INTEGRATED TARGETING, GUIDANCE, NAVIGATION, AND CONTROL FOR UNMANNED AERIAL VEHICLES

A DISSERTATION SUBMITTED TO THE GRADUATE DIVISION OF THE
UNIVERSITY OF HAWAIʻI AT MĀNOA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN

MECHANICAL ENGINEERING

DECEMBER 2020

By

Evan Kawamura

Dissertation Committee:

Dilmurat Azimov, Chairperson
John Allen
Zhouyuan Song
Sarah Post
Corey Ippolito

Keywords: targeting, guidance, navigation, control, unmanned aerial vehicles

To my family and God,

for their continuous love and support. To God, be the glory!

ACKNOWLEDGMENTS

Thank you Dr. Dilmurat Azimov for your patience and optimism throughout this long journey. Thank you for your mentorship and vast knowledge on targeting, guidance, navigation, and control. I have been so blessed and fortunate to have been mentored by one of the best, and it is has been an honor and privilege to learn from you.

Thank you committee members for serving on my committee and for your constructive feedback and criticism, which ultimately improved my work and helped with professional development. Thank you to all the HESTEMP and ACTUAS members, participants, mentors, and collaborators who supported me.

Thank you to the NASA Ames Research Center administrators and SAFE50 team members for your support during my summer internships. Special thanks to Nhan Nguyen and Kalmanje Krishnakumar, for your words of wisdom, advice, and expertise. Special thanks to Sean Swei for the helpful and insightful discussions regarding modeling and filtering and to Don Soloway for the invaluable and long discussions regarding gating networks and neural networks. Biggest mahalo to Corey Ippolito for mentoring me since summer 2018, serving as a member on my committee, and for being the bridge between NASA Ames Research Center and the University of Hawaiʻi at Mānoa through the NASA EPSCoR project.

Thank you to my prayer warriors for your support and words of encouragement: Mom, Dad, grandparents, aunties, uncles, Curtis Yoneda, Gilbert Ignacio, Jamie Lam, Josh Shiroma, Lionel Goo, Shaun Grindle, and Tina Li. Thank you God for your love, wisdom, guidance, support, provision, forgiveness, patience, and placing me at the right places at the right times with the right people when You knew I needed them most. Thank You for reminding me that I can do all things through You who gives me strength.

ABSTRACT

The goal of this dissertation research is to demonstrate the integration of targeting, guidance, navigation, and control (TGNC) functions for real-time implementation onboard unmanned aerial vehicles (UAVs) for a wide range of applications. This allows us to create a robust and accurate integrated TGNC software platform for UAVs, which enables them with real-time capabilities and leverages the flight autonomy. Target-relative guidance, real-time targeting and re-targeting capabilities are of great interest in today's UAV technology. This research proposes new guidance and estimation methods as well as new extremal control laws for UAV applications. In particular, this research focuses on quadcopter applications. The proposed guidance methods represent an extension of the existing explicit translational guidance (E-guidance) to include rotational guidance and exponential braking guidance to reach target points. The proposed estimation method is a hierarchical mixture of experts (HME) framework with extended Kalman filters (EKFs) and a modified softmax function to provide state and parameter estimations for navigation solutions. The proposed research utilizes the Hamiltonian formalism with the indirect method to solve the optimal control problem, which replaces existing PID control laws with extremal control laws based on first-order optimality conditions. Three illustrative examples demonstrate integration of targeting, guidance, and control functions for takeoff, waypoint, and roll maneuvers of quadcopters. It is shown that the proposed HME framework with acoustic parameters demonstrates a viable navigation solution. Implementation of the TGNC functions through the proposed HME, target-relative guidance, and extremal control with simulated acoustic parameter measurements demonstrates a completely integrated TGNC software system for quadcopters. Novelties of the proposed research include extension of E guidance, simulating an exponential braking guidance law to reach a target point, determination of the switching function for max-intermediate thrust arcs, and design and validation of a HME framework to provide navigation solutions. The proposed research results can be used to address environmental and agricultural problems that utilize UAVs. This research has been funded, in part, by the NASA EPSCoR ACTUAS (Autonomous Control Theory - Unmanned Aerial Systems) project. The core research contributions are deriving E Guidance for rotational maneuvers extending E Guidance to higher order integration methods, integrating E Guidance with extremal control satisfies the boundary conditions to yield an extremal for the guided trajectory, and incorporating acoustics with EKFs in HME shows the impact of considering several different models and parameters to have accurate state estimation. Future work may encompass determining accurate dynamic thrust and acoustic models and considering second-order conditions to determine optimal control with a corresponding trajectory.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 Goal

The goal of the proposed research is to design, develop, integrate and validate targeting guidance, navigation, and control (TGNC) software algorithms for an autonomous and real-time implementation onboard of unmanned aerial vehicles (UAV) for agricultural and environmental applications in the Hawaiian Islands [1]. The proposed TGNC integration will be demonstrated by the execution of the software algorithms for specified maneuvers, such as the monitoring and inspection of the fields of macadamia nuts, coffee plantations, and vegetation near the coastal areas of the Hawaiian Islands such as Oahu, Hawaii island, and Kauai under various flight conditions, including beyond line of sight conditions. The proposed research paves the way for a wide range of possibilities and opportunities for UAV development with potential applications in the Central Pacific and Micronesian Islands. This work will contribute to the fulfillment of the NASA strategic plans and current efforts to create an efficient and safe national airspace [4].

1.2 Significance

Digital elevation maps help provide information about Oahu's potential flood zones due to the rising sea levels, which is a time-sensitive issue, especially for an island with numerous buildings and structures near coastlines in locations such as Waikiki or Kaka'ako [5]. Strategic planning through these digital elevation maps can lead to coastal surveys and analysis to develop solutions to prevent exposing "approximately 6,500 structures and 19,800 people statewide" to potential flooding, which will occur in the next 30 to 70 years [6]. Overall, utilizing UAVs with TGNC capabilities for generating digital elevation maps can aid in these efforts.

1.3 Dynamics and Control of UAVs

This research emphasizes developing guidance, navigation, and control (GNC) schemes to leverage autonomy by enabling UAVs with targeting capabilities in performing various tasks and maneuvers. In the last two decades, new nonlinear control laws and schemes have been developed [7, 8, 9]. In particular, the existing onboard control systems primarily utilize well-known control schemes such as linear or proportional controllers, PID controllers, and tracking controllers implemented by the flight controllers, including "Pixhawk" and "Omnibus F4 V2" along with DJI, Mission Planner, and INAV software platforms [10, 11]. These controllers are primarily designed for ground-based computations and implemented onboard using iterative processes with humans-in-the-loop. However, the efficiency, optimality, and utility of these existing control laws and schemes onboard UAVs have yet to be studied and tested for applications. The implementation of these control laws and schemes require real-time computation of the state and control vectors to improve the accuracy and leverage the autonomy of UAVs. At the same time, onboard computation of the state and control vectors requires a) efficient, simple, and explicit control laws, b) computationally inexpensive estimation methods, c) simplistic guidance schemes or laws. Besides mathematically resolving linear PID limitations with nonlinear controllers, nonlinear controllers may have engineering merits over linear PID controllers. Comparing battery voltages between PID controllers and nonlinear controllers for maneuvers provides pilots insight in choosing between the controllers to save battery life during flights. Having this information allows them to prefer one controller over another if they desire to optimize battery life to have longer flight times. This research focuses on TGNC, but there are generally more studies devoted to GNC than TGNC. Thus, the literature review in the next section describes what has been done in UAV GNC research.

1.4 Literature Review

1.4.1 TGNC vs. GNC

Below is a comparison between GNC and TGNC, where the biggest differences are targeting and lack of ground control to provide navigation and guidance commands in conventional GNC. TGNC is important in UAVs because pilots typically desire to compute waypoints ahead of time or generate new waypoints during a mission. In the UAV community, the waypoints are target points, so targeting in UAVs involves computing waypoints. Typically, targets are pre-calculated, so deviations from the nominal trajectory would cause the mission to fail if there are unaccounted obstacles. Targeting provides information to guidance where the vehicle should go to avoid obstacles or to follow a nominal trajectory. Control gives the vehicle's actuators commands to move towards the target, and navigation provides state estimation of the vehicle. Overall, enabling the UAV to have autonomous TGNC capabilities allows it to search for targets or areas of interest without humans providing trajectories and waypoints of where to search, which increases autonomous capabilities. Section 3.1 provides definitions and more details of each of the TGNC functions.

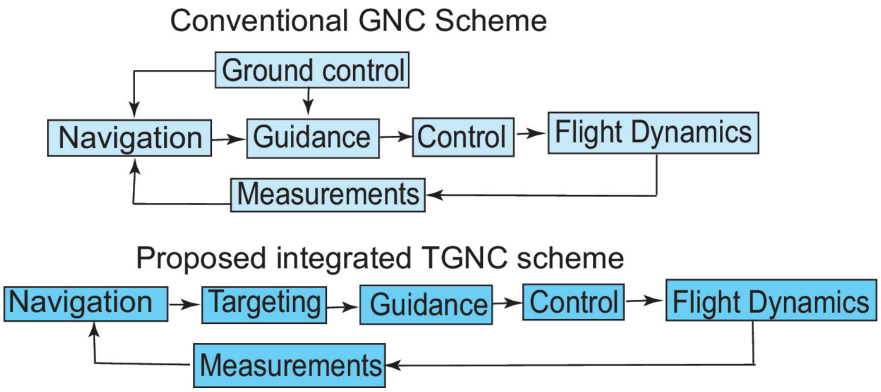


Figure 1.1: GNC & TGNC Architecture [1]

1.4.2 Guidance Functions

Guidance laws are not discussed as frequently as control or navigation functions in UAV flight software. However, UAV flight software such as PX4 and Ardupilot utilize a modified L1 controller, allowing tighter tracking of curved flight paths. The L1 controller got its name from drawing a vector, L1, from the vehicle's current position to a reference point along the desired trajectory. Ardupilot had a request from a user, which led to implementing and modifying the L1 controller based on MIT's work on nonlinear guidance for tight tracking [12, 13]. Ardupilot's tracking algorithm for circles during RTL, GUIDED, and LOITER modes is slightly different from MIT's guidance logic because Ardupilot uses a modified PD control law instead of the L1 controller [14]. Similarly, PX4 also adapts the concepts of the same L1 controller but also uses PD control for tracking circles with a loiter radius less than the length of L1 [15].

Lee et al. designed a nonlinear guidance and control system based on a sliding-mode control scheme for UAVs to automatically land on carriers out at sea [16]. Explicit, closed-form, analytical solutions of UAV non-steady flight in a vertical plane in which the solutions are explicit and complete, leading to targeting and guidance solutions [17]. Viswanathan et al. combine feedback control and guidance to send quadcopters to desired waypoints with a desired attitude trajectory [18]. Lu presents an entry guidance method using a single baseline predictor-corrector algorithm as the baseline algorithm for a wide range of vehicles "with varying lifting capabilities" [19].

1.4.3 Navigation Functions

Open-source UAV software such as Ardupilot and PX4 utilize Extended Kalman Filters for estimating the state vector. Ardupilot's EKF provides estimation for 22 states: position, velocity, quaternions, gyro biases, accelerometer Z bias, wind velocity, compass biases, and the earth's magnetic field. [20]. PX4 has implemented an EKF using their Estimation and Control Library (ECL). It provides state estimation of 24 states: the quaternions for rotating from the north, east, down local earth frame to the UAV's 3D Cartesian body frame, velocity at IMU, position at IMU, IMU angle errors, IMU angular velocity errors, earth magnetic field components, vehicle body frame magnetic field bias, and wind velocity [21, 22]. Betaflight, Cleanflight, and INAV currently do not use EKF or other types of Kalman filters [23]. However, Butterflight, a fork of Betaflight, has a Fast Kalman Filter (FKF) but only seems to provide estimates for gyros [24, 25].

1.4.3.1 Non-Hierarchical Mixture of Experts Navigation Methods

Kolpuke et al. were inspired by insects and birds to use a solar position algorithm to estimate a UAS's position when the "azimuth and zenith/incidence angles for solar position, attitude of the aircraft, and time are known." They propose to calculate the observer's position from the solar position and time [26]. Mendoza-Soto and Cortes tracked a quadcopter's trajectory using generalized predictive control with a geometric attitude controller. Optimal control actions are used in generalized predictive control to satisfy constraints on desired trajectories. [27]. Vetrella et al. improved attitude estimation using relative vision-based sensing and carrier-phase differential GPS between antennas. The multirotor flight test analyzes differential GPS and carrier-phase differential GPS solutions. Their experimental data demonstrated that the carrier-phase differential GPS had less noise than the code-based differential GPS. [28]. Gatter and Andert tested a model-based shadow estimation method on an unmanned helicopter to find self-cast shadows. Their algorithm can detect shadow positions at accuracies over 95% [29].

1.4.3.2 Hierarchical Mixture of Experts

Alpaydin discusses mixtures of experts as a generalization of a radial basis function (RBF) network, which uses supervised learning and is faster than multilayer perceptrons with distributed representations [30]. Zhang also describes hierarchical mixtures of experts as an extension of a mixture of experts, where the input space divides into regions, and each region performs data fitting independently. Figure 1.2 shows Zhang's figure of the HME architecture [2].

Chaer et al. utilized a mixture-of-experts framework for adaptive Kalman filtering with a gating network, where each expert is a Kalman filter with different system parameters and dynamical models [31]. Chaer et al. expanded [31] by utilizing multiple levels in the HME architecture and applying it to the Mars Pathfinder interplanetary navigation problem [32]. Jacobs and Jordan use a modular neural network to learn how to perform control tasks with a piecewise strategy and a trained adaptive feedforward controller with a fixed PID feedback controller to control a two-joint planar manipulator robot arm to carry six different payloads with different masses along a pre-determined trajectory [33]. Rao et al. implemented a deterministic annealing (DA) method in a hierarchical mixture of experts framework for regression modeling, which gradually reduces the system entropy and minimizes the cost at every entropy level with Lagrange multipliers acting as "temperature" parameters [34]. Boroujeni and Charkari utilized a hierarchical mixture of

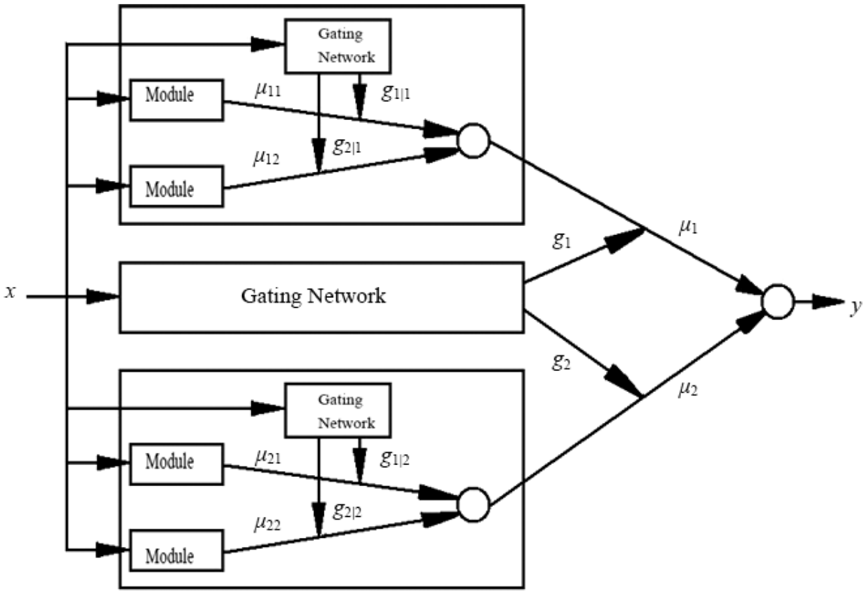


Figure 2. Architecture of a two-layer hierarchical mixture of experts.

Figure 1.2: HME Diagram [2]

multilayer perceptrons (MLP) experts with semi-supervised teacher-directed learning to detect shadows, and their proposed method has an average accuracy of about 85%, which is a 5-10% improvement in detection and discrimination rates for different environmental conditions [35].

Jordan and Jacobs present and apply an expectation-maximization (EM) algorithm to the HME architecture [36]. Titsias and Likas utilize a three-level HME architecture for classification problems, and their results demonstrate that both of their HME methods generally outperform the separate mixtures and common component models in most of the trials [37]. Ng and Deisenroth use HME for large-scale Gaussian process (GP) regression, and their model is practical and efficient by learning quickly, performing accurately, consuming little memory, and avoiding high-dimensional variational optimization of inducing inputs [38].

Bishop and Svensen apply a Bayesian treatment of HME, which avoids bias from maximum likelihood and optimizes the complexity and topology of the HME graph, and they used their Bayesian HME model to model the forward kinematics of an 8-link all-revolute robot arm to predict the distance between the end of the arm and a specified point [39]. Yuksel et al. provide a comprehensive survey of research involving mixture of experts with a wide range of applications relevant to robotics such as sensor fusion, deformable model fitting, filter selection, robot navigation, 3D object recognition, image transport regression, and nonlinear system identification of a robotic arm. A vital advantage of HME over other methods is that HME is flexible by working with different types of models and learning model parameters [3].

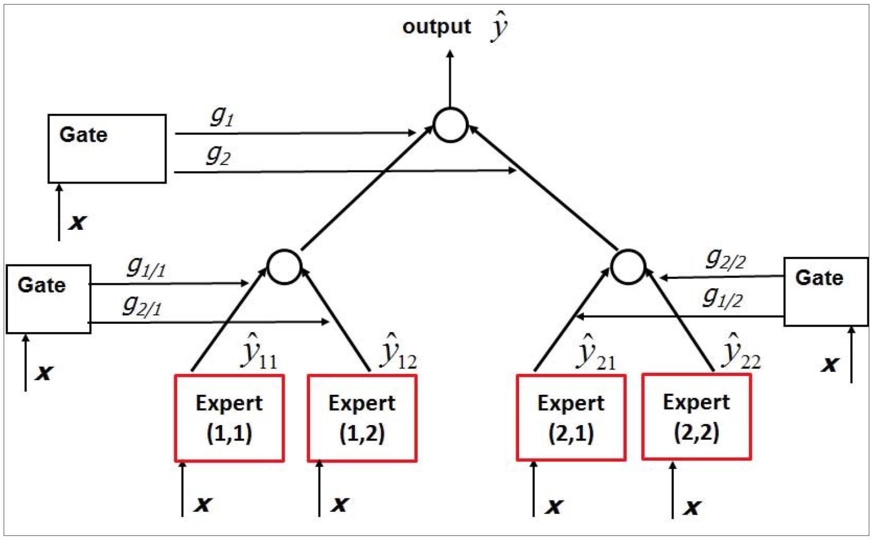


Figure 1.3: Two-level HME architecture for regression [3]

1.4.4 Control Functions

UAVs usually use linear controllers such as PID controllers, but these linear controllers linearize the UAV's nonlinear system [40, 41, 42, 43, 44]. PID controllers' major drawback is that tuning is specified to a particular system. Having the same PID coefficients but for different size quadcopters most likely will not work because PID tuning depends on the system. PID tuning is also subjective and relative to the pilot. Another drawback of linearizing nonlinear systems is that large perturbations decrease the accuracy, especially when the solution is not near equilibrium. For instance, the nonlinear and linear equations of motion of a rotating pendulum without friction converge for $\omega < \sqrt{\frac{g}{l}}$ where ω is the angular velocity, g is the acceleration from gravity, and l is the length of the pendulum. At higher angles, the solutions diverge [45]. Linearizing nonlinear systems should be avoided by having nonlinear controllers to reduce errors.

Garcia et al. use a nonlinear H-infinity approach with L2-gain and dissipativity concepts to control UAVs with a solution to a Hamilton-Jacobi-Isaacs equation by iteratively solving the partial differential equations with a state-feedback Taylor series expansion [46]. Ibarra and Castillo used a nonlinear super-twisting controller based on singular optimal control and Lyapunov theory with the sliding mode technique to stabilize a quadcopter's nonlinear attitude [47]. A nonlinear control study of aerial manipulators uses an optimizer and feed-forward action to optimize the a priori free degree of freedom in real-time, which increases efficiency and prevents instability in dangerous environments or configurations [48].

Matus-Vargas et al. use two optimization techniques to tune PD coefficients for nonlinear UAV dynamical models through conjugate gradient optimization strategies, and their second approach used Pontryagin's maximum principle, which lacked steady-state error, had a fast response, and demonstrated superior performance compared to PD controllers [49]. Yu et al. used a Markov decision process algorithm with a backtracking method that generates collision-free trajectories in a hostile environment with static and dynamic obstacles, and their flatness-based approach allows the UAV to avoid aggressive maneuvers and attempt to maintain flying on the collision-free trajectory [50]. Beall shows that adaptive control is a successful alternative to PID control by optimizing the system's performance concerning characteristics such as damping ratio and settling time [51].

Holtsov et al. use adaptive control for the UAV roll channel and the μ-synthesis method to establish robust control for stability and performance at varying airspeeds, demonstrating that adaptive control is useful for systems with a broad range of parameter changes [52]. Frazzoli et al. use a quadratic Lyapunov function with PD control for translational dynamics for a helicopter, and all four of the simulations had good results except despite having a singularity at zero thrust [53]. Waslander and Hoffmann used Integral Sliding Mode and Reinforcement Learning control, which performed better than classical control methods for flying quadrotor drones using outdoor altitude control [54]. Noble and Bhandari used an extended minimum resource allocating network adaptive algorithm in a model reference adaptive controller for controlling the pitch and roll of a fixed-wing unmanned aerial vehicle, which had better roll and pitch responses than PD or PID controllers [55]. Lunni et al. use a nonlinear model predictive control controller for unmanned aerial manipulators with the main task of performing 3D tracking of its arm effector [56].

Levin et al. use the rapidly-exploring random trees (RRT) algorithm for agile fixed-wing UAV motion planning with knife-edge maneuvers, and the UAV successfully flew and switched into knife-edge maneuvers through narrow passages and then reverted to normal flight once out of the passage for constant altitudes [57]. Bulka and Nahon utilize an autonomous controller for an agile fixed-wing UAV to perform complicated maneuvers such as knife-edge, rolling Harrier, hover, and aggressive turnaround, and their single control system with three subcontrollers could manage the knife-edge, rolling Harrier, hover, and aggressive turnaround maneuvers [58]. Pugach et al. use a nonlinear fixed-wing adaptive controller with an echo state network to replace Ardupilot's roll PID controller, and their ESN performs well even during disturbances and noises [59].

It is known that many research studies have been devoted to the development of control technology of unmanned aerial vehicles (UAVs), but not many works deal with or address questions related to optimality. It has been shown that none of the works surveyed here used the Hamiltonian based on the dynamical model of UAVs. One example includes using the Hamilton-Jacobi-Bellman equation but based on a Dubins vehicle model instead of the dynamical model of UAVs [60]. Beul and Behnke used the Hamiltonian but simplified the optimal trajectory and control with an ad hoc, intuitive approach by breaking the optimal trajectory into seven sections [61]. Another work showed that the optimal controls and trajectories could be expressed as Legendre polynomials, but neither the Hamiltonian nor the Euler-Lagrangian equations are used to determine extremals or satisfy the necessary or sufficient conditions of optimality [62]. It has been shown that linearizing nonlinear models of flight vehicles leads to linear controllers such as PID controllers [63].

It was shown that the $\mathbb{H}_\infty$ synthesis using robust control theory performed better than LQR synthesis with optimal control theory[64]. Another work used task-oriented optimal control strategies to improve flight time by generating energy-optimal coordinated motions. A Model Predictive Control (MPC) strategy tracks trajectories, and a PD controller creates set-point velocity screws as inputs to MPC [65]. Others showed that a reinforcement learning based controller with least squares policy iteration (LSPI) can learn the optimal control policies required for generating landing trajectories for a Parrot AR drone 2.0 [66]. Another work developed a flight controller that combines baseline, feedforward, and adaptive components and works with an optimal trajectory generation algorithm [67].

