endurance uav in time-varying energy fields," AIAA, Center for Autonomous Vehicle Research, Monterey, CA, 93943, Tech. Rep., 2020.

[8] G. Bader and P. Kunkel, "Continuation and collocation for parameter-dependent boundary value problems," Society for Industrial and Applied Mathematics, Department of Electrical and Computer Engineering, National University of Singapore, Tech. Rep. Vol. 10, No. 1, pp. 72-88, 1989.

THIS PAGE INTENTIONALLY LEFT BLANK

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California