There are recent studies in optimal control for UAVs at the 2019 ICUAS conference. A nonlinear model predictive control scheme computes optimal controls and trajectories in GPS denied environments [68]. A Least Squares Policy Iteration determines optimal controls for tracking a target drone while maintaining a fixed distance away from the target [69]. Using Pontryagin's Minimum Principle determines optimal trajectories for delivering fragile packages with a simple, linear dynamical model [70]. Numerical steepest descent solves the Mayer optimization problem for a guided projectile in a vertical plane [71]. Vision-based autonomous obstacle avoidance is solved using the algebraic Riccati equation for an LQR problem [72]. Minimum energy paths are determined by using GPOPS II, general-control software implemented in MATLAB, to solve nonlinear optimal control problems with variable-order adaptive orthogonal collocation and sparse nonlinear programming [73]. The Continuous-time Algebraic Riccati Equation uses Lie algebra to extend the LQR problem and linearizes the quadcopter's state at each time step [74]. Adaptive optimal guidance for UAV path following uses a State Dependent Riccati Equation approach with a simple aircraft dynamical model [75].

One can solve optimal control problems by using one of four methods: 1) Lagrange's formalism, 2) Hamilton-Jacobi-Bellmann's formalism, 3) direct and indirect methods, and 4) unique formalism based on the sufficient conditions of absolute optimality. Lagrange's formalism works if the set of admissible controls is open, and this method causes the solution to be reduced to the boundary problem and to test the conditions of optimality. The Hamiltonian-Jacobi-Belmann's formalism uses sufficient optimality conditions and reduces the problem of integrating the Hamilton-Jacobi equation in partial derivatives, which creates a field of extremals to solve the problem of optimal synthesis of trajectories. The direct and indirect methods solve the optimal control problem by approximating the minimum with some system of functions and Lagrange multipliers. Unique formalism uses the sufficient conditions of optimality and finds an absolute minimum. This method reduces the problem to the Jacobi-Bellmann equation or a boundary problem [76].

1.4.5 Path Planning

Chandarana et al. had thirteen subjects use gesture-based interfaces and mouse-based interfaces to define trajectories to generate paths for UAVs [77]. Sathyaraj et al. compared A* algorithm, Dijkstra's algorithm, Bellman-Ford's algorithm, and Floyd-Warshall's algorithm for UAVs and found A* performed best [78]. Ten Harmsel et al. had two emergency sensor-based and map-based planners to plan safe paths for UAVs using the A* algorithm with a cost function that considered a movement cost and an environment cost [79]. Olson et al. used a cost map with the A* algorithm to find safe landing paths for low energy quadrotor UAS flying in populated areas [80]. Zheng et al. designed a path planning algorithm using two-dimensional LIDAR data with static global path planning and dynamic local path planning, and the optimal path comes through a multi-objective optimization heuristic function [81]. Zhou and Gao used the global-best brain storming algorithm for avoiding static obstacles, and simulations show that global-best brainstorming and artificial potential field are effective for generating 3D trajectories while avoiding obstacles [82].

1.4.6 UAV Acoustic Studies

Benyamin and Goldman detected and tracked a UAV with a tetrahedral microphone array with beamforming and adaptive Kalman filters [83]. Feight et al. showed quadcopter rotor spectral characteristics through testing in an anechoic chamber with microphones placed radially around the UAV [84]. Kapoor et al. showed that acoustic sensors are useful in GPS-denied environments for obstacle detection and localization [85]. Kloet et al. conducted psychoacoustic analysis on a DJI Phantom 4 and a 250-size mini-quad [86]. Papa et al. evaluated UAV noise during typical flight operations from electric engines and propellers by taking measurements inside an anechoic room [87]. Leonardo and Chen tracked wildlife with a low-cost RF telemetry system onboard a fixed-wing aircraft, AggieAir [88]. Case et al. use a low-cost acoustic array to find and track small UAVs [89]. Finn and Franklin show successful 360° sense and avoid UAV capabilities with acoustic sensors to detect, track, and avoid obstacles autonomously [90]. Kloet et al. showed relationships between sound pressure level, distance, and altitude for a quadcopter [91]. Farassat describes the noncompact and compact source formulas for rotating blade noise based on the Ffowcs Williams-Hawkings (FW-H) equation [92]. Tinney and Valdez analyzed the thrust and acoustic performance for a coaxial corotating rotor in hover, and the rotor speed and sound depend more on the index angle than the stacking distance [93]. Tinney and Sirohi provide experimental results for acoustic analysis of quadcopter and hexacopter drones for only hovering conditions in terms of propeller size, propeller angular velocity, and the number of propeller blades [94]. Tinney and Valdez determine scaling laws for sound pressure levels of quadcopters while hovering by considering overall sound pressure level, sound pressure level, blade-pass frequency harmonic noise, rotor thrust, torque, power and blade tip speed, thrust and power coefficients, Reynolds number, blade tip Mach number, and figure of merit, defined as the ratio between the thrust and power coefficients

[95]. Huff and Henderson provide acoustic measurements of electric motors for small quadcopters with and without propellers [96].

1.4.7 UAV Flight Control Software: State of the Art

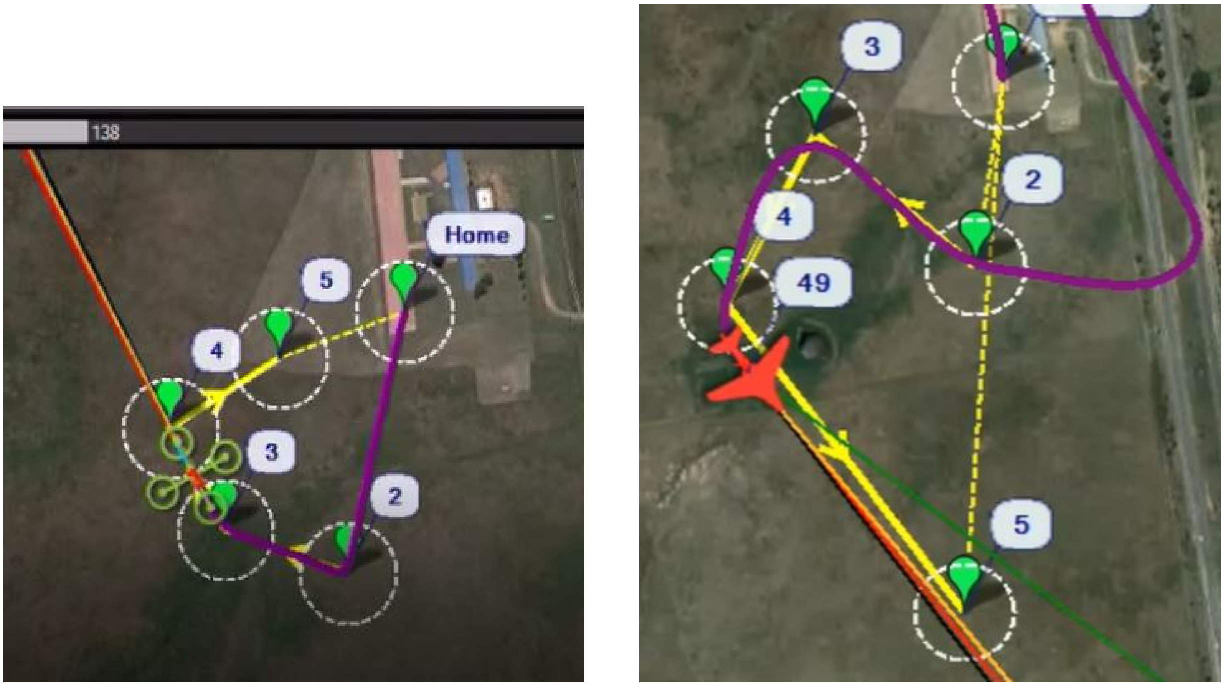
Common open-Source autopilot systems for inexpensive vehicles: ArduPilot & PX4. Downloading computer applications and installing several packages and functions in Ubuntu 14.04 (Linux) is required. The Linux computer applications utilized are Notepad++ for viewing a class's list of functions, QT Creator as the IDE (Integrated Development Environment) for viewing and editing the code, Cygwin terminal for simulation Graphical User Interface (GUI), Mission Planner for simulations and interfacing the UAV with the user, gedit for editing make files and doxyfiles, and doxygen for creating call graphs and function maps based on the code's documentation. The open-source Ardupilot code is cloned from Github, recursively updated, and then the Pixhawk's firmware gets updated after a few building steps. Then, simulations are executed using Cygwin and Mission Planner to view current flight trajectories of UAVs such as quadcopters and planes.

QT Creator views and edits the Ardupilot code. Several control functions exist within the code, and after changing several parameters with gedit, doxygen creates numerous dependency and call graphs for the code's classes and functions. Using this approach involves studying the code's functions for the motors to determine the control laws and schemes' outputs.

Software in the Loop. The Software in the Loop (SITL) simulations for quadcopters worked well, so none of the quadcopters crashed. However, one of the simulations for planes crashed when the motor disarmed prematurely to test if the plane could land safely at an altitude of approximately 30 meters or 98 feet. The plane hit the ground very roughly and had velocity variance errors. Shortly after, it flipped over twice and also had position variance errors. A few seconds later, the plane was at an altitude of approximately -150 meters or 492 feet, implying that the plane hit the ground so hard that it dug a hole and flew below the surface. This created an airspeed error, which makes sense since there is no airflow underground. Unfortunately, the plane continued to dig deeper into the ground and ended with an altitude of approximately -440 meters or -1440 feet. Figures 1.4a, 1.4b, and 1.5 show screenshots of the simulation flights.

SITL in Mission Planner can be used to test new GNC functions before conducting flight tests. Once SITL tests look promising, the next step would involve hardware in the loop (HITL) tests with the UAV hardware and equipment used in a flight test. Finally, the last step verifies the SITL and HITL tests after successful simulations.

Ardupilot Control Code Analysis. Analyzing the Ardupilot functions by hand requires thorough attention to details. To reduce human errors, Doxygen analyzes the Ardupilot functions by generating function maps and call/dependency graphs for the Ardupilot code to determine the interdependent relationships among the several functions. Altering numerous parameters through gedit in the doxyfile accurately produces the graphs. Figure 1.6 shows one of the "cleaner" func-



(a) Quadcopter SITL

(b) Plane SITL

Figure 1.4: Mission Planner SITL

Figure 1.5: Plane SITL Underground

tions, control_autotune. Some functions are complex with many other functions and classes, so their maps become unfortunately quite messy (see Figure 1.7 for a zoomed out map of acro_run.cpp) and can be difficult to decipher. This initial analysis was done in Spring 2017, so the function maps and call/dependency graphs may be more organized today.

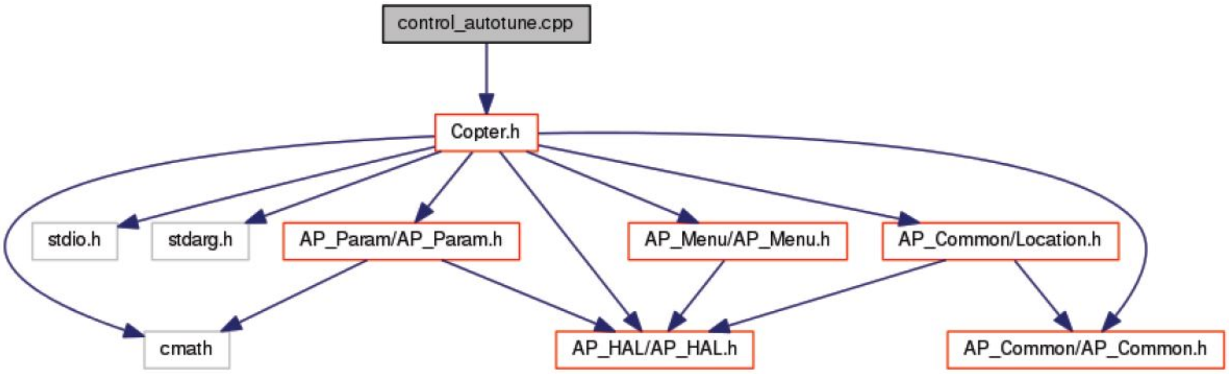


Figure 1.6: Doxygen Dependency Graph for control_autotune.cpp (Ardupilot)

Ardupilot's control_autotune function automatically determines PID coefficients during UAV flights. New GNC functions include using nonlinear control to replace PID control, so control_autotune is one of the control functions that would be replaced if Ardupilot is used to implement new GNC functions.

Ardupilot's acro_run is a very complicated function (see Figure 1.7). Therefore, integrating new GNC functions to replace acro_run can be extremely difficult to implement and debug. Ardupilot and PX4 have recently updated their code (this function map is outdated). If their software is used for GNC implementation, doxygen analysis would be redone to determine if the control functions' interdependent relationships are "cleaner" and easier to read.

INAV, Cleanflight, & Betaflight. Due to ArduPilot's complicated function calls and dependency graphs (as of Spring 2017) and the fact that the Pixhawk has an update rate of only 400 Hz [97], INAV was chosen later in Fall 2017 as the UAV project to work on instead of ArduPilot. INAV applies to racing drones, and some of these UAVs have flight controllers with update rates as high as 32,000 Hz, which is orders of magnitude faster than the Pixhawk's 400 Hz [11]. Having a fast update rate is essential for quick calculations and especially for onboard sensor processing in real-time to increase accuracy and performance during flight.

When new nonlinear control schemes get implemented, flight test data with PID control and flight test data with non-PID control from Betaflight Blackbox will be compared against each other to determine similarities and differences in stability and settling time to stabilize after violent maneuvers such as a roll. Overall, the Betaflight Blackbox is useful for evaluating PID tuning and reading sensor measurements.

INAV Control Code Analysis. INAV, Cleanflight, and Betaflight do not have doxyfiles like

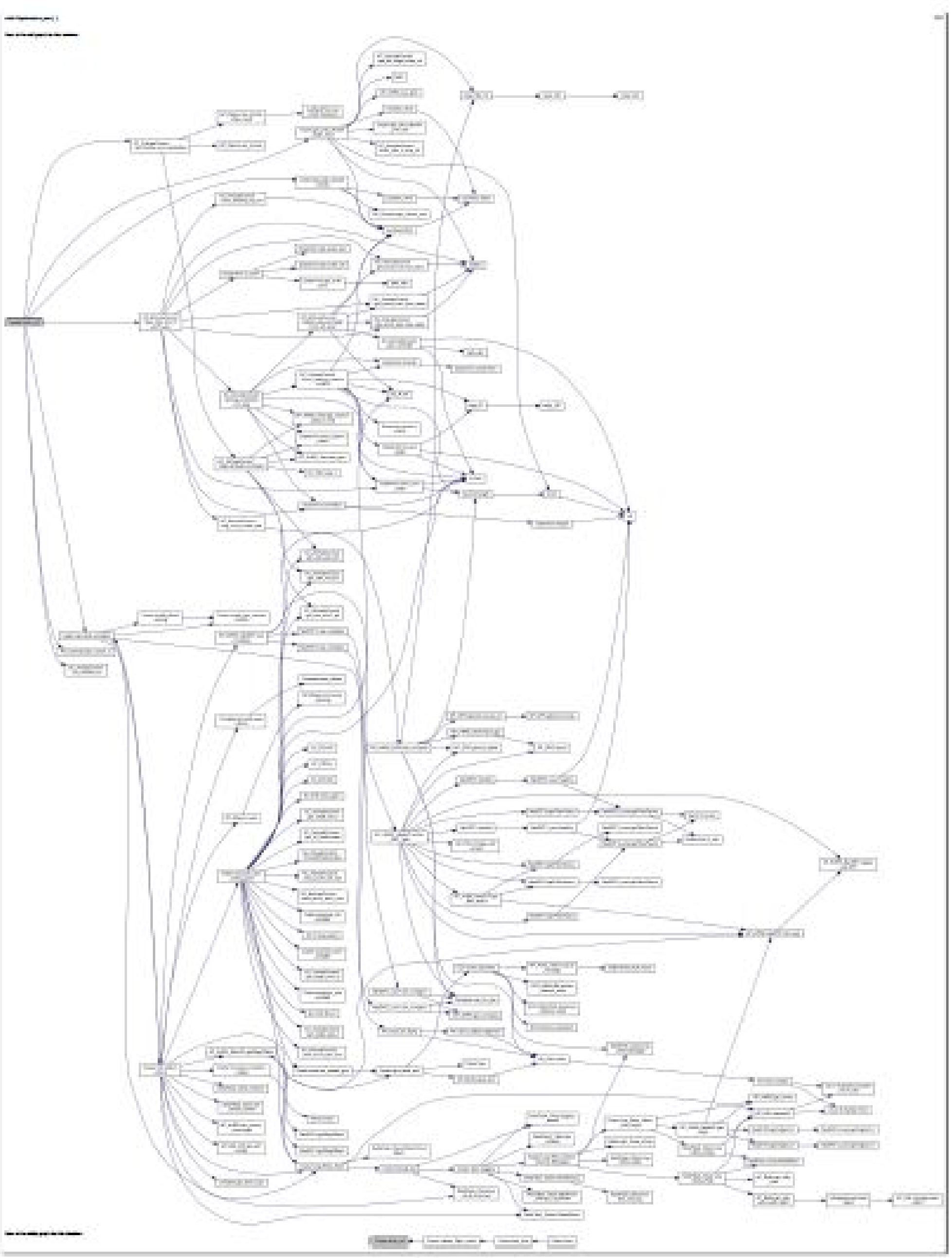


Figure 1.7: Function map for acro_run.cpp (Ardupilot)

Ardupilot, so Doxygen cannot electronically generate function maps to show the interdependent relationships. Instead of analyzing INAV functions like the handwritten approach used in the initial Ardupilot analysis, function dictionaries and math formulas provide a written analysis instead of a graphical analysis. There are numerous functions in INAV's source code, but the two main functions closely studied are pid.c and navigation.c. From this brief INAV code analysis, the author drew an overall block diagram for INAV's architecture (see Figure 1.8).

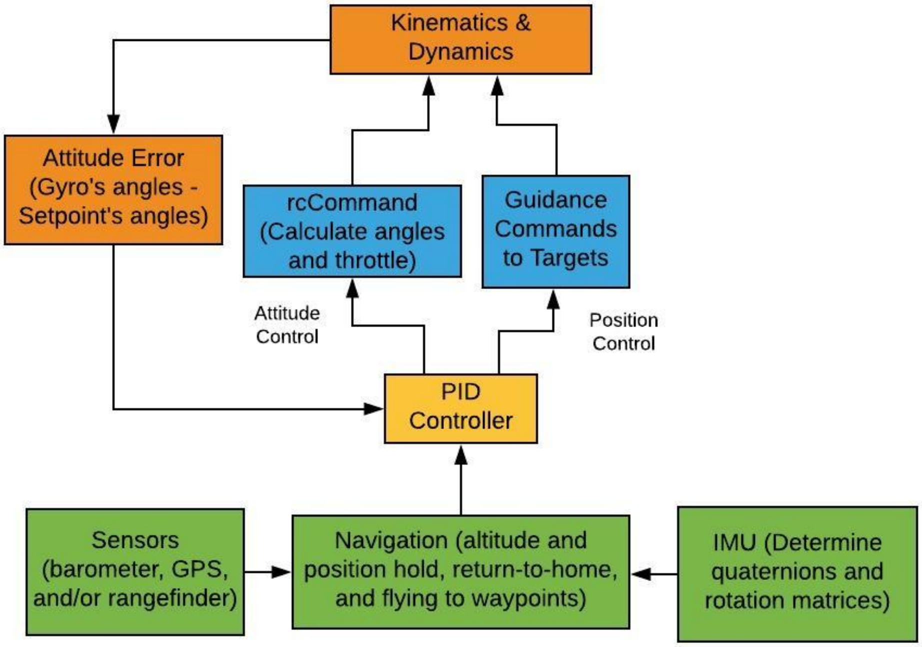


Figure 1.8: INAV Block Diagram

DJI SDK. There are five different types of DJI SDK (software development kit): mobile SDK, UX SDK, onboard SDK, payload SDK, and Windows SDK. Onboard SDK (OSDK) allows the user to create customized functions using the DJI SDK libraries and interface directly with the DJI flight controller, a desirable feature for TGNC implementation [98].

1.5 Core Contributions

This research of integrating TGNC functions has some core contributions. First, extending E Guidance to rotational maneuvers demonstrates attitude guidance. Second, E Guidance extensions with higher order integration of three functions have some impractical results. However, this approach can help avoid incoming obstacles and is advantageous if satisfying the velocity terminal conditions is not required. Third, translational E Guidance satisfies the boundary conditions for the optimal control problem, which leads to extremal control and an extremal trajectory. Fourth, utilizing HME with EKFs and acoustic parameters shows the impact of considering several different models and parameters to have accurate state estimation. Simulation results have near real-time capabilities, so onboard implementation is feasible.

1.6 Overview of Chapters and Appendices

Chapter 2 provides the quadcopter dynamical model equations. Chapter 3 describes the problem and methodology. Chapter 4 describes the guidance methods and results. Chapter 5 provides the details and approach of solving the optimal control problem using the Hamiltonian formalism. Chapter 6 provides the results of integrating extremal control and explicit guidance. Chapter 7 describes the navigation solution and results from the HME framework. Chapter 8 combines the TGNC functions together for the waypoint guidance maneuver, which serves as an illustrative example of an integrated TGNC system for UAVs. Chapter 9 expresses the author's recommendations to the community and lists novelties and research contributions to the field. Chapter 10 presents concluding remarks about main conclusions and future studies. Appendix A provides tables for the combinations of functions to extend E Guidance. Appendix B shows the closed-form solutions for the partials of the velocity. Appendix C describes the author's communications with the community such as publications, presentations, and internships. Appendix D discusses the author's educational and extracurricular activities.

CHAPTER 2
QUADCOPTER DYNAMICAL MODEL

2.1 State Vector

Let the general state vector for quadcopters be $\boldsymbol{s} \in \mathbb{R}^n$ where $n = 12$. The state variables are position vector components, velocity vector components, quaternions, and angular velocity components:

$$\boldsymbol{s} = \begin{bmatrix} p_N & p_E & p_U & v_N & v_E & v_U & q_0 & q_1 & q_2 & \omega_x & \omega_y & \omega_z \end{bmatrix}^T , \tag{2.1}$$

The state vector $\boldsymbol{s}$ can be decomposed into four vector functions:

$$\boldsymbol{s} = \begin{bmatrix} \boldsymbol{p} & \boldsymbol{v} & \boldsymbol{q} & \boldsymbol{\Omega} \end{bmatrix} , \tag{2.2}$$

where each of them are defined as

$$\boldsymbol{p} = \begin{bmatrix} p_N & p_E & p_U \end{bmatrix} , \boldsymbol{v} = \begin{bmatrix} v_N & v_E & v_U \end{bmatrix} , \boldsymbol{q} = \begin{bmatrix} q_0 & q_1 & q_2 \end{bmatrix} , \boldsymbol{\Omega} = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix} . \tag{2.3}$$

2.2 Coordinate Systems

The NEU inertial frame is fixed and set on the ground. This NEU frame resembles the SEZ (topocentric horizon coordinate system) frame, where the UAV acts as the satellite and assuming the earth is flat [99]. Gravity points in the negative U-direction. The body frame is on the quadcopter where the X and Y axes align with the arms, assuming a symmetrical frame, and the Z-axis points up in the same direction as the motor axes. Fig. 2.1 shows the inertial frame axes denoted by N, E, U and the body frame axes denoted by X, Y, Z [100]. A rotation matrix following the (3-2-1) sequence is applied to rotate the quadcopter from the body frame to the inertial frame: [101]

$$\mathbf{R} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} . \tag{2.4}$$

2.3 Forces

Fig. 2.1 also shows the thrust, drag, and weight forces acting on the quadcopter and are indicated by $\boldsymbol{T}_b, \boldsymbol{D}, \boldsymbol{W}$ respectively. The drag force opposes the velocity vector indicated by $\boldsymbol{v}$.

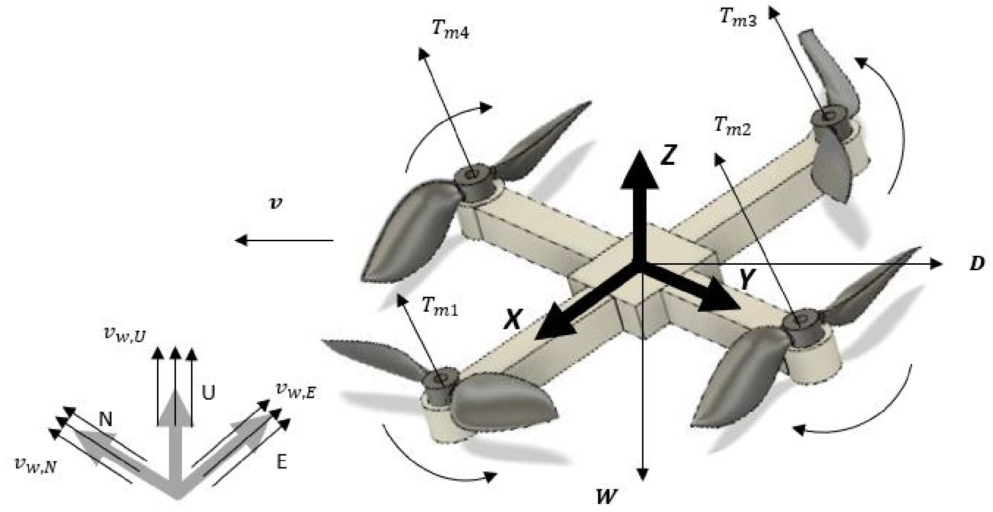


Figure 2.1: Body and Inertial Frames of a Quadcopter with Forces

Thrust. The motor thrust vector in the body frame is

$$\boldsymbol{T}_b = \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} T_{mi} \end{bmatrix}, \tag{2.5}$$

where T_{mi} is the thrust from the ith motor.

Static Thrust Model. The static motor thrust vector in the body frame is

$$\boldsymbol{T}_b = \sum_{i=1}^{4} T_{mi} = k \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} \omega_i^2 \end{bmatrix}, \tag{2.6}$$

where ω_i^2 is the squared angular rate of the ith motor and k is the motor thrust coefficient [100].

Dynamic Thrust Models. Most quadcopters typically move throughout their flights and do not stay static for long periods of time. Therefore, a dynamic thrust model should give more accurate state estimation results than a static thrust model.

Each motor contributes towards the overall thrust and uses either the Burgers (T_B) thrust model or the Staples (T_S) thrust model with T_{mi} being the Burgers thrust model or the Staples thrust model:

$$T_{mi} = \begin{cases} T_B \\ T_S \end{cases}. \tag{2.7}$$

Burgers Thrust Model. The Burgers thrust model depends on the airspeed (v_∞), propeller blade tip velocity (v_{tip}), the density of the fluid (ρ), normalized thrust ratio of available work to kinetic energy (η_L), and the area swept out by the propeller blade (S_b): [102]

$$T_B = \left(\frac{1}{2}v_\infty^2 + \frac{1}{6}v_{tip}^2\right)\rho\eta_L S_b. \tag{2.8}$$

The velocity at the propeller's tip is:

$$v_{tip} = \pi N d\ , \tag{2.9}$$

in which N is the number of revolutions per second and d is the diameter of the propeller [103].

Staples Thrust Model. The Staples thrust model utilizes Newton's second law and mass flow rate with the assumption that the velocity at the tip of the propeller (v_{tip}) is approximately equivalent to the velocity of the fluid exiting the propeller (v_e) [104]. The Staples model also depends on the density of the fluid (ρ), the area swept out by the propeller (S_b), and airspeed (v_∞). Other works have also referenced this model [105, 106, 107, 108, 109].

$$T_S = \rho S_b(v_e^2 - v_e v_\infty)\left(k_1\frac{d}{\theta_p}\right)^{k_2}\ , \tag{2.10}$$

where k_1 is the coefficient constant, and k_2 is the power constant. Assuming the exit velocity and propeller tip velocity are equivalent to each other gives exit velocity as:

$$v_e = m_{RPM} \cdot 0.0254\frac{m}{in} \cdot \theta_p\ \frac{1\,\mathrm{min}}{60\ sec}, \tag{2.11}$$

in which θ_p is the pitch of the propeller blade measured in inches and defined as how far the propeller translates forward after one revolution, and m_{RPM} is the propeller angular velocity measured in revolutions per minute. The conversion factors show that the exit velocity has units of m/s. Both models use S_b, the area swept out by the propeller, and is $\frac{\pi d^2}{4}$ with d as the propeller diameter.

Drag.

Linear Drag. A simple, linear (subscript lin) drag model for quadcopters is:

$$\boldsymbol{D}_{lin} = \begin{bmatrix} -k_d v_N \\ -k_d v_E \\ -k_d v_U \end{bmatrix}\ , \tag{2.12}$$

where k_d is the constant friction coefficient, which may or may not be unique.

Nonlinear Drag. The aerodynamic drag force in the body frame is: [110]

$$\boldsymbol{D} = \begin{bmatrix} D_x \\ D_y \\ D_z \end{bmatrix} = \frac{1}{2}\rho C_d A \begin{bmatrix} v_{\infty,x}^2 \\ v_{\infty,y}^2 \\ v_{\infty,z}^2 \end{bmatrix} , \tag{2.13}$$

with ρ as the density of the air, C_d as the constant drag coefficient to be determined or estimated, A as the cross-sectional frontal area of the quadcopter, and $v_{\infty,x}$, $v_{\infty,y}$, and $v_{\infty,z}$ as the airspeeds of the quadcopter in the body axes [110]. The International Standard Atmosphere (ISA) density model increases exponentially with altitude [111]. Using the standard air density, $\rho_0 = 1.225\ kg/m^3$, and an exponential curve fit of the ISA density model yields a simple form with h as the altitude:

$$\rho(h) = \rho_0 e^{-9.611 \cdot 10^{-5} h} , \tag{2.14}$$

Another exponential density model neglects gravity variations and assumes an isothermal atmosphere, which resembles Eq. (2.14) [112].

Weight. Weight is defined as:

$$\boldsymbol{W} = \begin{bmatrix} 0\ 0\ mg \end{bmatrix}^T , \tag{2.15}$$

where m is the mass and g is the acceleration due to gravity.

2.4 Moment of Inertia Matrix

The inertia matrix is modeled as two thin uniform rods intersecting at the origin with point masses at the ends to represent the motors and a rectangular prism to model the central chassis with the center of mass shifted slightly down the negative Z axis. Principle axes of moment of inertia are assumed to be aligned with the body axes, which gives an inertia matrix of:

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} . \tag{2.16}$$

2.5 Torques

Summing the torques gives the typical equation:

$$\boldsymbol{\tau} = I\dot{\boldsymbol{\Omega}} + \boldsymbol{\Omega} \times (I\boldsymbol{\Omega}) , \tag{2.17}$$

Expanding Eq. (2.17) with the inertia matrix components gives:

$$\boldsymbol{\tau} = \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} I_{xx}\alpha_x - I_{yy}\omega_y\omega_z + I_{zz}\omega_y\omega_z \\ I_{yy}\alpha_y + I_{xx}\omega_x\omega_z - I_{zz}\omega_x\omega_z \\ I_{zz}\alpha_z - I_{xx}\omega_x\omega_y + I_{yy}\omega_x\omega_y \end{bmatrix} .$$

A simple torque model about the z-axis assumes constant velocity and negligible angular acceleration:

$$\tau_z = (-1)^{i+1} b_\tau \omega_i^2 \tag{2.18}$$

with $(-1)^{i+1}$ positive for clockwise propellers and negative for counterclockwise propellers. Thus, the total torque about the z axis from all four propellers of a quadcopter is

$$\tau_\psi = b_\tau(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) ,$$

where b_τ is the torque drag coefficient.

Static Thrust Torque Model. If motors 1 and 3 are used for the pitch axis and motors 2 and 4 are used for the roll axis, then the roll and pitch torques are:

$$\tau_\theta = \ell k(\omega_1^2 - \omega_3^2) , \; \tau_\phi = \ell k(\omega_2^2 - \omega_4^2) , \tag{2.19}$$

where ℓ is the distance from the quadcopter to the center of a propeller (assuming symmetry) and k is the motor thrust coefficient. Combining the torques in the body frame results in a simple model:

$$\boldsymbol{\tau}_b = \begin{bmatrix} \ell k(\omega_2^2 - \omega_4^2) \\ \ell k(\omega_1^2 - \omega_3^2) \\ b_\tau(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} . \tag{2.20}$$

Dynamic Thrust Torque Model. The torque models for dynamic thrust use a different formulation for τ_ϕ and τ_θ, but τ_ψ is the same. The torques are defined in the body frame's X, Y, Z axes (see Fig. 2.1), and modifying a damping system of a circular disk immersed in oil yields a damping torque for quadcopters: [110]

$$\boldsymbol{\tau}_b = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} \ell(T_{m2} - T_{m4}) \\ \ell(T_{m1} - T_{m3}) \\ b_\tau(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} - \frac{1}{2}\pi\mu\ell^3 \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} , \tag{2.21}$$

with μ as the dynamic viscosity of air at 1 atm and 25° C. Same as before, the even numbered motors are on the roll axis, while the odd numbered motors are on the pitch axis.

2.6 Dynamical Model Equations

The velocity vector is defined as the derivative of the position vector as shown in Eq. (2.3) and is the same for the static thrust and dynamic thrust models.

Dynamical Model Equations with Static Thrust. Summing the forces in the inertial frame gives

$$m\ddot{\boldsymbol{p}} = -\boldsymbol{W} + \mathbf{R}\boldsymbol{T}_b + \boldsymbol{D}_{lin}\ , \tag{2.22}$$

where m is the mass, $\boldsymbol{p}$ is the position vector as defined in Eq. (2.3), g is the gravitational acceleration, $\boldsymbol{D}_{lin}$ is the linear drag force, R is the rotation matrix from the body frame to the inertial frame and $\boldsymbol{T}_b$ is the thrust vector in the body frame. The acceleration vector is:

$$\dot{\boldsymbol{v}} = \frac{1}{m}(-\boldsymbol{W} + \mathbf{R}\boldsymbol{T}_b + \boldsymbol{D}_{lin}). \tag{2.23}$$

The mass of the UAV is m, and the rotation matrix R from the inertial frame to the body frame uses the (3-2-1) sequence [101]. Combining Eq. (2.15) and the rotation matrix with the expansion of the thrust vector in the body frame from Eq. (2.6) and the drag force $\boldsymbol{D}$ from Eq. (2.13) turns Eq. (2.23) into:

$$\dot{\boldsymbol{v}} = \begin{bmatrix} \gamma(2q_0q_2 + 2q_1q_3) - \frac{k_d v_N}{m} \\ \gamma(2q_0q_1 - 2q_2q_3) - \frac{k_d v_E}{m} \\ -g + \gamma(q_0^2 - q_1^2 - q_2^2 + q_3^2) - \frac{k_d v_U}{m} \end{bmatrix}, \tag{2.24}$$

where

$$\gamma = \frac{k}{m}\left(\sum_{i=1}^{4}\omega_i^2\right). \tag{2.25}$$

The time derivatives of the Euler angle ($\dot{\boldsymbol{\Theta}}$) and angular velocity vectors are

$$\dot{\boldsymbol{\Theta}} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \cos\theta\sin\phi \\ 0 & -\sin\phi & \cos\theta\cos\phi \end{bmatrix}^{-1} \boldsymbol{\Omega}. \tag{2.26}$$

The Euler angles and quaternions are related by the (3-2-1) direction cosine matrix sequence and the quaternion identity, which forces the fourth quaternions to be a function of the other three quaternions to satisfy the identity:

$$\begin{aligned}
q_0 &= s\left(\frac{\phi}{2}\right)s\left(\frac{\theta}{2}\right)s\left(\frac{\psi}{2}\right)+c\left(\frac{\phi}{2}\right)c\left(\frac{\theta}{2}\right)c\left(\frac{\psi}{2}\right),\\
q_1 &= s\left(\frac{\phi}{2}\right)c\left(\frac{\theta}{2}\right)c\left(\frac{\psi}{2}\right)-c\left(\frac{\phi}{2}\right)s\left(\frac{\theta}{2}\right)s\left(\frac{\psi}{2}\right),\\
q_2 &= c\left(\frac{\phi}{2}\right)s\left(\frac{\theta}{2}\right)c\left(\frac{\psi}{2}\right)+s\left(\frac{\phi}{2}\right)c\left(\frac{\theta}{2}\right)s\left(\frac{\psi}{2}\right),\\
q_3 &= \pm\sqrt{1-q_0^2-q_1^2-q_2^2}\,,
\end{aligned} \tag{2.27}$$

where s denotes *sin* and c denotes *cos* [101]. The angular velocity vector and time derivative of quaternions are related by

$$\begin{bmatrix}\dot{q}_0\\ \dot{q}_1\\ \dot{q}_2\\ \dot{q}_3\end{bmatrix} = \frac{1}{2}\begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3\\ q_1 & q_0 & -q_3 & q_2\\ q_2 & q_3 & q_0 & -q_1\\ q_3 & -q_2 & q_1 & q_0\end{bmatrix}\begin{bmatrix}0\\ \omega_x\\ \omega_y\\ \omega_z\end{bmatrix}. \tag{2.28}$$

The angular acceleration of the static thrust model is

$$\dot{\mathbf{\Omega}} = \begin{bmatrix}\tau_\phi/I_{xx}\\ \tau_\theta/I_{yy}\\ \tau_\psi/I_{zz}\end{bmatrix} - \begin{bmatrix}\frac{I_{yy}-I_{zz}}{I_{xx}}\omega_y\omega_z\\ \frac{I_{zz}-I_{xx}}{I_{yy}}\omega_x\omega_z\\ \frac{I_{xx}-I_{yy}}{I_{zz}}\omega_x\omega_y\end{bmatrix},$$

where the torques, moment of inertia matrix components, and angular velocity components are determined from the equations of motion. In summary, the dynamical model equations with static thrust in their general vector form are

$$\dot{\boldsymbol{s}} = \begin{bmatrix}\dot{\boldsymbol{p}}\\ \dot{\boldsymbol{v}}\\ \dot{\boldsymbol{q}}\\ \dot{\mathbf{\Omega}}\end{bmatrix} = \begin{bmatrix} v_N\ v_E\ v_U\\ \frac{1}{m}(-\boldsymbol{W}+\mathbf{R}\boldsymbol{T}_b+\boldsymbol{D})\\ (1/2)\bar{\mathbf{Q}}\boldsymbol{\omega}\\ I^{-1}(\boldsymbol{\tau}-\boldsymbol{\omega}\times(I\boldsymbol{\omega}))\end{bmatrix}, \tag{2.29}$$

where $\bar{\mathbf{Q}}$ is the 4×4 matrix in (2.28). Overall, this quadcopter model is simplified and neglects complexities such as nonlinear quadcopter dynammics, rotational drag forces, propeller deformation caused by high velocities or flexible propeller material, and winds. Ref. [100] uses $k = 3\cdot 10^{-6}\ kg\cdot m, b_\tau = 1\cdot 10^{-7}\ kg\cdot m^2$, and $k_d = 0.25\ kg/s$.

Dynamical Model Equations with Dynamic Thrust. The quaternion equations are the same as the static thrust model and are not repeated here. The general, unexpanded acceleration vector without a specific dynamic thrust model ($\boldsymbol{T}$) is defined as:

$$\dot{\boldsymbol{v}} = \boldsymbol{a} = \frac{1}{m}\mathbf{R}(\boldsymbol{T} - \boldsymbol{D}) - \mathbf{g}\,, \tag{2.30}$$

Assuming a constant wind speed model in the NEU frame gives:

$$\boldsymbol{v}_w = \begin{bmatrix} v_{w,N} \\ v_{w,E} \\ v_{w,U} \end{bmatrix}, \tag{2.31}$$

and using Eqs. (2.3) and (2.31) gives the airspeed in the body frame as: [113]

$$\boldsymbol{v}_\infty = \begin{bmatrix} v_{\infty,x} \\ v_{\infty,y} \\ v_{\infty,z} \end{bmatrix} = \mathbf{R}^T(\boldsymbol{v} - \boldsymbol{v}_w)\,,$$

where $\mathbf{R}^T$ from Eq. (2.4) rotates the difference of the inertial velocity vectors, $\boldsymbol{v} - \boldsymbol{v}_w$, into the body frame. Using Burgers's thrust model, expanding terms, and assuming a constant wind speed model from Eq. (2.31) gives Eq. (2.30) as:

$$\dot{\boldsymbol{v}}_B = \begin{bmatrix} 2(q_0q_2 + q_1q_3)(N_x - \frac{D_x}{m}) \\ 2(q_2q_3 - q_0q_1)(N_y - \frac{D_y}{m}) \\ (q_0^2 - q_1^2 - q_2^2 + q_3^2)(N_z - \frac{D_z}{m}) - g \end{bmatrix}, \tag{2.32}$$

with N_x, N_y, N_z, n_1, and n_2 defined as:

$$N_x = n_1\left(n_2 + \frac{v_{\infty,x}^2}{2}\right),\ N_y = n_1\left(n_2 + \frac{v_{\infty,y}^2}{2}\right),\ N_z = n_1\left(n_2 + \frac{v_{\infty,z}^2}{2}\right),$$
$$n_1 = \frac{S_b\eta_L\rho}{m}\,,\ n_2 = \frac{N^2d^2\pi^2}{6}.$$

Using Staples's thrust model, expanding terms, and assuming the same constant wind speed model from Eq. (2.31) gives gives Eq. (2.30) as:

$$\dot{\boldsymbol{v}}_S = \begin{bmatrix} 2(q_0q_2 + q_1q_3)(M_x - \frac{D_x}{m}) \\ 2(q_2q_3 - q_0q_1)(M_y - \frac{D_y}{m}) \\ (q_0^2 - q_1^2 - q_2^2 + q_3^2)(M_z - \frac{D_z}{m}) - g \end{bmatrix}, \tag{2.33}$$

where $M_x, M_y, M_z, \Gamma, m_1, m_2,$ and m_3 are defined as:

$$M_x = m_1(m_2 - m_3 v_{\infty,x})\ ,\ M_y = m_1(m_2 - m_3 v_{\infty,y})\ ,\ M_z = m_1(m_2 - m_3 v_{\infty,z})\ ,$$
$$m_1 = \frac{\pi d^2 \rho}{m}\left(\frac{0.3034d}{\theta_p}\right)^{1.5}\ ,\ m_2 = \frac{16129 m_{RPM}^2 \theta_p^2}{9 \cdot 10^{10}}\ ,\ m_3 = \frac{127\theta_p}{300000}\ ,$$

where the numbers come from simplifying the conversion factors and using the default constants from Staples's model: $k_1 = 0.3034$ for the coefficient constant and $k_2 = 1.5$ for the power coefficient [104].

The angular acceleration of the dynamic thrust model is

$$\dot{\mathbf{\Omega}} = \begin{bmatrix} \tau_\phi / I_{xx} + \frac{\pi\mu\ell^3}{2I_{xx}} + \frac{I_{yy}-I_{zz}}{I_{xx}}\omega_y\omega_z \\ \tau_\theta / I_{yy} + \frac{\pi\mu\ell^3}{2I_{yy}} + \frac{I_{zz}-I_{xx}}{I_{yy}}\omega_x\omega_z \\ \tau_\psi / I_{zz} + \frac{\pi\mu\ell^3}{2I_{zz}} + \frac{I_{xx}-I_{yy}}{I_{zz}}\omega_x\omega_y \end{bmatrix}\ , \tag{2.34}$$

2.7 Control Vector

Let the control vector $\boldsymbol{u} \in \mathbb{R}^4$ have the motor angular velocities:

$$\boldsymbol{u} = [u_1\ u_2\ u_3\ u_4]\ , \tag{2.35}$$

where $u_i = \omega_i$ and $i = 1, 2, 3, 4$ based on the static thrust model from Eq. (2.6), the Burgers dynamic thrust model from Eqs. (2.8)-(2.9), or the Staples dynamic thrust model from Eqs. (2.10)-(2.11). The controls are admissible if they are defined and piece-wise continuous on some time interval $[t_0, t_1]$ and satisfy the constraints and conditions, which are mentioned in Chapter 5. To clarify, "motor angular velocity" refers to ω_i, while "motor spin rate" refers to ω_i^2 due to the thrust formulas.

CHAPTER 3
PROBLEM FOR INTEGRATING TARGETING, GUIDANCE, NAVIGATION, AND CONTROL FUNCTIONS

3.1 Description of Targeting, Guidance, Navigation, and Control Problems

Targeting. It is required for the targeting function to utilize sensor position data to locate a pursuer relative to a target. Targeting sensors can provide pursuer-to-target coordinates to determine the distance of the target from a pseudo-target or aim point, which is usually known a priori [114].

Guidance. It is required for the guidance function to determine a trajectory to reach a desired position. Other tasks include pre-calculating boost maneuvers based on time and duration, creating position, velocity, and attitude profiles to reach a desired point, and propagating the vehicle's center of mass in the body axes based on propellant consumption [115, 114].

Navigation. It is required for the navigation function to provide present information of the state of the vehicle (state estimation) with respect to a coordinate reference frame by processing several inputs and measurements from sensors, where Kalman filters are common navigation solutions for estimating the state vector [115, 114].

Optimal Control. This study considers using optimal control for UAVs, where it is required to find the state and control functions that satisfy Eq. (2.1), constraint equations, and extremizes a functional. The proposed research will use the Hamiltonian formalism with the indirect method, which was briefly mentioned in Chapter 1 [76].

3.2 Problem Statement Factors

This study combines targeting, guidance, navigation, and control functions to create a robust and accurate integrated TGNC system for UAVs for applications, which leverages autonomous capabilities with real-time target-relative guidance and re-targeting capabilities.

Description of Integrating TGNC on UAVs. GNC systems were first successful in aerospace problems such as Apollo, where Apollo utilized guidance for lunar-descent, Kalman filtering, and optimal control theory [116, 114]. Developing GNC systems were initially and heavily emphasized on the math instead of solving practical problems. Using relevant GNC functions and concepts to solve practical problems is essential to solving real-world problems [114]. Real-world problems for implementing GNC into UAV include problems such as sense-and-avoid obstacles autonomously, determining targets autonomously in real time, performing complicated maneuvers autonomously, or guidance to a location when communication is lost. Integrating TGNC functions to the UAV flight controller is a major element in leveraging autonomy and enabling UAVs with these types of autonomous capabilities [1]. Thus, the main issue is learning and implementing effective methods to build practical GNC systems for solving UAV problems autonomously in real-time such as obstacle avoidance given only a starting location and landing at some destination while responding to unexpected situations [1, 114].

Onboard Implementation of TGNC: Advantages and Disadvantages. Disadvantages of implementing TGNC include:

1. Difficult to understand and apply GNC functions to perform specific UAV tasks and maneuvers [114]
2. Gaps between theories and applications of GNC [114]

Advantages of implementing TGNC include:

1. Solving real-world practical problems autonomously such as real-time autonomous targeting, autonomous obstacle avoidance, and autonomous precision landing
2. In comparison to space shuttles or lunar landers, not much expensive hardware is needed for integrating GNC into UAV (thousands of dollars or less)
3. Diminishing GNC research efforts since Apollo era without much practical development since then [114]

Shortcomings and Progress in Current TGNC for UAVs. The main shortcomings in current UAV TGNC functions are:

1. Separated uses of novel, complicated GNC functions, i.e. no integration
    (a) Comparing a nonlinear guidance law with linear PD and PID controllers [12].
    (b) Adopting concepts of nonlinear guidance but continuing to use PD control [14, 15].
    (c) Extended Kalman Filters with linear PID controllers, lack of robust navigation functions, and using nonlinear adaptive control without focusing on navigation or guidance functions [20, 21, 23, 51].
2. Need to advance GNC to be considered as a discipline with potential targeting capability [114]
    (a) Aircraft GNC is traditionally considered as a secondary role.
    (b) GNC technology does not produce anything tangible.
3. Lack of understanding fundamentals of advanced GNC, which causes a dependency on aerodynamic design instead of advanced GNC design [114]
4. Using linear control for UAVs, which are nonlinear systems, so using linear PID controllers for nonlinear dynamic systems creates errors [40, 41, 42, 43, 44, 45].
5. EKFs are typically used for navigation in UAVs, but there exist other accurate estimation techniques such as Hierarchical Mixture of Experts (HME) [1, 20, 21, 23, 31]

6. Underutilization of guidance laws, schemes, and their integration with navigation and control functions

3.3 Proposed Approach to the Integration of Targeting, Guidance, Navigation, and Control Functions

1. Implement guidance laws and schemes including explicit guidance (E guidance) [117]
2. Replace PID control with nonlinear control laws using optimal control to find extremal control solutions
3. Utilize multiple EKFs for estimating the state vector to form a HME framework
4. Design safe trajectories to avoid static and dynamic obstacles (if applicable)

The proposed solution intends to consider increasing the accuracy of UAV GNC performance by modeling the UAV as a nonlinear system instead of linearizing it.

3.3.1 Explicit guidance (E guidance)

In 1964, George Cherry derived a general explicit, optimizing guidance law for rocket-propelled spacecraft. Explicit guidance laws express the formulas for steering commands directly in terms of the current and desired boundary values of the position and velocity vectors' components. Guidance laws are explicit only when they are derived as direct solutions to the equations of motion. Sometimes explicit guidance laws cannot be exact, so derive explicit equations that become more accurate as the current and desired boundary conditions approach each other closer and closer [117].

The thrust acceleration, a_{Tx}, is the control function and moves the vehicle from the initial conditions to the desired/terminal conditions. It is usually convenient to determine $\ddot{x}$ first and then a_{Tx}. Overall, it is required to determine the commanded thrust acceleration:

$$a_{Tx}(t) = \ddot{x} - g_x \; , \; a_{Ty}(t) = \ddot{y} - g_y \; , \; a_{Tz}(t) = \ddot{z} - g_z \; , \tag{3.1}$$

in which:

$$\ddot{x} = c_1 p_1 + c_2 p_2 \; , \; \ddot{y} = c_3 p_3 + c_4 p_4 \; , \; \ddot{z} = c_5 p_5 + c_6 p_6 \tag{3.2}$$

with $c_i \in \mathbb{R}$ and $p_i = p_i(t)$ are linearly independent, pre-specified functions of time.

Discussion and Comments about E Guidance.

1. Recompute the $\bar{\mathbf{E}}$ matrix to determine the coefficients periodically, but the coefficients will not change if the navigation system and flight control system perform perfectly, i.e. invariant constants under perfect conditions [117].

2. However, recompute if the information from the navigation system improves. For instance, landing radar data improves with decreasing altitude [117].
3. Recomputations should occur at a rate corresponding to the period over which the improving radar data makes a significant change in the constants [117].
4. Error predictions [117]
    - Note the difference of $\dot{x}_D - \dot{x}_0$, where this difference is essentially a prediction of the final error in $\dot{x}$ at $t = T$.
    - Also note that $x_D - x_0 - \dot{x}_0 T_{go}$ is the prediction of the final error in x.
    - Therefore, the $\bar{\mathbf{E}}$ matrix is essentially a final-value control scheme, so the general version is:
    $$\begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} e_{11} & e_{12} \\ e_{21} & e_{22} \end{bmatrix} \begin{bmatrix} predicted\ final\ speed\ error \\ predicted\ final\ displacement\ error \end{bmatrix} \tag{3.3}$$
    - The correction acceleration program in [117] is given by, in particular, for the x axis:
    $$\ddot{\boldsymbol{x}} = c_1 p_1(t) + c_2 p_2(t) \tag{3.4}$$
    Similar forms exist for the y and z axes.
5. The navigation system measures the state variables such as $p, \dot{p}, q, \dot{q}, ...$ [117].
6. The general solution to final-value control problems require maintaining specified values of speed and displacement of a general coordinate such as $\bar{q}$, where the differential equation of motion defined in [117] is
$$\frac{\mathrm{d}^2\bar{\boldsymbol{q}}}{\mathrm{d}t^2} = \boldsymbol{f}(p, \dot{p}, q, \dot{q}, ..., t) + \boldsymbol{a}_{Tq} \tag{3.5}$$
7. Thus, a_{Tq} is the control or choice variable that is chosen to force $q(t)$ and $\dot{q}(t)$ to the specified values at the terminal/desired time T [117].
8. Note that the $\bar{\mathbf{E}}$ matrix's elements approach ∞ as T_{go} approaches 0 or vanishingly small [117].
9. Unfortunately, when T_{go} approaches 0, the non-vanishing errors in the boundary conditions need an infinite acceleration to correct them, which causes c_1 and c_2 to blow up [117].
10. This singularity can be avoided by not recomputing the $\bar{\mathbf{E}}$ matrix, c_1, and c_2 during the last few seconds of powered flight, i.e. when the vehicle is close to its desired target position and velocity [117].
11. Ultimately, the crucial step in E guidance involves good choices for polynomials $p_1(t)$ and $p_2(t)$. One possibility is to choose a 2nd order polynomial for $p_1(t)$ and a 3rd order polynomial for and $p_2(t)$ to compare against Cherry's polynomials of $p_1(t) = 1$ and $p_2(t) = T - t$. Performance criteria can be based on estimation errors, which set of polynomials causes the UAV to follow the desired trajectory closer, and which set of polynomials produces an

acceleration command that steers the UAV towards the desired trajectory the quickest.

3.3.2 Replacing PID Control with Optimal Control

Typically, nonlinear UAVs use PID control, which is linear. It makes sense for nonlinear controllers to control nonlinear systems. Optimal control methods to find extremal control solutions can be used to formulate extremal control laws, which will replace PID control.

3.3.3 HME-EKF

Extended Kalman filters (EKFs) are used to provide state estimations of nonlinear systems [118]. The EKF algorithm is well-known and will not be discussed in detail here. Multiple EKFs can be combined into one bank to create a simplified Hierarchical Mixture of Experts (HME). Every expert in the HME framework is a Kalman filter or extended Kalman filter. Each bank contains an internal gating network, and a top-level gating network is outside the banks and weighs the bank outputs. Each filter has a different model of unknown parameters such as process and measurement noise.

The bank of filters uses two search algorithms in a feedback loop. One algorithm operates in real-time and utilizes a recursive quadratic programming approach to find extrema of a modified maximum likelihood function. The second algorithm works better for post-processing and has a genetic algorithm that searches for the parameter vector [1, 31]. HME has been used in applications such as interplanetary orbit determination problems for the Mars mission [32]. Figure 3.1 shows the general architecture for HME.

3.4 Dedicated Quadcopter Platforms for TGNC

3.4.1 DJI M100

To avoid ruining the default flight control software, the DJI Onboard Software Development Kit (OSDK) allows users to write code using OSDK functions without affecting the default underlying code. This increases productivity because only the user's code will have problems and errors, which makes it easier to debug and troubleshoot code.

The DJI M100 is the selected quadcopter for the proposed solution since it comes with OSDK capabilities. Fig. 3.2a shows the DJI M100, which has arms of length $\ell = 0.31\ m$ and a mass of $m = 3.133\ kg$ with one TB48D 6S (5700 mAhr) battery, a Raspberry Pi 2 Model B, and a 3D printed case for the Raspberry Pi. The linear drag coefficients are $k_{dx} = 0.5, k_{dy} = 1.75$, and $k_{dz} = 135\ kg/s$, the drag coefficient from the standard parabolic drag model is 0.5576, the motor thrust coefficient is $k = 3.19 \cdot 10^{-5}\ kg \cdot m$, and the drag torque coefficient is $b_\tau = 1 \cdot 10^{-7}\ kg \cdot m^2$, which were determined by algebraically solving for these parameters from the telemetry data after

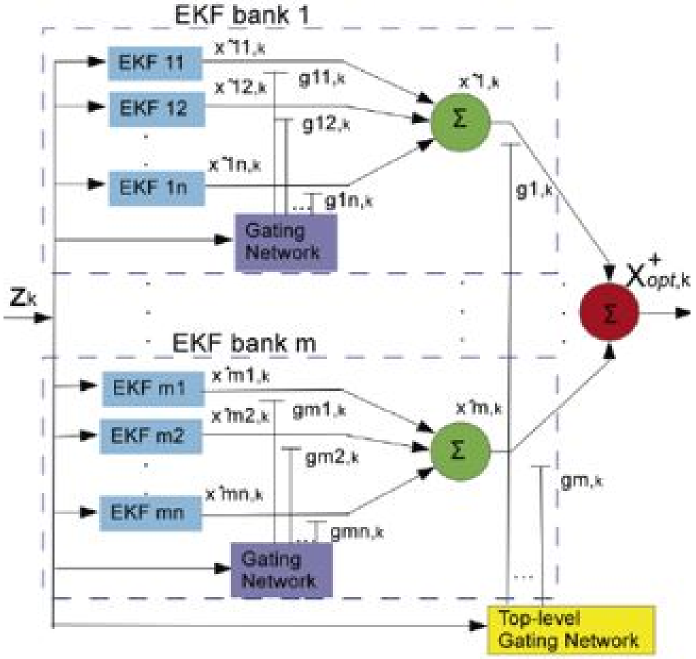


Figure 3.1: General HME Architecture

conducting flight tests. The moment of inertia matrix components are $I_{xx} = 0.02$, $I_{yy} = 0.02$, and $I_{zz} = 0.03\ kg \cdot m^2$ by the mass properties of the DJI M100 Solidworks assembly, which verifies that the principal axes of the inertia matrix align with the body axes. Measuring the frontal area of the DJI M100 in Solidworks gives $A = 0.0407\ m^2$. The propeller diameter is 0.343 m with a pitch of 4.5 in.

3.4.2 FliteTest 270 Chase Quad

Figure 3.2b shows a picture of the FliteTest 270 Chase Quad, which is a quadcopter for drone racing. It has arm lengths of 0.136 m, mass of 0.677 kg, a motor thrust coefficient of $3 \cdot 10^{-6}\ kg \cdot m$, with moment of inertia components: $I_{xx} = 5 \cdot 10^{-5}$, $I_{yy} = 5 \cdot 10^{-5}$, $I_{zz} = 10 \cdot 10^{-5}\ kg \cdot m^2$, torque drag coefficient of $1 \cdot 10^{-7}\ kg \cdot m^2$ and a linear drag coefficient of $k_d = 0.25\ kg/s$.

(a) DJI M100 with Raspberry Pi 2 Model B (b) FliteTest 270 Chase Quad

Figure 3.2: Exponential Braking Guidance: Distances from Waypoint

CHAPTER 4
PROPOSED GUIDANCE METHODS

4.1 Explicit and Analytical Guidance

Current guidance results are applied to a quadcopter roll maneuver and waypoint guidance maneuver. These change of variables provide a compact form throughout this chapter: $\tau = T - t$ and $T_{go} = T - t_0$.

4.1.1 Translational Guidance

For translational E guidance, the standard commanded acceleration is used in Eq. (3.1) with this choice of linearly independent polynomials:

$$p_1 = \tau^2 \ , \ p_2 = \tau^3 .$$

This yields an $\bar{\mathbf{F}}$ matrix of:

$$\bar{\mathbf{F}} = \begin{bmatrix} \frac{1}{3}T_{go}^3 & \frac{1}{4}T_{go}^4 \\ \frac{1}{4}T_{go}^4 & \frac{1}{5}T_{go}^5 \end{bmatrix} ,$$

and this yields an $\bar{\mathbf{E}}$ matrix of:

$$\bar{\mathbf{E}} = \begin{bmatrix} 48/T_{go}^3 & -60/T_{go}^4 \\ -60/T_{go}^4 & 80/T_{go}^5 \end{bmatrix} .$$

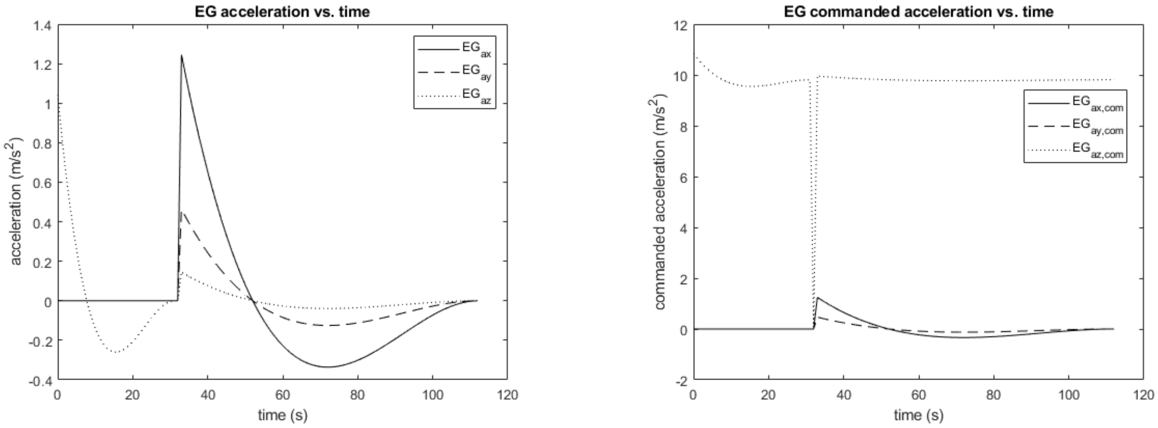
For the waypoint guidance maneuver, there are three points in $\mathbf{R}^3$:

1. Origin (0,0,0)
2. Takeoff altitude (0,0,50)
3. Final point of the maneuver (430,160,100) ,

and the waypoint guidance maneuver hits each of these three points. This gives two sets of coefficients, one set between points 1 (origin) and 2 (takeoff altitude) and the other set between points 2 (takeoff altitude) and 3 (waypoint guidance). The desired translational velocity at each point is (0,0,0), which yields a go-stop-go-stop sequence. The acceleration and commanded acceleration profiles are:

Directly integrating the acceleration instead of using the forward Euler method as in Ref. [119] gives the guided velocity and position profiles:

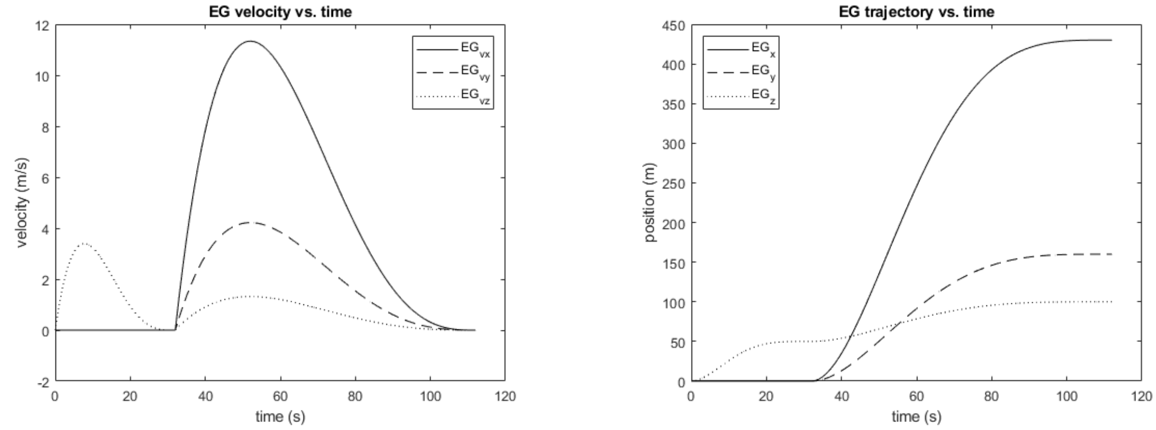
More details about the waypoint guidance maneuver will be covered in the next chapter.



(a) Acceleration

(b) Commanded Acceleration

Figure 4.1: Guided Waypoint Acceleration and Commanded Acceleration



(a) Guided Motion Velocity

(b) Guided Trajectory

Figure 4.2: Guided Waypoint Velocity and Trajectory

4.1.2 Rotational Guidance

For attitude E guidance, the linearly independent polynomials are:

$$p_1 = \tau^2\ ,\ p_2 = \tau^3.$$

Ref. [117] does not include attitude guidance, so explicit attitude guidance is derived to be:

$$\begin{aligned}
\alpha_\phi(t) &= c_1\tau^2 + c_2\tau^3 - \frac{I_{yy} - I_{zz}}{I_{xx}}\omega_y\omega_z\ , \\
\alpha_\theta(t) &= c_3\tau^2 + c_4\tau^3 - \frac{I_{zz} - I_{xx}}{I_{yy}}\omega_x\omega_z\ , \\
\alpha_\psi(t) &= c_5\tau^2 + c_6\tau^3 - \frac{I_{xx} - I_{yy}}{I_{zz}}\omega_x\omega_y.
\end{aligned} \tag{4.1}$$

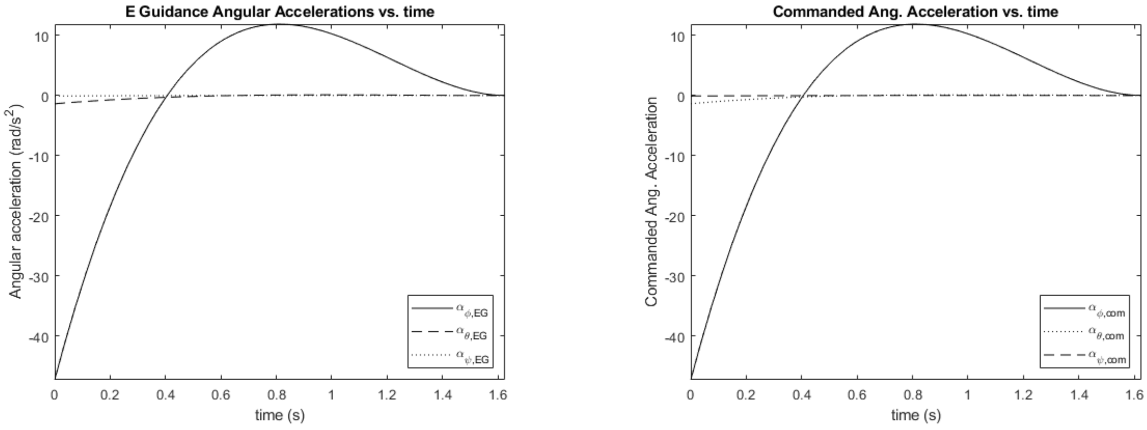
The choice of polynomials is the same as before, so the F and E matrices are the same as before. For the guided 360° roll maneuver, there are three points in $\mathbf{R}^3$:

1. Origin (0,0,0)
2. Takeoff altitude (0,0,100)
3. After the 360° roll (0,0,100) ,

and the roll maneuver hits each of these three points. Same as before, this gives two sets of coefficients, one set between points 1 (origin) and 2 (takeoff altitude) and the other set between points 2 (takeoff altitude) and 3 (final position). The desired angular and translational velocity at each point is (0,0,0), which yields a go-stop-go-stop sequence as before. In the guided roll maneuver, translational E guidance is applied between points 1 and 2 to takeoff to the desired altitude of 100 m, while attitude E guidance is applied between points 2 and 3 to generate the roll maneuver. The angular acceleration and commanded acceleration profiles are:

Directly integrating the angular acceleration instead of using the forward Euler method as in Ref. [120] gives the angular velocity and Euler angle profiles:

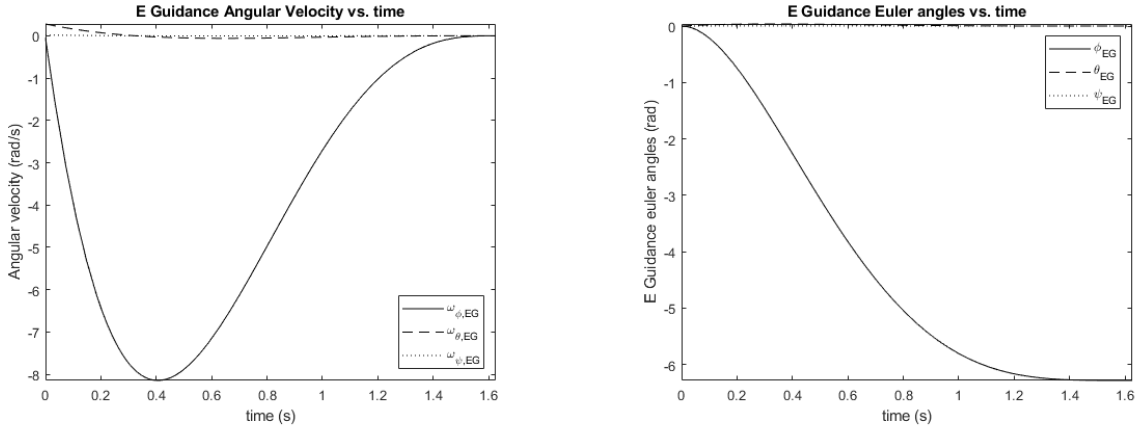
Converting the Euler angles into quaternions using the (3-2-1) sequence gives:



(a) Angular Acceleration

(b) Commanded Angular Acceleration

Figure 4.3: Guided Roll Maneuver Angular and Commanded Acceleration



(a) Guided Angular Velocity

(b) Guided Angles

Figure 4.4: Guided Roll Maneuver Angular Velocity and Angles

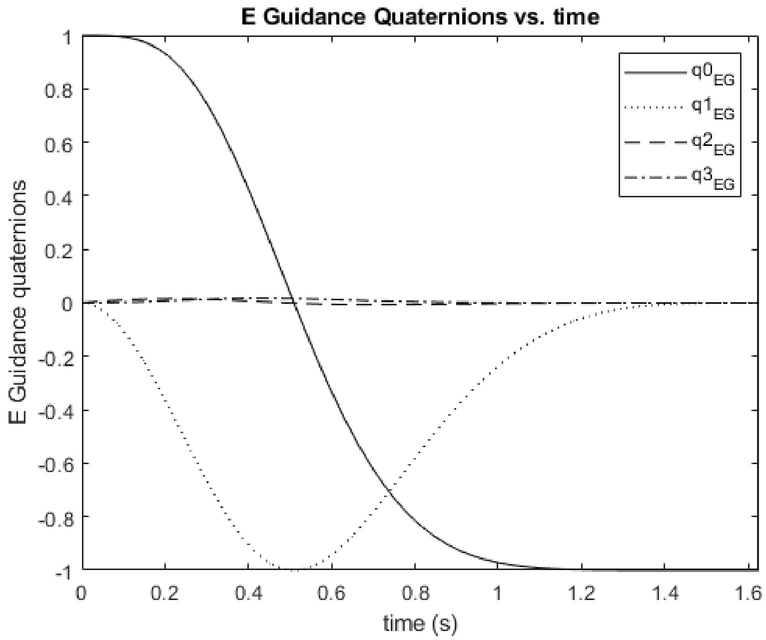


Figure 4.5: Quaternions of the Guided Roll Maneuer

More details about the roll maneuver will be discussed in the next chapter.

4.2 Proposed Extensions to E Guidance

The author of this paper considered extending E Guidance by adding more polynomials and constants. Thus, the $\bar{\mathbf{E}}$ matrix could be $\mathbf{R}^{3\times3}$ or $\mathbf{R}^{4\times4}$ instead of the original $\mathbf{R}^{2\times2}$. In the original E guidance document, final desired thrust acceleration is used for $E \in \mathbf{R}^{3\times3}$ to have neutral attitude at the end of the maneuver by aligning the velocity and acceleration vectors [117].

4.2.1 Intermediate Positions and Velocities

The first attempt used desired intermediate positions and velocities, so $T_{go,int} = T_{go}/2$. The intermediate desired positions are defined by dividing the final desired position in half and similarly for the intermediate velocities. The polynomials chosen are:

$$p_{i+1} = \tau^i ,$$

where $i = 0, ..., 3$. However, neither the intermediate nor final desired conditions are satisfied.

4.2.2 Desired Acceleration and Jerk

The second attempt uses desired acceleration and jerk for the third and fourth constraints without using the $\mathbf{\bar{E}}$ matrix, similar to what Cherry did for final desired attitude guidance by aligning the velocity and acceleration vectors. This attempt uses the same polynomials from Eq. (4.2.1). However, the acceleration, velocity, and position profiles are not even close to the desired values.

4.2.3 Higher Order Integration of E Guidance Functions

Integrating E Guidance Functions Four Times with Desired Final Acceleration and Jerk. This method uses the final desired acceleration as the third condition and the fourth condition as the final desired jerk. The final desired acceleration is satisfied, while the final desired potion, velocity, and jerk are not. Since this approach and the previous approach both used jerk and did not satisfy the target points, it might not be possible to incorporate jerk for computing the constraints, coefficients, or the $\mathbf{\bar{E}}$ matrix.

4.2.3.1 Integrating E Guidance Functions Three Times with Desired Final Acceleration

This method uses the final desired acceleration as the third condition, which is desired to be zero at the end of the maneuver for both translational and rotational motion.

Of the four methods, this method is the most promising but has some drawbacks for the roll, takeoff, and waypoint maneuvers. Overall, the boundary conditions are usually satisfied, but using this method with a 3×3 $\mathbf{\bar{E}}$ matrix may not be practical or realistic for applications due to moving in the opposite direction for some time before heading to the target point. Initial motion in the opposite direction resembles physical examples such as getting a head start, winding up a baseball pitch, or getting pulled back before receiving a big push on the swing set to move forward to a higher height on the opposite side. A potential application of this approach involves avoiding an incoming obstacle but with intention to still move forward to reach a target point behind the incoming obstacle. Starting the maneuver in the opposite direction from the target point would not be realistic for the takeoff maneuver because the UAV starts on the ground and cannot fly underground. It would also not be realistic if there were obstacles below or behind during mid-flight. Thus, the 2×2 $\mathbf{\bar{E}}$ matrix would be more practical to use because it lacks movement in the opposite direction from the target point.

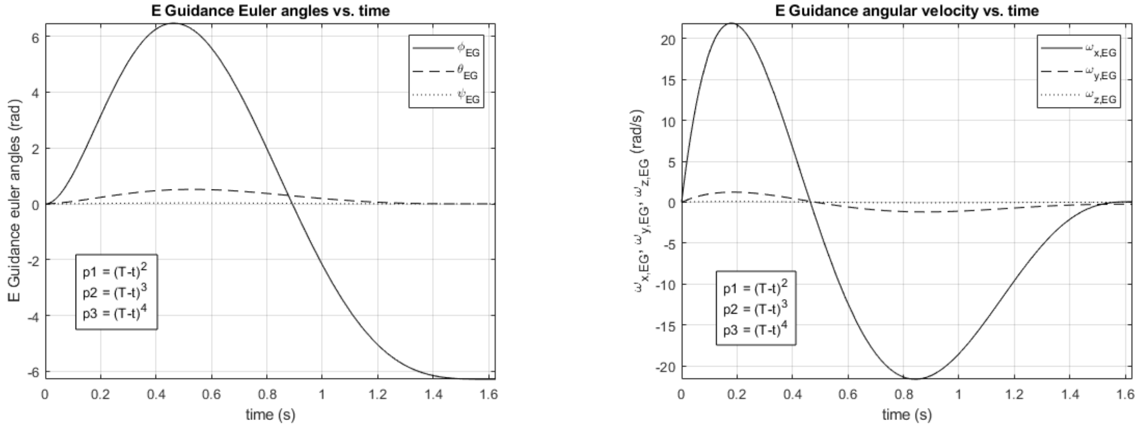
4.2.4 Proposed Extension of E Guidance to Maneuvers

4.2.4.1 Roll Maneuver E Guidance Results - Three Polynomials

For the 360° roll maneuver, the final desired roll angle is 360° or -6.2832 *rad* and a final roll velocity of 0 *rad/s*. The "best" polynomial choice that satisfies the desired final conditions is:

$$p_1(t) = \tau^2 \ , \ p_2(t) = \tau^3 \ , \ p_3(t) = \tau^4.$$

Figure 4.6a shows the Euler angles for the roll maneuver with polynomials from Eq. (4.2.4.1). Fig. 4.6b the final roll angle is -6.2831 *rad*. However, there is an initial roll in the opposite direction before heading towards the target point. This implies that the quadcopter would roll approximately 360° roll maneuver counterclockwise and then clockwise by about 720° to move back to the original orientation. Unfortunately, such a large roll in the opposite direction is impractical for a desired roll maneuver in only the clockwise direction, and this opposite motion away from the target point is a common phenomenon in most of the results. Numerous combinations with polynomials, exponential functions, and sinusoidal functions did not satisfy the boundary conditions and are not shown. Fig. 4.6b shows the angular velocity profile, and the final roll velocity of 0 *rad/s*. The torque required is 1.4114 *J*, which is less than the max torque of 24.56 *J* for the FliteTest 270 Chase Quad.



(a) Guided Angles

(b) Guided Angular Velocity

Figure 4.6: Guided Roll Maneuver Angles and Angular Velocity with Three Polynomials

4.2.4.2 Takeoff Maneuver E Guidance Results - Three Polynomials

The takeoff maneuver does not have practical results from this method because the UAV moves in the opposite direction before heading towards the desired altitude of 50 *m* above the ground. Some of the numerous combinations with polynomials, exponential functions, and sinusoidal functions satisfied the terminal conditions but all had initial movements in the opposite direction, i.e., underground, which is not physically possible.

4.2.4.3 Waypoint Maneuver E Guidance Results - Three Polynomials

The point of interest maneuver in Ref. [119] has the quadcopter fly to a waypoint at (430, 160, 100) [m] starting from an altitude of 50 m. Just like the roll manevuer and takeoff maneuver, there is an initial motion away from the target point before heading towards the target point. Thus, this method is somewhat practical assuming there are no obstacles behind and below the UAV. Moving backwards helps give a "bigger push" similar to someone initially moving someone backwards on a swing set before pushing them forward to generate more force to reach a higher height on the opposite side. However, none of the numerous combinations with polynomials, exponential functions, and sinusoidal functions satisfy all the terminal conditions. Some fulfilled reached the desired position but had non-zero velocity. Other combinations fulfilled the terminal position and velocity conditions but had non-zero acceleration at the end. Overall, this approach for the waypoint maneuver is impractical and unacceptable unless non-zero accelerations at the target point is acceptable. If so, then the "best" option is:

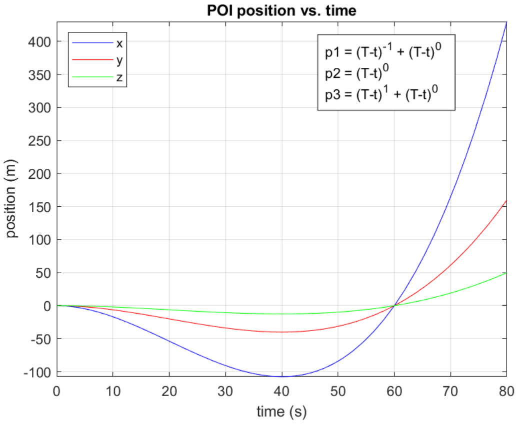
$$p_1(t) = \tau^{-1} + \tau^0 \ , \ p_2(t) = \tau^0 \ , \ p_3(t) = \tau + \tau^0.$$

Figure 4.7a shows the trajectory of the waypoint guidance maneuver in x, y, z Cartesian coordinates for this choice of functions. Once again, motion initially starts moving away from the target point, but forward motion towards the target point begins about halfway in the maneuver. Even though the final position coordinates are satisfied, motion in the x-direction away from the target goes back as far as approximately 220 m, which is roughly half of the distance to the target point in the x-direction.

Figure 4.7b shows the velocity profile and that there are larger velocities in the opposite direction from the target point. The speed in the x-direction goes backwards as high as approximately 10 m/s, which is about 22 mph. However, the terminal velocity conditions are satisfied.

Figure 4.8 shows the acceleration profile. The final acceleration values do not end at zero, which does not satisfy the desired acceleration conditions.

Appendix A shows all the combinations of the choices of $p_i(t)$. Combinations include exponential, trigonometric, and polynomials, and the best combinations were polynomials. Trigonometric and exponential functions tended to diverge, oscillate, and/or not satisfy the boundary conditions. Some combinations were so impractical that they did not produce real numbers and outputted NA, i.e., not a number.



(a) Guided Trajectory

(b) Guided Velocity

Figure 4.7: Guided Waypoint Maneuver Trajectory and Velocity: Three Polynomials

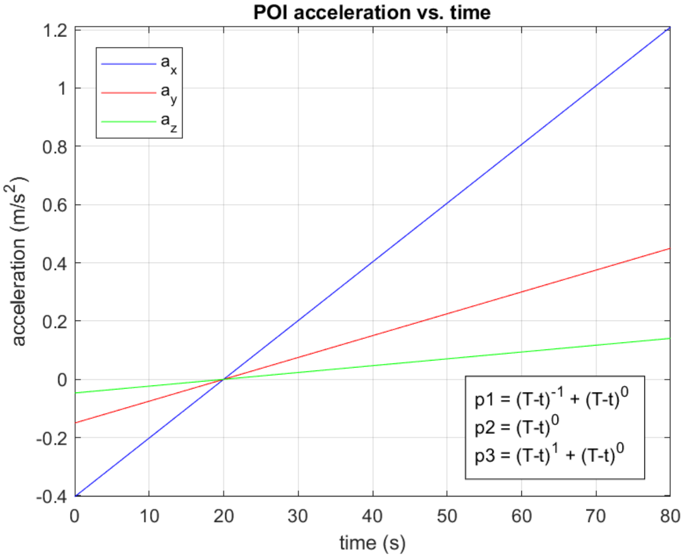


Figure 4.8: Guided Waypoint Maneuver Acceleration: Three Polynomials

4.2.5 Cherry's Formulation of E Guidance for Final Attitude

Cherry uses final desired attitude at the end of the maneuver by having the velocity and acceleration vectors aligned. He chooses:

$$p_1(t) = 1\,,\; p_2(t) = \tau\,,\; p_3(t) = \tau^2$$

with a non-symmetric $\bar{\mathbf{E}}$ matrix of:

$$E = \begin{bmatrix} 0 & 0 & 1 \\ 18/T_{go}^3 & -24/T_{go}^3 & -6/T_{go}^2 \\ -24/T_{go}^3 & 36/T_{go}^4 & 6/T_{go}^2 \end{bmatrix},$$

and a constraint vector of:

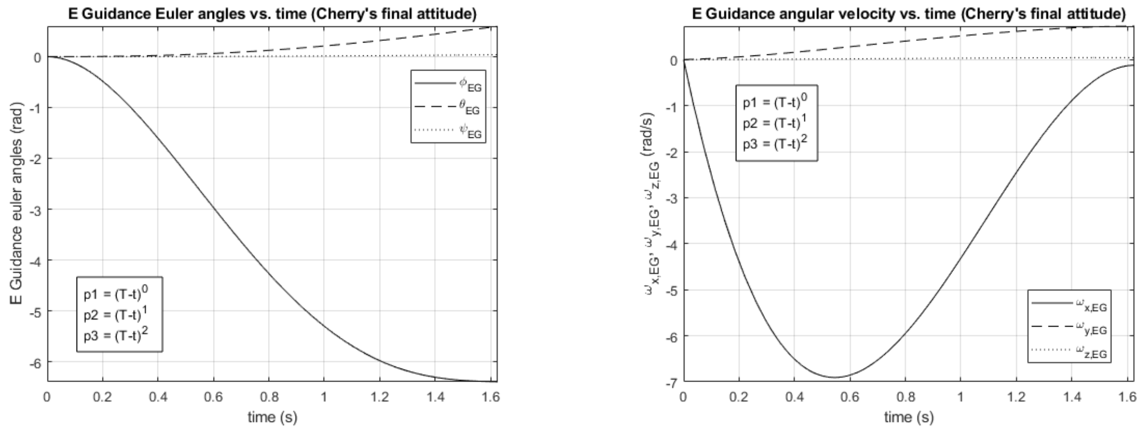
$$\begin{bmatrix} \dot{x}_D - \dot{x}_0 \\ x_D - x_0 - \dot{x}_0 T_{go} \\ a_{Tx,D} + g_x(T). \end{bmatrix}$$

Thus, he computes the coefficients by [117]:

$$\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 18/T_{go}^3 & -24/T_{go}^3 & -6/T_{go}^2 \\ -24/T_{go}^3 & 36/T_{go}^4 & 6/T_{go}^2 \end{bmatrix} \begin{bmatrix} \dot{x}_D - \dot{x}_0 \\ x_D - x_0 - \dot{x}_0 T_{go} \\ a_{Tx,D} + g_x(T) \end{bmatrix}.$$

Applying these polynomials, $\bar{\mathbf{E}}$ matrix, and constraints to the roll maneuver, takeoff maneuver, and waypoint guidance maneuver gives good results for only the takeoff maneuver. For the takeoff maneuver, the results are highly accurate because the takeoff maneuver is one-dimensional with the velocity and acceleration vectors aligned. The results are not shown because it does not have any novelties compared to Cherry's formulation of E guidance for final attitude. The waypoint maneuver does not have good results with this approach, so the results are not shown.

For the roll maneuver, Fig. 4.9a shows the Euler angle profile, while Fig. 4.9b shows the angular velocity profile. It is unusual that the pitch angle and pitch velocity do not converge to zero. Thus, the physical interpretation is that the quadcopter has to pitch slightly up to perform this 360° roll maneuver. An advantage of this method is that the backward motion is eliminated. However, the final desired conditions are not satisfied. The final roll angle is -6.3947 radians, which is about -366.4°, and the final roll velocity is $-0.1291\ rad/s$, which is about $-7.4^\circ/s$. Thus, the final desired roll angle and roll velocity conditions are nearly satisfied. The final roll angle error is about 2%, while the final roll velocity error is off by 0.1291 rad/s or $7.4^\circ/s$. Thus, Cherry's final attitude method nearly satisfies the boundary conditions for the roll maneuver but with some slight errors.



(a) Guided Angles

(b) Guided Angular Velocity

Figure 4.9: Roll Maneuver with Cherry's Formulation of E Guidance for Final Attitude

4.2.6 Summary and Conclusion on Extending E Guidance

The first method used intermediate position and velocity conditions with a sparse $\bar{\mathbf{E}}$ matrix and four functions: p_1, p_2, p_3, p_4, but this method did not satisfy the boundary conditions. The second method is similar to what Cherry did for desired attitude guidance by using final acceleration and jerk without using the $\bar{\mathbf{E}}$ matrix. However, the boundary conditions are not satisfied either. The third method used final acceleration and jerk with integrating up to four times. For this method, only the final acceleration is satisfied, but not the final desired position, velocity, and jerk conditions.

Based on the simulation results presented, extending E guidance to three or four functions does not enhance performance compared to p_1 and p_2. Intuitively, it was thought that increasing the size of the $\bar{\mathbf{E}}$ matrix with more functions and integrals would increase the performance similar to how one can gain accuracy by truncating less terms in a Taylor series expansion. However, further research and investigation may find a combination of functions for p_1, p_2, p_3 or p_1, p_2, p_3, p_4 that perform better than just using p_1 and p_2. There are an infinite number of possible functions to choose from if the method of multiple integrals is used, so it is possible that one could be luckier than the author and choose the right functions.

Alternatively, a different method instead of using multiple integrals may increase performance. Recall that Cherry did not use three integrals for final attitude guidance to align the velocity and acceleration vectors. Note that his method yielded an anti-symmetric $\bar{\mathbf{E}}$ matrix, so symmetric $\bar{\mathbf{E}}$ matrices may only be valid for $\bar{\mathbf{E}} \in \mathbf{R}^{2\times 2}$.

Overall, the last method works the best but with backwards motion. Unfortunately, it is currently unknown if the backwards motion can be eliminated for using three functions: p_1, p_2, p_3. The analysis for extending E guidance to three functions and three integrals shows that sticking to the original $\mathbf{R}^{2\times 2}$ matrix using p_1 and p_2 has better performance because it moves only forwards and satisfies all the boundary conditions.

There are some drawbacks to this E Guidance extension with three polynomials. First, the backward motion is generally impractical, especially for a takeoff maneuver because it would cause the aircraft to descend towards the ground, which would damage itself and anything or anyone nearby. Another drawback of this method is that the guidance performance is incomplete due to the final velocities being non-zero (see Fig. 4.7b). Finally, this method would definitely not be applicable to fixed-wing drones since they typically cannot "reverse" during flight, so this methods is useful only for multi-rotor UAVs.

There are some advantages to this E Guidance extension with three polynomials despite the backwards motion. First, this method may be practical for a particular case for a dynamic object approaching from the front, but the aircraft still desires to move forward. Thus, the aircraft would move backwards and out of the incoming object's path to avoid possible collisions. This is helpful because the trajectory to reach the target point is already determined after the quadcopter would move backwards to avoid the incoming object. However, this would be most successful for utilizing E guidance discretely instead of continuously to update the current state vector and have sensors to estimate the state of the incoming object. Second, if only the terminal positions need to be satisfied and the terminal velocities do not need to be satisfied, then this method could be applicable for maneuvers such as a multiple waypoint maneuver with an additional waypoint further ahead. However, the aircraft must ensure that there are no obstacles behind it before proceeding.

4.3 Proposed Exponential Braking to Reach Targets

Besides E guidance, an exponential braking law has the aircraft or spacecraft decelerate to reach a target point:

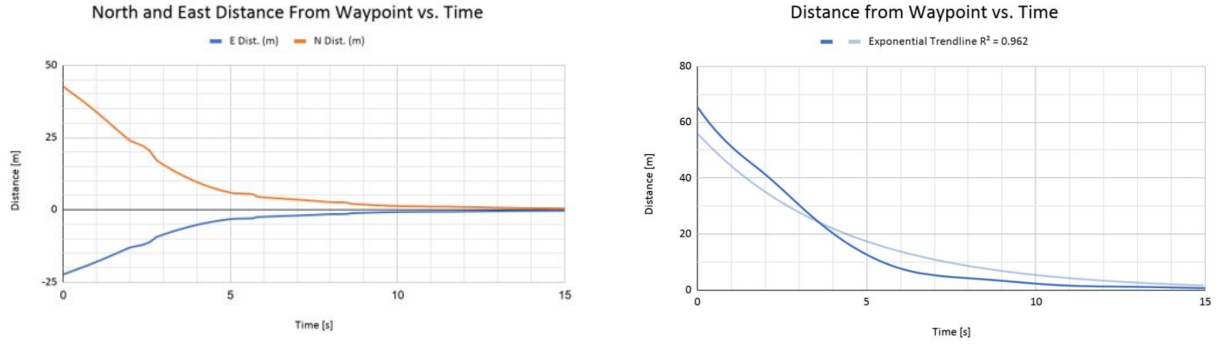
$$\boldsymbol{p}(t) = X_0 \cdot e^{-\frac{t-t_0}{\lambda}} \ , \ \dot{\boldsymbol{p}}(t) = \dot{X}_0 \cdot e^{-\frac{t-t_0}{\lambda}} \ ,$$

where λ is a tuning parameter or proportionality factor [115]. The goal involved guiding the quadcopter to a desired waypoint in 2D by exponentially braking after traveling at max speed. Preliminary testing in MATLAB demonstrates proof-of-concept before implementing onboard a Raspberry Pi on the DJI M100 using OSDK functions such as *positionAndYawCtrl*. Since the DJI OSDK *positionAndYawCtrl* requires feedback of the remaining distance to travel, the position commands slightly deviate from the original formulation in [115]:

$$p_{N,exp} = p_{N,dist} - p_{N_0,dist} \cdot exp\left(\frac{-\Delta t}{\beta_N}\right) \ , \ p_{E,exp} = p_{E,dist} - p_{E_0,dist} \cdot exp\left(\frac{-\Delta t}{\beta_E}\right) \ ,$$

where $p_{N,dist}$ and $p_{E,dist}$ denote the current distance in NE coordinates to the target point, $p_{N_0,dist}$ and $p_{E_0,dist}$ denote the original distance in NE coordinates from the target point, Δt denotes the elapsed time, and β_N and β_E are the tuning parameters or proportionality constants. Eq. (4.3) uses β instead of λ as in the original formulation to avoid confusion with Lagrange multipliers.

Figure 4.10 shows the hardware in the loop (HITL) tests of the exponential braking guidance law to reach a waypoint. As expected, Figure 4.10a and Figure 4.10b have exponential trajectories. An exponential curve fit to the distance from the waypoint reveals an accurate fit ($R^2 = 0.962$). The next step would be to obtain experimental results to verify the HITL simulation and compare against the default PID controller in a similar manner to Ref. [121, 122, 123].



(a) North and East Distance from Waypoint vs. Time

(b) Distance from Waypoint vs. Time

Figure 4.10: Exponential Braking Guidance: Distances from Waypoint

CHAPTER 5
OPTIMAL CONTROL PROBLEM WITH HAMILTONIAN FORMALISM

The optimal control problem and analysis follows the works of Ref. [119, 120, 121, 124].

5.1 Optimal Control Problem Statement

There are initial, intermediate, and final state constraint equations, each with four functions. The first function has the position constraints or desired coordinates in meters, the second function contains the velocity constraints in m/s, sthe third function constrains the Euler angles in radians, and the last function has constraints on the angular velocity in rad/s. The quadcopter must maintain an altitude below 400 ft throughout the entire flight [125].

Initial State Constraint Equations. There are four initial constraint equations, denoted by 0:

$$\begin{gathered} E_1(p_N, p_E, p_U) = (N_0, E_0, U_0)\ ,\ E_2(v_N, v_E, v_U) = (v_{N_0}, v_{E_0}, v_{U_0})\ , \\ E_3(\phi, \theta, \psi) = (\phi_0, \theta_0, \psi_0)\ ,\ E_4(\omega_x, \omega_y, \omega_z) = (\omega_{x_0}, \omega_{y_0}, \omega_{z_0}). \end{gathered} \tag{5.1}$$

Intermediate State Constraint Equations. The intermediate state constraint also has four equations and denoted by int:

$$\begin{gathered} F_1(p_N, p_E, p_U) = (N_{int}, E_{int}, U_{int})\ ,\ F_2(v_N, v_E, v_U) = (v_{N_{int}}, v_{E_{int}}, v_{U_{int}})\ , \\ F_3(\phi, \theta, \psi) = (\phi_{int}, \theta_{int}, \psi_{int})\ ,\ F_4(\omega_x, \omega_y, \omega_z) = (\omega_{x_{int}}, \omega_{y_{int}}, \omega_{z_{int}}). \end{gathered} \tag{5.2}$$

Final State Constraint Equations. There are four final constraint equations, denoted by f:

$$\begin{gathered} G_1(p_N, p_E, p_U) = (N_f, E_f, U_f)\ ,\ G_2(v_N, v_E, v_U) = (v_{N_f}, v_{E_f}, v_{U_f})\ , \\ G_3(\phi, \theta, \psi) = (\phi_f, \theta_f, \psi_f)\ ,\ G_4(\omega_x, \omega_y, \omega_z) = (\omega_{x_f}, \omega_{y_f}, \omega_{z_f}). \end{gathered} \tag{5.3}$$

Constraint Equations on the Control Variables. Constraints exist on the motors such that they can operate between minimum and maximum angular rates throughout the flight:

$$\omega_{i,min}^2 \leq \omega_i^2 \leq \omega_{i,max}^2\ ,\ i = 1, 2, 3, 4. \tag{5.4}$$

Approximation of the max motor spin rate occurs by multiplying the max battery voltage by the motor's KV rating to get rpm, which is converted to rad/s and then squared. The auxiliary control variables (slack variables), η_i, are defined such that $\boldsymbol{\eta} = (\eta_1, \eta_2, \eta_3, \eta_4) \in \mathbb{R}^d$ for $d = 4$ [126]. The slack variables convert the inequality constraint to equality constraints:

$$\Phi_i = (\omega_{i,max}^2 - \omega_i^2)(\omega_i^2 - \omega_{i,min}^2) - \eta_i^2 = 0\,, \tag{5.5}$$

where $i = 1, 2, 3, 4$. The control vector $\widetilde{\boldsymbol{u}} \in \mathbb{R}^8$ can be modified to include the slack variables:

$$\widetilde{\boldsymbol{u}} = [\omega_1\ \omega_2\ \omega_3\ \omega_4\ \eta_1\ \eta_2\ \eta_3\ \eta_4]. \tag{5.6}$$

The controls are admissible if they are defined and piece-wise continuous on some time interval $[t_0, t_1]$ and satisfy Eq. (5.5) [126].

Constraints on the State Variables. Besides maintaining altitudes below 400 ft, there are also velocity constraints on the DJI M100, in which the max ascent speed is 5 m/s, the max speed in ATTI mode is 22 m/s, and the max speed in GPS mode is 17 m/s [127].

Functional. It is required to find state and control functions that satisfy Eqs. (2.1), (5.1) - (5.5), and the functional based on the Mayer problem with a terminal cost for finite time is minimized [128]:

$$J = t_f - t_0 \to\ min. \tag{5.7}$$

5.2 First Differential of Functional

The first differential of the functional J is written as [126]:

$$\begin{aligned} dJ &= (\Gamma_{\boldsymbol{s}_0} + \boldsymbol{\lambda}_0^T)^T d\boldsymbol{s}_0 + (\Gamma_{\boldsymbol{s}_f} - \boldsymbol{\lambda}_f^T)^T d\boldsymbol{s}_f + (\Gamma_{t_0} - H_0)dt_0 + (\Gamma_{t_f} + H_f)dt_f \\ &+ \int_{t_0}^{t_f} \left[(H_{\boldsymbol{s}} + \dot{\boldsymbol{\lambda}}^T)^T \delta\boldsymbol{s} + (H_{\boldsymbol{\lambda}} - \dot{\boldsymbol{s}}^T)^T \delta\boldsymbol{\lambda} + H_{\boldsymbol{u}}^T \delta\boldsymbol{u} + H_{\boldsymbol{\eta}}^T \delta\eta \right] dt\,, \end{aligned} \tag{5.8}$$

where 0 and f denote initial and final respectively. The terms with Γ are defined as:

$$\begin{aligned} \Gamma_{t_0} &= \epsilon\frac{\partial J}{\partial t_0} + \boldsymbol{\mu}^T\frac{\partial \boldsymbol{E}}{\partial t_0}\,, \qquad \Gamma_{t_f} = \epsilon\frac{\partial J}{\partial t_f} + \boldsymbol{\beta}^T\frac{\partial \boldsymbol{F}}{\partial t_f}\,, \\ \Gamma_{s_0} &= \epsilon\frac{\partial J}{\partial \boldsymbol{s}_0} + \boldsymbol{\mu}^T\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{s}_0}\,, \qquad \Gamma_{s_f} = \epsilon\frac{\partial J}{\partial \boldsymbol{s}_f} + \boldsymbol{\beta}^T\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{s}_f}. \end{aligned} \tag{5.9}$$

Satisfying the first order necessary conditions of optimality means $dJ = 0$, so each of the terms in Eq. (5.8) must be zero. Each of the terms outside of the integral in Eq. (5.8) are zero by the transversality conditions. The first two terms in the integrand in Eq. (5.8) are zero from the canonical equations in Eq. (5.13). The last two terms in the integral are also zero by the local optimality conditions in Eq. (5.11). To summarize, when all the terms in Eq. (5.8) are zero, then the necessary conditions of optimality are satisfied. The second differential of the functional is not considered, so optimal control with a corresponding optimal trajectory cannot be claimed. Therefore, this paper can claim extremality at best but not optimality. Future work may encompass studying second order conditions and analyzing the second differential to determine if optimality can be claimed or not.

5.3 Hamiltonian Formalism

Figure 5.1 shows the overall flowchart of solving the optimal control problem. There are two main cases: 1) intermediate or variable motor thrust, and 2) constant motor thrust. The trivial variable motor thrust case yields zeros for all the Lagrange multipliers, while the nontrivial variable motor thrust case has six non-zero Lagrange multipliers. The constant motor thrust case contains non-zero Lagrange multipliers.

Both cases use the same state vector from Eq. (2.1), control vector and constraints from Eqs. (2.35) - (5.6), and functional from Eq. (5.7). However, the Pontryagin functions are slightly different, depending on which case is used. This leads to slightly different canonical equations and consequently, different canonical systems. The second order conditions are not considered, so both cases stop at extremal control. Considering second order conditions would result in a 16×16 matrix by taking second-order partial derivatives of the Pontryagin function with respect to the twelve state variables and the four control variables. For the controls to be optimal, the submatrix, $\mathbf{H}_{uu} = \frac{\partial^2 H}{\partial u^2}$, must be semi-positive definite by the Legendre-Clebsch condition. Then, the other submatrices, $\mathbf{H}_{xu} = \frac{\partial^2 H}{\partial x \partial u}$ and $\mathbf{H}_{xx} = \frac{\partial^2 H}{\partial x^2}$, must also be semi-positive definite, with the assumption that $\mathbf{H}_{xu} = \mathbf{H}_{ux}$ is symmetrical. Finally, the second order differential of the functional must also be semi-positive definite to demonstrate optimality [126].

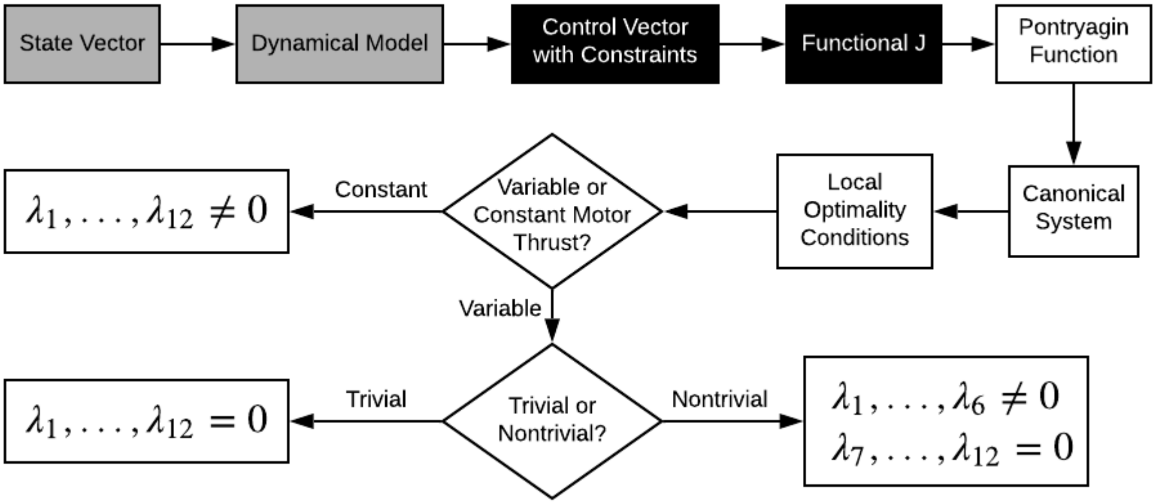


Figure 5.1: Hamiltonian Formalism Flowchart

Modified form of Pontryagin Function. Writing the Pontryagin function for time optimization with λ_i as the Lagrange multipliers and the intermediate constraints from Eq. (5.5) gives:

$$
\begin{aligned}
H(t,\boldsymbol{x},\widetilde{\boldsymbol{u}},\boldsymbol{\lambda},\boldsymbol{\nu}) &= \lambda_1 v_N + \lambda_2 v_E + \lambda_3 v_U + \lambda_4 \left[\gamma(2q_0q_2 + 2q_1q_3) - \frac{k_{dx}v_N}{m}\right] \\
&+ \lambda_5 \left[-\gamma(2q_0q_1 - 2q_2q_3) - \frac{k_{dy}v_E}{m}\right] + \lambda_6 \left[-g + \gamma(q_0^2 - q_1^2 - q_2^2 + q_3^2) - \frac{k_{dz}v_U}{m}\right] \\
&+ \frac{\lambda_7}{2}(-\omega_x q_1 - \omega_y q_2 - \omega_z q_3) + \frac{\lambda_8}{2}(-\omega_x q_0 + \omega_z q_2 - \omega_y q_3) + \sum_{i=1}^{4} \nu_i \Phi_i \\
&+ \frac{\lambda_9}{2}(-\omega_y q_0 - \omega_z q_1 - \omega_x q_3) + \lambda_{10}\left(\frac{\ell k(\omega_1^2 - \omega_3^2)}{I_{xx}} - \frac{I_{yy} - I_{zz}}{I_{xx}}\omega_y\omega_z\right) \\
&+ \lambda_{11}\left(\frac{\ell k(\omega_2^2 - \omega_4^2)}{I_{yy}} - \frac{I_{zz} - I_{xx}}{I_{yy}}\omega_x\omega_z\right) + \lambda_{12}\left(\frac{b(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)}{I_{zz}} - \frac{I_{xx} - I_{yy}}{I_{zz}}\omega_x\omega_y\right),
\end{aligned}
\tag{5.10}
$$

where $\widetilde{\boldsymbol{u}}$ is the modified control vector from Eq. (5.6) to include the auxiliary control variables. Note that the original form of the Pontryagin function does not include $\boldsymbol{\nu}^T\Phi$ [129], and adding $\boldsymbol{\nu}^T\Phi$ does not change the overall expression since $\Phi_i = 0$ as defined in Eq. (5.5). Recall that γ was defined in Eq. (2.25).

Local Optimality Conditions. The first-order local optimality condition must be satisfied to find extremals:

$$
\left[\frac{\partial H}{\partial \widetilde{\boldsymbol{u}}}\right]^T = 0. \tag{5.11}
$$

Applying Eq. (5.11) allows one to determine the auxiliary control variables:

$$
\begin{gathered}
\left[\frac{\partial H}{\partial \eta_i}\right]^T = 0 \;\text{ or }\; -2\nu_i\eta_i = 0\,,\; i = 1,2,3,4\,, \\
\begin{cases} \eta_i \neq 0 \text{ if } \nu_i = 0 \\ \eta_i = 0 \text{ if } \nu_i \neq 0 \end{cases},
\end{gathered}
\tag{5.12}
$$

where η_i depends on ν_i for $i = 1, ..., 4$. Therefore, there are two main cases from Eq. (5.12). The first case has variable motor thrust, which occurs when $\eta_i \neq 0, \nu_i = 0$. The second case has constant motor thrust when $\eta_i = 0, \nu_i \neq 0$ [126]. Other cases involve combinations of minimum, intermediate, and maximum motor spin rates.

Canonical Equations. The canonical equations form a canonical system and must be satisfied to find extremals:

$$
\left[\frac{\partial H}{\partial \boldsymbol{s}}\right]^T = -\dot{\boldsymbol{\lambda}}\,,\; \left[\frac{\partial H}{\partial \boldsymbol{\lambda}}\right]^T = \dot{\boldsymbol{s}}\,, \tag{5.13}
$$

where $\boldsymbol{s}$ is the state vector and $\boldsymbol{\lambda}$ is the vector of Lagrange multipliers.

Case I: Variable Motor Thrust. Case I has variable motor thrust such that $\eta_i \neq 0, \nu_i = 0$. For this case, $\nu_i = 0$, which causes the summation in the Pontryagin function to vanish. The η_i are arbitrary since they vanish by $\nu_i = 0$. Then, applying Eq. (5.11) for the motor spin rates and factoring the ω_i gives:

$$\begin{aligned} \frac{\partial H}{\partial \omega_1} &= 0 = \omega_1 \left[L + \frac{2\ell k \lambda_{10}}{I_{xx}} + \frac{2b\lambda_{12}}{I_{zz}} \right], \; \frac{\partial H}{\partial \omega_2} = 0 = \omega_2 \left[L + \frac{2\ell k \lambda_{11}}{I_{yy}} - \frac{2b\lambda_{12}}{I_{zz}} \right], \\ \frac{\partial H}{\partial \omega_3} &= 0 = \omega_3 \left[L - \frac{2\ell k \lambda_{10}}{I_{xx}} + \frac{2b\lambda_{12}}{I_{zz}} \right], \; \frac{\partial H}{\partial \omega_4} = 0 = \omega_4 \left[L - \frac{2\ell k \lambda_{11}}{I_{yy}} - \frac{2b\lambda_{12}}{I_{zz}} \right], \end{aligned} \tag{5.14}$$

in which L contains three Lagrange multipliers and other variables:

$$L = \frac{4k\lambda_4}{m}(q_0 q_2 + q_1 q_3) + \frac{4k\lambda_5}{m}(q_2 q_3 - q_0 q_1) + \frac{2k\lambda_6}{m}(q_0^2 - q_1^2 - q_2^2 + q_3^2). \tag{5.15}$$

Two subcases exist for the variable motor thrust case associated with Eq. (5.14). One is a trivial case where the Lagrange multipliers are identically zero, while the other subcase is nontrivial with six nonzero Lagrange multipliers, as shown in Fig. 5.1. Conveniently, the first three Lagrange multipliers are the same for each subcase. Applying the canonical equations gives the first three Lagrange multipliers:

$$\frac{\partial H}{\partial x} = -\dot{\lambda}_1 = 0\,, \; \frac{\partial H}{\partial y} = -\dot{\lambda}_2 = 0\,, \; \frac{\partial H}{\partial z} = -\dot{\lambda}_3 = 0\,, \tag{5.16}$$

in which $\lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}$.

Variable Motor Thrust - Trivial Case. For the trivial case, Eq. (5.14) demonstrates that either the ω_i or the terms in the brackets must be zero to satisfy the first-order necessary condition from Eq. (5.11). Since this is the variable motor thrust case, it does not make sense for $\omega_i = 0$. Therefore, the sum in the brackets must be zero. For the first-order necessary condition, Eq. (5.14) will always be zero for any state if the Lagrange multipliers are identically zero, which quickly and conveniently determines six of the twelve Lagrange multipliers. Applying $\frac{\partial H}{\partial \boldsymbol{\lambda}}$ from Eq. (5.13) to Eq. (5.10) gives the original dynamical model with static thrust, so no new information exists regarding the remaining six Lagrange multipliers. Since the terms in the brackets of Eq. (5.10) are zero, one can use $\lambda_4 = \lambda_5 = \lambda_6 = 0$ to find $\lambda_1, \lambda_2, \lambda_3$:

$$\dot{\lambda}_4 = \lambda_4 k_{dx} - \lambda_1 = 0\,, \; \lambda_1 = \lambda_4 k_{dx} = 0 \tag{5.17}$$

and $\lambda_5 = \lambda_6 = 0$ by the same computation. Applying $\frac{\partial H}{\partial \boldsymbol{s}}$ from Eq. (5.13) to the last three $\dot{\lambda}_i$ and using $\lambda_{10} = \lambda_{11} = \lambda_{12} = 0$ gives:

$$\begin{aligned}
\dot{\lambda}_{10} &= \frac{\lambda_7 q_1}{2} - \frac{\lambda_8 q_0}{2} - \frac{\lambda_9 q_3}{2} = 0\,, \\
\dot{\lambda}_{11} &= \frac{\lambda_7 q_2}{2} - \frac{\lambda_8 q_2}{2} - \frac{\lambda_9 q_3}{2} = 0\,, \\
\dot{\lambda}_{12} &= \frac{\lambda_7 q_1}{2} - \frac{\lambda_8 q_0}{2} - \frac{\lambda_9 q_3}{2} = 0.
\end{aligned} \tag{5.18}$$

From Eqs. (5.14) and (5.18), $\lambda_{10}, \lambda_{11}$, and λ_{12} become arbitrary constants. In particular, it is possible to have $\lambda_7 = \lambda_8 = \lambda_9 = \lambda_{10} = \lambda_{11} = \lambda_{12} = 0 \ \forall q_i$ for $i = 0, 1, 2, 3$, which means $\boldsymbol{\lambda} = \mathbf{0}$. The transversality condition for a minimum time problem is $\frac{\partial J}{\partial t_f} = 1$ [126].

The Hamiltonian does not explicitly depend on time, so energy is conserved, resulting in H being constant, so $H = 1$ by the transversality condition. In the unlikely but possible event that one or more of the motors stops spinning due to some type of mechanical or electrical failure, ω_i would become zero, which can make the terms in the bracket be non-zero. One study shows that horizontal attitude can be stable while sacrificing heading control (yaw) if a motor stops working [130]. Since all the Lagrange multipliers are zero, the dynamical model equations can be used to determine the state variables, which generates a field of feasible trajectories instead of a field of extremals. Overall, the trivial variable motor thrust case leads to feasible trajectories and not extremal trajectories since all its Lagrange multipliers are zero, as shown in Fig. 5.1.

Variable Motor Thrust - Nontrivial Case. Applying $\frac{\partial H}{\partial \boldsymbol{s}}$ from Eq. (5.13) to the next three state variables (v_N, v_E, v_U) to Eq. (5.10) gives:

$$\frac{\partial H}{\partial v_N} = -\dot{\lambda}_4 = \lambda_1 - \frac{\lambda_4 k_{dx}}{m}\,, \ \frac{\partial H}{\partial v_E} = -\dot{\lambda}_5 = \lambda_2 - \frac{\lambda_5 k_{dy}}{m}\,, \ \frac{\partial H}{\partial v_U} = -\dot{\lambda}_6 = \lambda_3 - \frac{\lambda_6 k_{dz}}{m}. \tag{5.19}$$

Using the method of integrating factors provides solutions to these differential equations:

$$\lambda_4 = \frac{\lambda_1 m}{k_{dx}} + C_1\,, \qquad \lambda_5 = \frac{\lambda_2 m}{k_{dy}} + C_2\,, \qquad \lambda_6 = \frac{\lambda_3 m}{k_{dz}} + C_3\,, \tag{5.20}$$

where C_1, C_2, C_3 are integration constants. The next logical step involves determining these integration constants. However, notice that the differential equations in Eq. (5.19) can be rewritten as:

$$\frac{d\lambda_4}{dt} = \frac{\lambda_4 k_{dx}}{m} - \lambda_1\,, \qquad \frac{d\lambda_5}{dt} = \frac{\lambda_5 k_{dy}}{m} - \lambda_2\,, \qquad \frac{d\lambda_6}{dt} = \frac{\lambda_6 k_{dz}}{m} - \lambda_3. \tag{5.21}$$

Integrating by using the clever derivative substitution in Ref. [119] with the chain rule gives

$\lambda_4(t), \lambda_5(t), \lambda_6(t)$ as:

$$\begin{aligned}
\lambda_4(t) &= \frac{\lambda_1}{A_x} + \frac{exp(A_x t)}{A_x}\left(\lambda_4(t_0)A_x - \lambda_1\right), \\
\lambda_5(t) &= \frac{\lambda_2}{A_y} + \frac{exp(A_y t)}{A_y}\left(\lambda_5(t_0)A_y - \lambda_2\right), \\
\lambda_6(t) &= \frac{\lambda_3}{A_z} + \frac{exp(A_z t)}{A_z}\left(\lambda_6(t_0)A_z - \lambda_3\right),
\end{aligned} \tag{5.22}$$

where a change of variables makes the notation more compact:

$$A_x = \frac{k_{dx}}{m}, \; A_y = \frac{k_{dy}}{m}, \; A_z = \frac{k_{dz}}{m}, \tag{5.23}$$

and the clever substitution is: [119]

$$d(\ln(\lambda_4 k_d - \lambda_1) = \frac{1}{\lambda_4 k_d - \lambda_1}(d\lambda_4)(k_d).$$

Notice how Eq. (5.22) resembles Eq. (5.20), which means the integration constants can be determined if the Lagrange multiplier at the initial time t_0 is known. Recall that Eq. (5.14) have $\lambda_{10}, \lambda_{11}, \lambda_{12}$. By setting the terms in the brackets of the $\frac{\partial H}{\partial \omega_1}$ and $\frac{\partial H}{\partial \omega_3}$ equations in Eq. (5.14) equal to each other shows that the L and λ_{12} terms cancel, so only the λ_{10} terms remain:

$$\frac{2\ell k \lambda_{10}}{I_{xx}} = -\frac{2\ell k \lambda_{10}}{I_{xx}}.$$

Canceling terms implies that $\lambda_{10} = 0$. Setting the brackets of the $\frac{\partial H}{\partial \omega_1}$ and $\frac{\partial H}{\partial \omega_2}$ equations in Eq. (5.14) equal to each other yields similar results such that $\lambda_{11} = 0$. Setting the $\frac{\partial H}{\partial \omega_2}$ and $\frac{\partial H}{\partial \omega_4}$ equations Eq. (5.14) equal to each other causes the L terms to cancel:

$$\frac{2\ell k \lambda_{10}}{I_{xx}} + \frac{2\lambda_{12} b}{I_{zz}} = \frac{2\ell k \lambda_{11}}{I_{yy}} - \frac{2\lambda_{12} b}{I_{zz}}.$$

Using the previous results with $\lambda_{10} = \lambda_{11} = 0$ and solving for λ_{12} reveals $\lambda_{12} = 0$. Thus, $L = 0$ from Eq. (5.15), which ensures $\frac{\partial H}{\partial \omega_i} = 0$, and $L = 0$ if one of the Lagrange multipliers can be solved in terms of the other two Lagrange multipliers. A quick recap up to this point shows that the first three Lagrange multipliers are real numbers, and the last three Lagrange multipliers are all zero.

Applying the canonical equations to the last three state variables $(\omega_x, \omega_y, \omega_z)$ gives:

$$\begin{aligned}
\dot{\lambda}_{10} &= \frac{\lambda_7 q_1}{2} - \frac{\lambda_8 q_0}{2} - \frac{\lambda_9 q_3}{2} = 0\,, \\
\dot{\lambda}_{11} &= \frac{\lambda_7 q_2}{2} - \frac{\lambda_8 q_3}{2} - \frac{\lambda_9 q_0}{2} = 0\,, \\
\dot{\lambda}_{12} &= \frac{\lambda_7 q_3}{2} - \frac{\lambda_8 q_2}{2} - \frac{\lambda_9 q_1}{2} = 0.
\end{aligned} \tag{5.24}$$

The differential equations in Eq. (5.24) are zero because $\lambda_{10} = \lambda_{11} = \lambda_{12} = 0$. Therefore, these equations can be solved algebraically, which gives $\lambda_7 = \lambda_8 = \lambda_9 = 0$. This is confirmed through MATLAB's *solve* function. A quick recap up to this point shows that nine of the twelve Lagrange multipliers are determined in which only $\lambda_4(t_0), \lambda_5(t_0)$, and $\lambda_6(t_0)$ are unknown, and they will be determined using the first-order necessary condition for optimality.

Recall from Eq. (5.15) that $L = 0$ to fulfill the first-order necessary condition for optimality. Thus, setting one of the Lagrange multipliers as a function of the other two Lagrange multipliers ensures $L = 0$. In particular, one can choose $\lambda_6(t_0)$ as a function of $\lambda_4(t_0)$ and $\lambda_5(t_0)$ to ensure $L = 0$:

$$\lambda_6(t_0) = \frac{-\lambda_4(t_0)(q_0 q_2 + q_1 q_3) - \lambda_5(t_0)(q_2 q_3 + q_0 q_1)}{q_0^2 - q_1^2 - q_2^2 + q_3^2}. \tag{5.25}$$

Next, selecting initial values for $\lambda_4(t_0)$ and $\lambda_5(t_0)$ gives the initial value for $\lambda_6(t_0)$. Once values for $\lambda_1, \lambda_2, \lambda_3$ are chosen, the costate equations for $\lambda_4(t), \lambda_5(t)$, and $\lambda_6(t)$ can be computed at any time. Now, all the costates are known and the first-order necessary condition is satisfied, so a field of extremals is generated. It is interesting to point out that extremality is guaranteed for any arbitrary time with any $\lambda_4(t)$ and $\lambda_5(t)$ regardless of the other parameters and variables if and only if Eq. (5.25) is satisfied.

Overall, the nontrivial variable motor thrust case generates a field of extremals with six nonzero Lagrange multipliers $(\lambda_1, ..., \lambda_6)$ and six Lagrange multipliers $(\lambda_7, ..., \lambda_{12})$ equal to zero as shown in Fig. 5.1. The takeoff maneuver and the waypoint guidance maneuvers utilize the nontrivial variable motor thrust case.

Case II: Constant Motor Thrust. Case II has constant motor thrust such that $\eta_i = 0, \nu_i \neq 0$. It is conventional for controls to take on their boundary values, i.e., minimum or maximum. However, the controls can take intermediate values for hovering. Applying the first-order necessary condition for control and factoring the ω_i gives:

$$
\begin{aligned}
\frac{\partial H}{\partial \omega_1} &= 0 = \omega_1\left[L + \frac{2\ell k \lambda_{10}}{I_{xx}} + \frac{2b\lambda_{12}}{I_{zz}} + 2\nu_1(\omega_{1,max}^2 + \omega_{1,min}^2)\right] - 4\nu_1\omega_1^3\,,\\
\frac{\partial H}{\partial \omega_2} &= 0 = \omega_2\left[L + \frac{2\ell k \lambda_{11}}{I_{yy}} - \frac{2b\lambda_{12}}{I_{zz}} + 2\nu_2(\omega_{2,max}^2 + \omega_{2,min}^2)\right] - 4\nu_2\omega_2^3\,,\\
\frac{\partial H}{\partial \omega_3} &= 0 = \omega_3\left[L - \frac{2\ell k \lambda_{10}}{I_{xx}} + \frac{2b\lambda_{12}}{I_{zz}} + 2\nu_3(\omega_{3,max}^2 + \omega_{3,min}^2)\right] - 4\nu_3\omega_3^3\,,\\
\frac{\partial H}{\partial \omega_4} &= 0 = \omega_4\left[L - \frac{2\ell k \lambda_{11}}{I_{yy}} - \frac{2b\lambda_{12}}{I_{zz}} + 2\nu_4(\omega_{4,max}^2 + \omega_{4,min}^2)\right] - 4\nu_4\omega_4^3\,,
\end{aligned}
\tag{5.26}
$$

where L is the same as in Eq. (5.15). Since the control is either minimum or maximum (constant motor thrust case), ν_i can be determined from Eq. (5.26):

$$
\begin{aligned}
\nu_1 &= \frac{L + \frac{2\ell k\lambda_{10}}{I_{xx}} + \frac{2b\lambda_{12}}{I_{zz}}}{4\omega_1^2 - 2(\omega_{1,max}^2 - \omega_{1,min}^2)}\,,\ \nu_2 = \frac{L + \frac{2\ell k\lambda_{11}}{I_{yy}} - \frac{2b\lambda_{12}}{I_{zz}}}{4\omega_2^2 - 2(\omega_{2,max}^2 - \omega_{2,min}^2)}\,,\\
\nu_3 &= \frac{L - \frac{2\ell k\lambda_{10}}{I_{xx}} + \frac{2b\lambda_{12}}{I_{zz}}}{4\omega_3^2 - 2(\omega_{3,max}^2 - \omega_{3,min}^2)}\,,\ \nu_4 = \frac{L - \frac{2\ell k\lambda_{11}}{I_{yy}} - \frac{2b\lambda_{12}}{I_{zz}}}{4\omega_4^2 - 2(\omega_{4,max}^2 - \omega_{4,min}^2)}.
\end{aligned}
\tag{5.27}
$$

As mentioned before, the first three Lagrange multipliers are the same for the variable motor thrust case and the constant motor thrust case, which is very convenient. Applying Eq. (5.13) for the first three state variables (p_N, p_E, p_U) gives the first three Lagrange multipliers:

$$
\frac{\partial H}{\partial x} = -\dot{\lambda}_1 = 0\,,\ \frac{\partial H}{\partial y} = -\dot{\lambda}_2 = 0\,,\ \frac{\partial H}{\partial z} = -\dot{\lambda}_3 = 0
\tag{5.28}
$$

where $\lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}$ such that $\lambda_i = a_i$ with a_i as integration constants for $i = 1, 2, 3$, which is the same result as before. The closed-form solutions for $\lambda_4, \lambda_5, \lambda_6$ are also conveniently the same as the variable motor thrust case:

$$
\begin{aligned}
\lambda_4(t) &= \frac{\lambda_1}{A_x} + \frac{exp(A_x t)}{A_x}\Big(\lambda_4(t_0)A_x - \lambda_1\Big)\,,\\
\lambda_5(t) &= \frac{\lambda_2}{A_y} + \frac{exp(A_y t)}{A_y}\Big(\lambda_5(t_0)A_y - \lambda_2\Big)\,,\\
\lambda_6(t) &= \frac{\lambda_3}{A_z} + \frac{exp(A_z t)}{A_z}\Big(\lambda_6(t_0)A_z - \lambda_3\Big).
\end{aligned}
\tag{5.29}
$$

Applying $\frac{\partial H}{\partial \boldsymbol{s}}$ from Eq. (5.13) to the next state variables (q_0, q_1, q_2) to Eq. (5.10) yields:

$$\begin{aligned} \frac{\partial H}{\partial q_0} &= \dot{\lambda}_7 = -\lambda_4 \Omega(2q_2) + \lambda_5 \Omega(2q_1) - \lambda_6 \Omega(2q_0) + \frac{\lambda_8 \omega_x}{2} + \frac{\lambda_9 \omega_y}{2} \ , \\ \frac{\partial H}{\partial q_1} &= \dot{\lambda}_8 = -\lambda_4 \Omega(2q_3) + \lambda_5 \Omega(2q_0) + \lambda_6 \Omega(2q_1) + \frac{\lambda_7 \omega_x}{2} + \frac{\lambda_9 \omega_z}{2} \ , \\ \frac{\partial H}{\partial q_2} &= \dot{\lambda}_9 = -\lambda_4 \Omega(2q_0) - \lambda_5 \Omega(2q_3) + \lambda_6 \Omega(2q_2) + \frac{\lambda_7 \omega_y}{2} - \frac{\lambda_8 \omega_z}{2} \ , \end{aligned} \tag{5.30}$$

in which a change of variables provides compact notation:

$$\Omega = \frac{k}{m}\left(\sum_{i=1}^{4} \omega^2_{i,max/min}\right). \tag{5.31}$$

Just to clarify Eq. (5.31), the "/" in $\omega^2_{i,max/min}$ means "or" and not divide. Applying Eq. (5.13) to the last three state variables $(\omega_x, \omega_y, \omega_z)$ to Eq. (5.10) yields:

$$\begin{aligned} \frac{\partial H}{\partial \omega_x} &= \dot{\lambda}_{10} = \frac{\lambda_7 q_1}{2} + \frac{\lambda_8 q_0}{2} + \frac{\lambda_9 q_3}{2} + \frac{\lambda_{11} \omega_z}{I_{yy}}\Big(I_{zz} - I_{yy}\Big) + \frac{\lambda_{12} \omega_y}{I_{zz}}\Big(I_{xx} - I_{yy}\Big), \\ \frac{\partial H}{\partial \omega_y} &= \dot{\lambda}_{11} = \frac{\lambda_7 q_2}{2} + \frac{\lambda_8 q_3}{2} + \frac{\lambda_9 q_0}{2} + \frac{\lambda_{10} \omega_z}{I_{xx}}\Big(I_{yy} - I_{zz}\Big) + \frac{\lambda_{12} \omega_x}{I_{yy}}\Big(I_{xx} - I_{yy}\Big) \ , \\ \frac{\partial H}{\partial \omega_z} &= \dot{\lambda}_{12} = \frac{\lambda_7 q_3}{2} - \frac{\lambda_8 q_2}{2} + \frac{\lambda_9 q_1}{2} + \frac{\lambda_{10} \omega_y}{I_{xx}}\Big(I_{yy} - I_{zz}\Big) + \frac{\lambda_{11} \omega_x}{I_{yy}}\Big(I_{zz} - I_{xx}\Big). \end{aligned} \tag{5.32}$$

Constant Motor Thrust: Dynamical Model Comparison Recall that the angular acceleration equations of the quadcopter dynamical model given in [100] are:

$$\dot{\boldsymbol{\Omega}} = \boldsymbol{\alpha} = \begin{bmatrix} \tau_\phi / I_{xx} \\ \tau_\theta / I_{yy} \\ \tau_\psi / I_{zz} \end{bmatrix} - \begin{bmatrix} \frac{I_{yy} - I_{zz}}{I_{xx}} \omega_y \omega_z \\ \frac{I_{zz} - I_{xx}}{I_{yy}} \omega_x \omega_z \\ \frac{I_{xx} - I_{yy}}{I_{zz}} \omega_x \omega_y \end{bmatrix} \ , \tag{5.33}$$

where $\boldsymbol{\alpha}$ is the angular acceleration vector. Applying Eq. (5.13) with respect to the last three costate variables yields:

$$\begin{aligned} \frac{\partial H}{\partial \lambda_{10}} &= \dot{\omega}_x = \alpha_\phi = -\frac{I_{yy} - I_{zz}}{I_{xx}} \omega_y \omega_z \ , \\ \frac{\partial H}{\partial \lambda_{11}} &= \dot{\omega}_y = \alpha_\theta = -\frac{I_{zz} - I_{xx}}{I_{yy}} \omega_x \omega_z \ , \\ \frac{\partial H}{\partial \lambda_{12}} &= \dot{\omega}_z = \alpha_\psi = -\frac{I_{xx} - I_{yy}}{I_{zz}} \omega_x \omega_y . \end{aligned} \tag{5.34}$$

Comparing Eq. (5.33) with Eq. (5.34) shows that $\boldsymbol{\tau} = \boldsymbol{0}$ in Eq. (5.33). Recall that the static thrust torque model in the body frame is [100]:

$$\boldsymbol{\tau} = \begin{bmatrix} \ell k(\omega_2^2 - \omega_4^2) \\ \ell k(\omega_1^2 - \omega_3^2) \\ b(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} . \tag{5.35}$$

Since $\boldsymbol{\tau} = \mathbf{0}$ and all the motor spin rates are identical, the motor spin rates cancel each other out. Then, torques and rotations about the body axes cease to exist, which means $\omega_x = \omega_y = \omega_z = 0$. Overall, the constant motor thrust case applies to quadcopter vertical maneuvers such as ascent, hovering (constant intermediate motor thrust), and descent. The max motor thrust takeoff maneuver uses the constant motor thrust case. Since there are no angular velocities in the constant motor thrust case from the discussion in the previous subsection, this simplifies the last six costate ordinary differential equations to:

$$\begin{aligned} \frac{\partial H}{\partial q_0} &= \dot{\lambda}_7 = -2\lambda_4\Omega q_2 + 2\lambda_5\Omega q_1 - 2\lambda_6\Omega q_0 \,, \\ \frac{\partial H}{\partial q_1} &= \dot{\lambda}_8 = -2\lambda_4\Omega q_3 + 2\lambda_5\Omega q_0 + 2\lambda_6\Omega q_1 \,, \\ \frac{\partial H}{\partial q_2} &= \dot{\lambda}_9 = -2\lambda_4\Omega q_0 - 2\lambda_5\Omega q_3 + 2\lambda_6\Omega q_2 \,, \end{aligned} \tag{5.36}$$

$$\begin{aligned} \frac{\partial H}{\partial \omega_x} &= \dot{\lambda}_{10} = \frac{\lambda_7 q_1}{2} + \frac{\lambda_8 q_0}{2} + \frac{\lambda_9 q_3}{2} \,, \\ \frac{\partial H}{\partial \omega_y} &= \dot{\lambda}_{11} = \frac{\lambda_7 q_2}{2} + \frac{\lambda_8 q_3}{2} + \frac{\lambda_9 q_0}{2} \,, \\ \frac{\partial H}{\partial \omega_z} &= \dot{\lambda}_{12} = \frac{\lambda_7 q_3}{2} - \frac{\lambda_8 q_2}{2} + \frac{\lambda_9 q_1}{2} . \end{aligned} \tag{5.37}$$

One can solve the costates in Eq. (5.36) by direct integration. Consider rewriting $\dot{\lambda}_7$ as:

$$\frac{d\lambda_7}{dt} = -2\lambda_4\Omega q_2 + 2\lambda_5\Omega q_1 - 2\lambda_6 q_0 \,, \tag{5.38}$$

Similar equations can be written for λ_8 and λ_9. Integrating, regrouping terms, and using Eq. (5.22) yields:

$$\begin{aligned} \lambda_7(t) &= -2q_2 L_1 + 2q_1 L_2 - 2q_0 L_3 + \lambda_7(t_0) \,, \\ \lambda_8(t) &= -2q_3 L_1 + 2q_0 L_2 + 2q_1 L_3 + \lambda_8(t_0) \,, \\ \lambda_9(t) &= -2q_0 L_1 - 2q_3 L_2 + 2q_2 L_3 + \lambda_9(t_0) \,, \end{aligned} \tag{5.39}$$

where a change of variables gives a more compact form:

$$\begin{aligned}
L_1 &= \Omega\left[\frac{\lambda_1 t}{A_x} + \frac{exp(A_x t)}{A_x^2}\left(\lambda_4(t_0)A_x - \lambda_1\right)\right], \\
L_2 &= \Omega\left[\frac{\lambda_2 t}{A_y} + \frac{exp(A_y t)}{A_y^2}\left(\lambda_5(t_0)A_y - \lambda_2\right)\right], \\
L_3 &= \Omega\left[\frac{\lambda_3 t}{A_z} + \frac{exp(A_x t)}{A_z^2}\left(\lambda_6(t_0)A_z - \lambda_3\right)\right].
\end{aligned} \tag{5.40}$$

Even though there are no angular velocities in the constant motor thrust case, the costates corresponding to angular velocities can still be determined using Eq. (5.37) through direct integration. Consider rewriting $\dot{\lambda}_{10}$, $\dot{\lambda}_{11}$, and $\dot{\lambda}_{12}$ as

$$\begin{aligned}
\frac{d\lambda_{10}}{dt} &= \frac{\lambda_7 q_1}{2} + \frac{\lambda_8 q_0}{2} + \frac{\lambda_9 q_3}{2}, \\
\frac{d\lambda_{11}}{dt} &= \frac{\lambda_7 q_2}{2} + \frac{\lambda_8 q_3}{2} + \frac{\lambda_9 q_0}{2}, \\
\frac{d\lambda_{12}}{dt} &= \frac{\lambda_7 q_3}{2} - \frac{\lambda_8 q_2}{2} + \frac{\lambda_9 q_1}{2}.
\end{aligned} \tag{5.41}$$

Defining the following terms help provide a compact form for solving (5.41):

$$M_1 = \frac{\Omega\lambda_1 t^2}{2A_x}, \; M_2 = \frac{\Omega\lambda_2 t^2}{2A_y}, \; M_3 = \frac{\Omega\lambda_3 t^2}{2A_z}, \tag{5.42}$$

$$\begin{aligned}
N_1 &= \frac{\Omega exp(A_x t)}{A_x^3}\left(\lambda_4(t_0)A_x - \lambda_1\right), \\
N_2 &= \frac{\Omega exp(A_y t)}{A_y^3}\left(\lambda_5(t_0)A_y - \lambda_2\right), \\
N_3 &= \frac{\Omega exp(A_z t)}{A_z^3}\left(\lambda_6(t_0)A_z - \lambda_3\right).
\end{aligned} \tag{5.43}$$

Integrating, regrouping terms, and using Eqs. (5.39)-(5.43) gives $\lambda_{10}(t), \lambda_{11}(t), \lambda_{12}(t)$ as:

$$\begin{aligned}
\lambda_{10}(t) &= -q_1q_2(M_1+N_1) + q_1^2(M_2+N_2) - q_1q_0(M_3+N_3) + \frac{\lambda_7(t_0)q_1t}{2} \\
&\quad + q_0q_3(M_1+N_1) - q_0^2(M_2+N_2) - q_0q_1(M_3+N_3) - \frac{\lambda_8(t_0)q_0t}{2} \\
&\quad + q_3q_0(M_1+N_1) + q_3^2(M_2+N_2) - q_3q_2(M_3+N_3) - \frac{\lambda_9(t_0)q_3t}{2} + \lambda_{10}(t_0)\,, \\
\lambda_{11}(t) &= -q_2^2(M_1+N_1) + q_2q_1(M_2+N_2) - q_2q_0(M_3+N_3) + \frac{\lambda_7(t_0)q_2t}{2} \\
&\quad - q_3^2(M_1+N_1) + q_3q_0(M_2+N_2) + q_3q_1(M_3+N_3) + \frac{\lambda_8(t_0)q_3t}{2} \\
&\quad + q_0^2(M_1+N_1) + q_0q_3(M_2+N_2) - q_0q_2(M_3+N_3) - \frac{\lambda_9(t_0)q_0t}{2} + \lambda_{11}(t_0)\,, \\
\lambda_{12}(t) &= -q_3q_2(M_1+N_1) + q_3q_1(M_2+N_2) - q_3q_0(M_3+N_3) + \frac{\lambda_7(t_0)q_3t}{2} \\
&\quad + q_2q_3(M_1+N_1) - q_2q_0(M_2+N_2) - q_2q_1(M_3+N_3) - \frac{\lambda_8(t_0)q_2t}{2} \\
&\quad - q_1q_0(M_1+N_1) - q_1q_3(M_2+N_2) + q_1q_2(M_3+N_3) + \frac{\lambda_9(t_0)q_1t}{2} + \lambda_{12}(t_0)\,,
\end{aligned} \tag{5.44}$$

Overall, the constant motor thrust case generates a field of extremals with all non-zero Lagrange multipliers ($\lambda_1, ..., \lambda_{12}$) as shown in Fig. 5.1.

CHAPTER 6
MANEUVERS WITH EXTREMAL CONTROL AND EXPLICIT GUIDANCE

This chapter follows the work of Ref. [131, 119, 120, 121, 123, 124] and describes the procedures and implementation of the extremal control and explicit guidance maneuvers.

6.1 Guided Takeoff and Waypoint Maneuver Procedures

Figure 6.1 shows the overall flowchart for the takeoff and waypoint guidance maneuvers. The gray-filled boxes show the steps of the waypoint guidance maneuver, and the white-filled boxes show the steps of the takeoff maneuver.

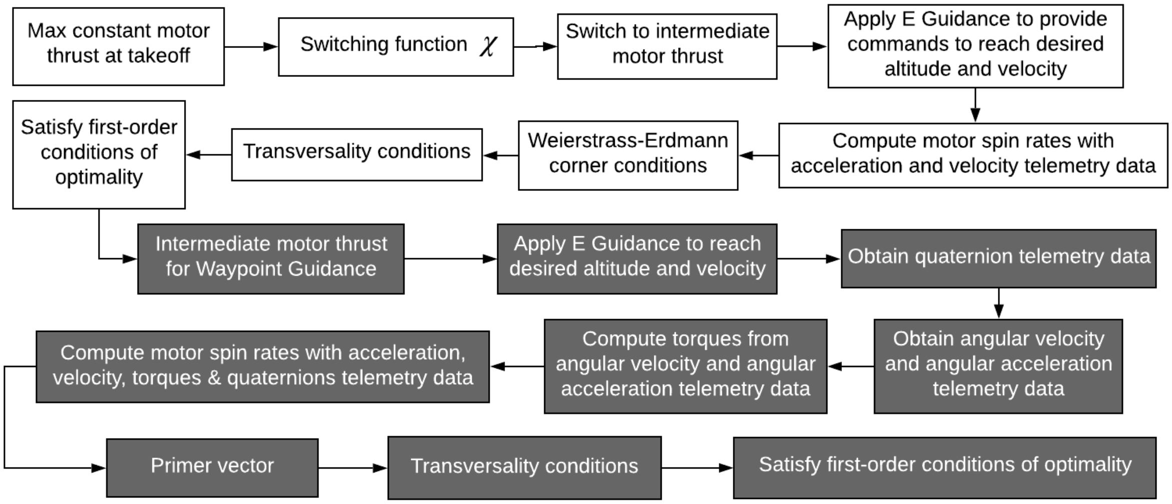


Figure 6.1: Flowchart for Takeoff and Waypoint Guidance Maneuvers

The C++ DJI OSDK libraries and functions integrate the proposed extremal control and explicit guidance. Figure 6.2 shows the flowchart of the C++ DJI OSDK implementation. The light gray-filled boxes indicate steps that use DJI OSDK's *velocityAndYawRateCtrl* function, the dark gray-filled boxes use DJI OSDK's *positionAndYawCtrl* function, and the black-filled boxes show the start of the maneuvers.

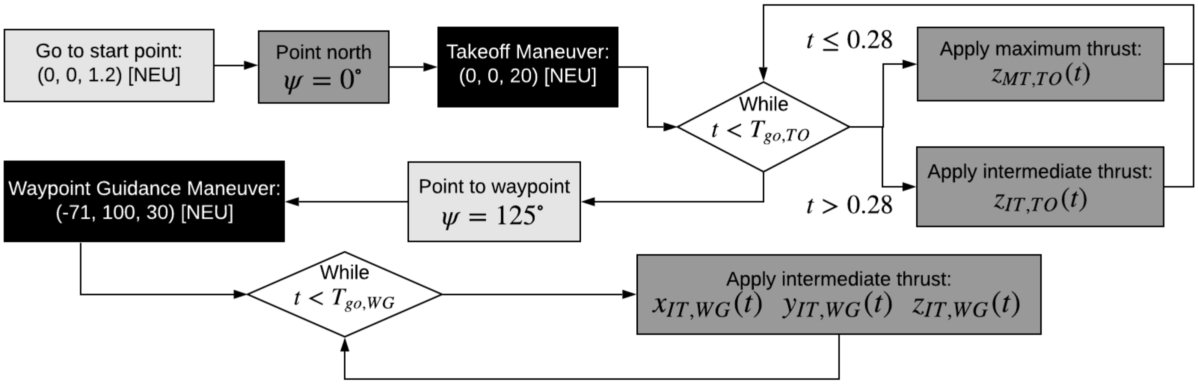


Figure 6.2: DJI OSDK Flowchart for Extremal Control and Explicit Guidance Integration

Initial State Constraint Equations. There are four initial constraint equations, which act as the initial constraints for the takeoff maneuver. The DJI M100 has an auto-takeoff function to takeoff to an altitude of 1.2 m, so the initial state constraint equations start after auto-takeoff:

$$\begin{aligned} E_1(p_N, p_E, p_U) = (0,0,1.2)\ ,\ E_2(v_N, v_E, v_U) = (0,0,0)\ , \\ E_3(\phi, \theta, \psi) = (0,0,0)\ ,\ E_4(\omega_x, \omega_y, \omega_z) = (0,0,0). \end{aligned} \tag{6.1}$$

Intermediate State Constraint Equations. The intermediate state constraint is the takeoff altitude of 20 m, so the quadcopter ascends to 20 m before heading to the waypoint while maintaining an altitude below 400 ft [125].

$$\begin{aligned} F_1(p_N, p_E, p_U)) = (0,0,20)\ ,\ F_2(v_N, v_E, v_U) = (0,0,0)\ , \\ F_3(\phi, \theta, \psi) = (0,0,0)\ ,\ F_4(\omega_x, \omega_y, \omega_z) = (0,0,0). \end{aligned} \tag{6.2}$$

Final State Constraint Equations. Once the quadcopter reaches an altitude of 20 m, it turns and flies towards the waypoint. There are four final constraint equations:

$$\begin{aligned} G_1(p_N, p_E, p_U) = (-71,100,30)\ ,\ G_2(v_N, v_E, v_U) = (0,0,0)\ , \\ G_3(\phi, \theta) = (0,0)\ ,\ G_4(\omega_x, \omega_y, \omega_z) = (0,0,0). \end{aligned} \tag{6.3}$$

These final position coordinates are determined and rounded to the nearest whole number from the difference between the takeoff location and the final desired location's latitude and longitude. The final yaw angle is not constrained because the heading has a smaller impact than the roll and pitch angles for applications such as taking pictures for inspection purposes at a waypoint [1].

6.2 Guided Takeoff and Roll Maneuvers

Figure 6.3 shows the overall flowchart for the takeoff and roll maneuvers. The gray-filled boxes show the steps of the roll maneuver, and the white-filled boxes show the steps of the takeoff maneuver.

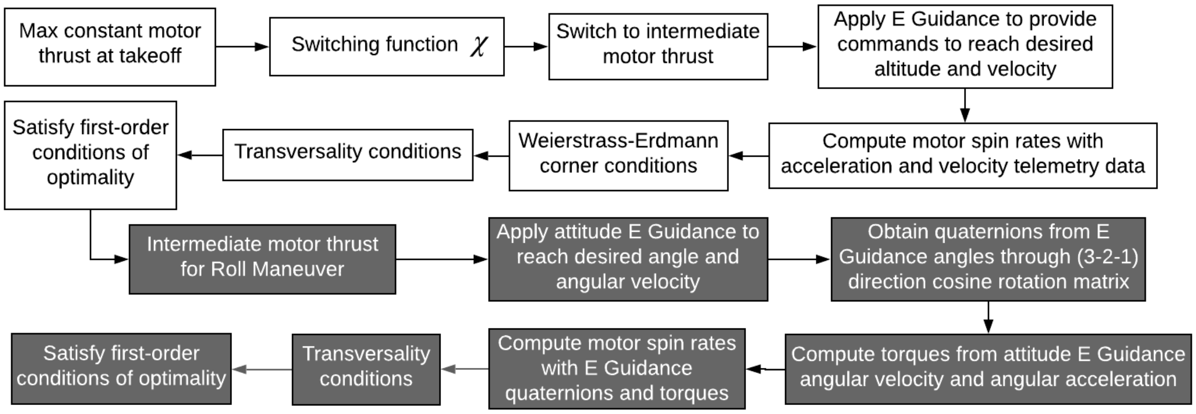


Figure 6.3: Flowchart for Takeoff and Roll Maneuvers

The takeoff maneuver follows exactly the same steps in Fig. 6.2 but with a desired altitude of 60 m instead of 20 m, so conceptually everything remains the same. Unfortunately, it is unknown how to remove the "training wheels" on the DJI M100 to permit aerobatic maneuvers such as the 360° roll maneuver. Thus, the DJI OSDK function, *attitudeAndVertPosCtrl*, cannot produce the 360° roll maneuver because the "training wheels" constrain the roll and pitch angles to only 35°. Thus, there is simulated data for the roll maneuver, but experimental data exists for the takeoff manevuer.

Initial State Constraint Equations. There are four initial constraint equations, which act as the initial constraints for the takeoff maneuver. The DJI M100 has an auto-takeoff function to takeoff to an altitude of 1.2 m, so the initial state constraint equations start after auto-takeoff:

$$\begin{aligned} E_1(p_N, p_E, p_U) = (0, 0, 1.2)\ ,\ E_2(v_N, v_E, v_U) = (0, 0, 0)\ , \\ E_3(\phi, \theta, \psi) = (0, 0, 0)\ ,\ E_4(\omega_x, \omega_y, \omega_z) = (0, 0, 0). \end{aligned} \tag{6.4}$$

Intermediate State Constraint Equations. The intermediate state constraint is the takeoff altitude of 20 m, so the quadcopter ascends to 60 m before performing a 360° roll maneuver.

$$\begin{aligned} F_1(p_N, p_E, p_U)) = (0, 0, 60)\ ,\ F_2(v_N, v_E, v_U) = (0, 0, 0)\ , \\ F_3(\phi, \theta, \psi) = (0, 0, 0)\ ,\ F_4(\omega_x, \omega_y, \omega_z) = (0, 0, 0). \end{aligned} \tag{6.5}$$

Final State Constraint Equations. Once the quadcopter reaches an altitude of 60 m, it performs a 360° roll maneuver. Thus, the state variables are the same after completing the maneuver:

$$\begin{gathered} G_1(p_N, p_E, p_U) = (0,0,60)\ ,\ G_2(v_N, v_E, v_U) = (0,0,0)\ , \\ G_3(\phi, \theta) = (0,0)\ ,\ G_4(\omega_x, \omega_y, \omega_z) = (0,0,0). \end{gathered} \tag{6.6}$$

6.3 Explicit Guidance Solutions for the Manevuers

6.3.1 Takeoff Maneuver to Altitude of 20 m.

Choosing $p_1(t) = p_3(t) = p_5(t) = \tau$ and $p_2(t) = p_4(t) = p_6(t) = \tau^2$ yields:

$$a_x(t) = c_1\tau + c_2\tau^2 - g_x(t)\ ,\ a_y(t) = c_3\tau + c_4\tau^2 - g_y(t)\ ,\ a_z(t) = c_5\tau + c_6\tau^2 - g_z(t). \tag{6.7}$$

For the takeoff maneuver, denoted by *TO*, E Guidance provides translational acceleration commands for 8 seconds during the intermediate thrust arc immediately after the max thrust arc. The takeoff maneuver uses the constraints from Eqs. (6.1) & (6.2), which provides the boundary conditions for E Guidance. Following the E Guidance method gives the coefficients as:

$$c_{1,TO} = 0\ ,\ c_{2,TO} = 0\ ,\ c_{3,TO} = 0\ ,\ c_{4,TO} = 0\ ,\ c_{5,TO} = -1.3921\ ,\ c_{6,TO} = 0.3046. \tag{6.8}$$

The first four coefficients are zero because they act in the x and y components, which are invariant for this vertical takeoff maneuver.

6.3.2 Waypoint Maneuver

Choosing $p_1(t) = p_3(t) = p_5(t) = \tau^2$ and $p_2(t) = p_4(t) = p_6(t) = \tau^3$ yields:

$$a_x(t) = c_1\tau^2 + c_2\tau^3 - g_x(t)\ ,\ a_y(t) = c_3\tau^2 + c_4\tau^3 - g_y(t)\ ,\ a_z(t) = c_5\tau^2 + c_6\tau^3 - g_z(t). \tag{6.9}$$

For the waypoint guidance maneuver, E Guidance provides translational acceleration commands for 25 seconds. This is approximately how long it took for the DJI M100 to travel from 20 *m* above its takeoff location to the desired location, based on a flight test using DJI's standard waypoint navigation capabilities. The waypoint guidance maneuver uses the constraints from Eqs. (6.2) & (6.3), which provide the boundary conditions for E Guidance. Datcon and CsvView, free offline applications, allow users to view and analyze telemetry data and the flight path. For the waypoint guidance (denoted by *WG*) maneuver, the coefficients, c_i, are:

$$\begin{gathered} c_{1,WG} = 0.0109\ ,\ c_{2,WG} = -5.8396 \cdot 10^{-4}\ ,\ c_{3,WG} = -0.0154\ , \\ c_{4,WG} = 8.2019 \cdot 10^{-4}\ ,\ c_{5,WG} = -0.0015\ ,\ c_{6,WG} = 8.1920 \cdot 10^{-5}. \end{gathered} \tag{6.10}$$

After all the coefficients are determined, the translational acceleration commands can be computed. Then, the accelerations can be computed, which lead to integrating to get position and velocity components.

6.3.3 Roll Maneuver

Choosing $p_1(t) = p_3(t) = p_5(t) = \tau^2$ and $p_2(t) = p_4(t) = p_6(t) = \tau^3$ yields:

$$\alpha_x(t) = c_1\tau^2 + c_2\tau^3 - \frac{I_{yy} - I_{zz}}{I_{xx}}\omega_y\omega_z \ , \ \alpha_y(t) = c_3\tau^2 + c_4\tau^3 - \frac{I_{zz} - I_{xx}}{I_{yy}}\omega_x\omega_z \ , \\ \alpha_z(t) = c_5\tau^2 + c_6\tau^3 - \frac{I_{xx} - I_{yy}}{I_{zz}}\omega_x\omega_y. \tag{6.11}$$

For the guided roll maneuver, E Guidance provides translational acceleration commands for 1.626 seconds. This is approximately how long it took for the FliteTest 270 Chase Quad to complete a manual 360° roll maneuver flight test. The roll maneuver uses the constraints from Eqs. (6.5) & (6.6), which provide the boundary conditions for E Guidance. For the guided roll maneuver (denoted by RM), the coefficients, c_i, are:

$$c_{1,RM} = 53.7857 \ , \ c_{2,RM} = -44.0747 \ , \ c_{3,RM} = 0.7819 \ , \\ c_{4,RM} = -0.8010 \ , \ c_{5,RM} = 0.0489 \ , \ c_{6,RM} = -0.0501. \tag{6.12}$$

After all the coefficients are determined, the angular acceleration commands can be computed. Then, the accelerations can be computed, which lead to integrating to get position and velocity components.

6.4 Integrated Extremal Control and Explicit Guidance Results for the Maneuvers

6.4.1 Takeoff Maneuver to Altitude of 20 m.

The 60 m takeoff maneuver in Ref. [121] has similar results to the 20 m takeoff maneuver in Ref. [123, 124], so only the 20 m takeoff maneuver results are shown in this section.

Pontryagin Function. The quadcopter takeoff maneuver acts only the positive z-direction and maintains constant neutral attitude. Therefore, $\mathbf{\Omega}$, p_N, p_E, v_N, v_E, and $\boldsymbol{q}$ are constant and known: $p_N = p_E = v_N = v_E = \omega_x = \omega_y = \omega_z = q_1 = q_2 = q3 = 0$, $q_0 = 1$. Then the Pontryagin function simplifies to:

$$H = \lambda_3 v_U + \lambda_6\bigg(-g + \frac{k(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2)}{m} - \frac{k_d v_z}{m}\bigg). \tag{6.13}$$

Switching Function. Typically, one computes the switching function by $\frac{\partial H}{\partial u} = 0$ to find the factors multiplied by the control variables. Recall from Eq. (5.22) that $\lambda_6(t)$ depends on λ_3 and $\lambda_6(t_0)$. Therefore, the switching function, χ, is also a function of λ_3 and $\lambda_6(t_0)$:

$$\chi = \frac{2\lambda_6(t)k(q_0^2 - q_1^2 - q_2^2 + q_3^2)}{m}. \tag{6.14}$$

An example in Ref. [132] finds thrust for a maximum altitude of a sounding rocket, and the solution involves bang-singular-bang arcs. There are mainly two differences from their example and the analysis here. The first difference is that this work lacks a second bang arc with minimum thrust at the end of the maneuver. The second difference is that differentiating the switching function for singular arcs is not considered in this work. Overall, the takeoff maneuver uses max constant motor thrust and then switches to intermediate motor thrust through the switching function.

Velocity and Altitude Profiles of the Max Thrust Arc. The canonical equations provide a second order differential equation for altitude, i.e. $a_U = dv_U/dt = d^2U/dt^2$:

$$\frac{\partial H}{\partial \lambda_6} = \dot{v}_U = -g + \frac{4k\omega_{i,max}^2}{m} - \frac{k_d v_U}{m} \tag{6.15}$$

Solving Eq. (6.15) with the initial velocity and altitude set to zero yields:

$$\begin{aligned} v_{z,MT}(t) &= 0.6175 - 0.6175e^{-43.0899t}\,, \\ z_{MT}(t) &= 0.6175t + 0.01433e^{-43.0899t} - 0.01433\,, \end{aligned} \tag{6.16}$$

which are verified by MATLAB's *diff* and *dsolve* functions and with MT to denote max thrust. These altitude and velocity profiles are used for the max motor thrust arc of the takeoff maneuver.

Velocity and Altitude Profile Solutions of the Intermediate Thrust Arc. Fortunately, the intermediate thrust arc uses the same dynamical model as the max thrust arc. However, it is more complicated to solve than the maximum thrust arc because the $\omega_{i,int}^2$ profile is unknown. Overall, there were five approaches for solving the intermediate thrust arc, but each of them has some issues and problems.

The first approach uses kinematics with altitude and gravity to solve for time. Then, the motor spin rate profile is assumed to decrease linearly by using the linear point-slope formula using the larger root of time. Just like the motor spin rate, velocity can be determined in a similar way using the point-slope formula. Then, integrating the velocity determines the altitude. However, there is only one integration constant, the drag force is ignored, and energy is not conserved when comparing initial and final energies due to the neglected drag force, which should be considered.

Another option is to integrate the second-order differential equation for altitude, i.e. $a_z = d^2z/dt^2$. Two initial conditions are used to satisfy the initial velocity and altitude. The profile of the motor spin rate is unknown and is assumed to decrease linearly. The motor spin rate can be determined using the linear point-slope formula and choosing a final time when hovering will occur. However, the solutions do not reach the desired velocity or altitude at the end because two more boundary conditions are needed. The third attempt considered the possibility that the max thrust arcs and intermediate thrust arcs could overlap, which yields the switching time from max thrust to intermediate thrust. Unfortunately, the overlap time of the velocity arcs does not match the overlap time of the altitude arcs.

The fourth method chose a desired constant deceleration is chosen and integrated it twice to find the linear velocity and quadratic altitude profiles. The final time is determined using the selected deceleration as the change of the velocity with respect to time, where the initial velocity is the max thrust velocity at the switching time, and the final velocity is zero. Then velocity is determined using the linear point-slope formula. Integrating velocity yields the altitude but with only one integration constant.

The last method considered solving the second-order differential equation as a boundary value problem with the initial and final altitude values as the boundary values. However, the initial and final velocities are not satisfied.

Without going into excruciating details, there are some common downsides to these five methods. One issue is a lack of integration constants to satisfy the boundary values for velocities and altitudes. Another issue is that the dynamical model from $\dot{v}_U = \frac{\partial H}{\partial \lambda_6}$ includes velocity, so using the method of integrating factors to solve this differential equation yields an exponential term in the solution. Consequently, the velocity and altitude profiles become monotonic, which makes it complicated to satisfy the velocity and altitude boundary values. The third issue is that using simple linear models for deceleration and the motor spin rates significantly simplifies the problem, which produces undesired inaccuracies.

Therefore, this motivated the idea of finding another method where the boundary conditions for velocity and altitudes are satisfied without using linear assumptions for deceleration and motor spin rates. Other approaches and methods may exist, but using explicit guidance, mentioned in the previous section, satisfies the boundary conditions.

Unforced and Forced Cases of the Switching Function. Figure 6.4 shows the forced switching function for the takeoff maneuver. The gray shaded area under the curve indicates where the maximum thrust arc occurs, which is left of the vertical dotted line. The intermediate thrust occurs to the right of the vertical dotted line. Recall from Eq. (6.14) that the switching function depends on λ_6, which depends on λ_3 and $\lambda_6(t_0)$. Choosing $\lambda_3 = 43.0899$ and $\lambda_6(t_0) = 1$ shows that the switching function reaches zero at approximately 0.28 seconds. Before 0.28 seconds, it initially has small positive values.

At 0.28 seconds, the switching function becomes zero, indicating to start the intermediate thrust arc. However, after 0.28 seconds, the switching function becomes and stays negative, so minimum thrust occurs until the end of the maneuver. Therefore, the control takes its minimum

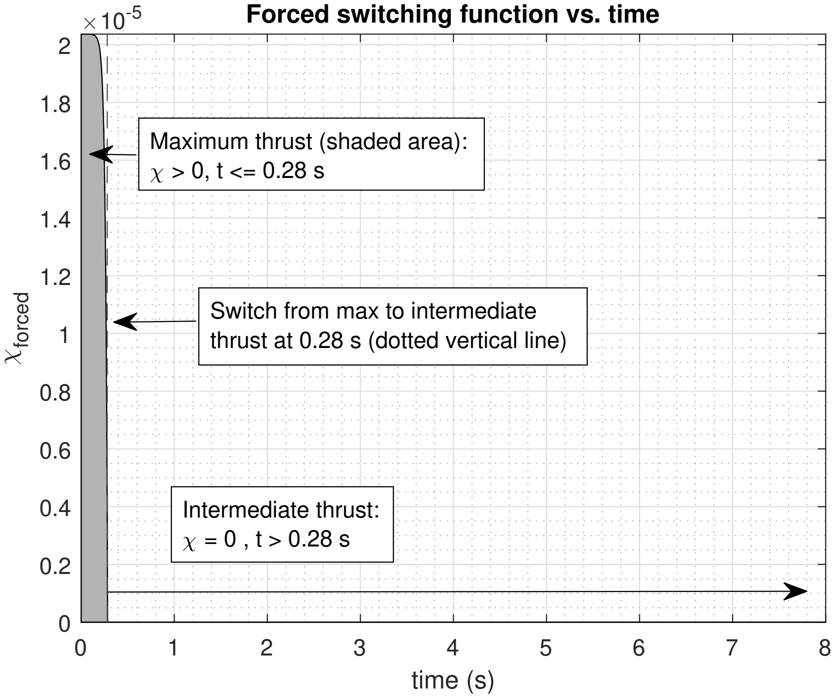


Figure 6.4: Forced Switching Function

values after 0.28 seconds. The physical interpretation implies that the quadcopter applies maximum thrust to takeoff for 0.28 seconds, applies intermediate thrust instantaneously, and then applies minimum thrust after 0.28 seconds.

Due to poor scaling, the unforced switching function is not shown, which makes it difficult to interpret the figure's overall results. However, strictly following the unforced switching function means that the quadcopter would ascend at maximum thrust for only 0.28 seconds and then apply minimum thrust until the end of the maneuver. Forcing the switching function to remain zero to use intermediate thrust until the quadcopter reaches the desired altitude will satisfy the boundary conditions. Thus, the intermediate thrust arc begins at 0.28 seconds, so the switching function is forced to remain identically zero from 0.28 seconds until the end of the takeoff maneuver. Consequently, forcing the switching function to remain zero also forces λ_6 to be identically zero.

Altitude and Velocity Plots for Max-Intermediate Thrust. Fig. 6.5 shows the simulated altitude and velocity profiles for the maximum thrust arc, while Fig. 6.6 shows the simulated altitude and velocity profiles for the intermediate thrust arc. The velocity at the junction is 0.6175 $m/s \approx 1.4$ mph, which is well within the max speed of some FPV quadcopters [133].

Figure 6.7 compares the position profiles of the standard out-of-the-box DJI PID controller against the integrated extremal control and explicit guidance HITL simulation and the flight

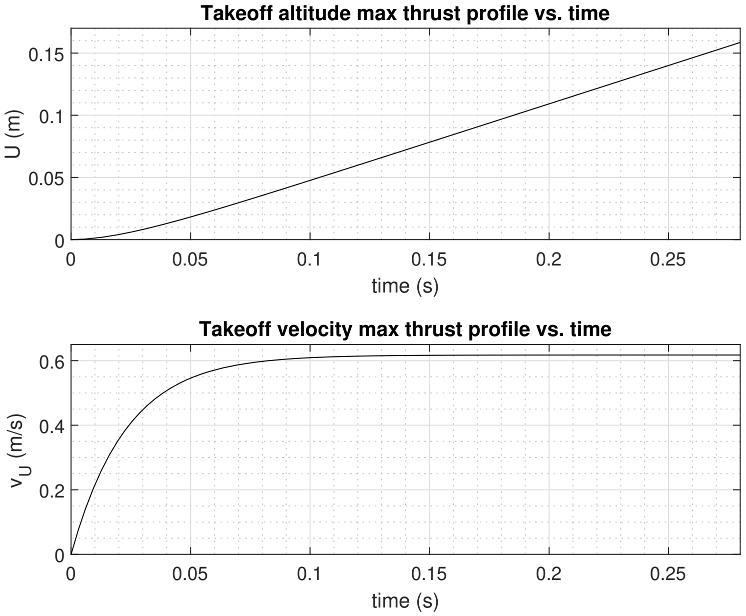


Figure 6.5: Simulated Altitude and Velocity Profiles for Maximum Thrust

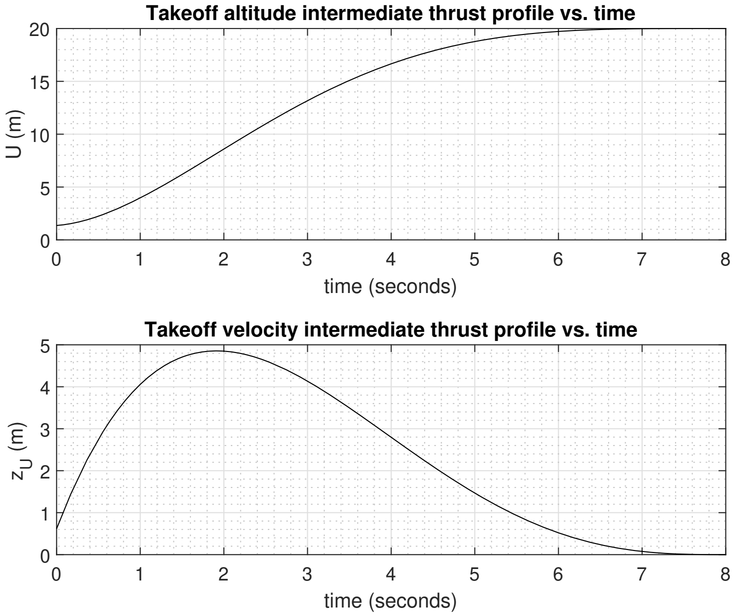


Figure 6.6: Simulated Altitude and Velocity Profiles for Intermediate Thrust

test. The PID controller finishes before the extremal controller (flight test and HITL) because the elapsed time in the DJI OSDK implementation was a counter for computing position commands instead of the real world clock time. The PID altitude slightly overshoots the desired altitude, while the integrated extremal control and explicit guidance does not. The PID altitude is generally more linear and has shorter times smoothening out at the beginning and end. Contrarily, the integrated extremal control and explicit guidance ascent is more nonlinear with a much smoother ascent. Even though winds were present during the flight test, the overall altitude profile closely matches the HITL altitude. MATLAB's *immse* function computes the mean squared error (MSE) of the HITL and flight test altitude as 0.1398 m^2. Taking the square root gives a root mean square (RMS) value of 0.3739 m. Table 6.1 shows the final position of the standard PID controller, HITL,

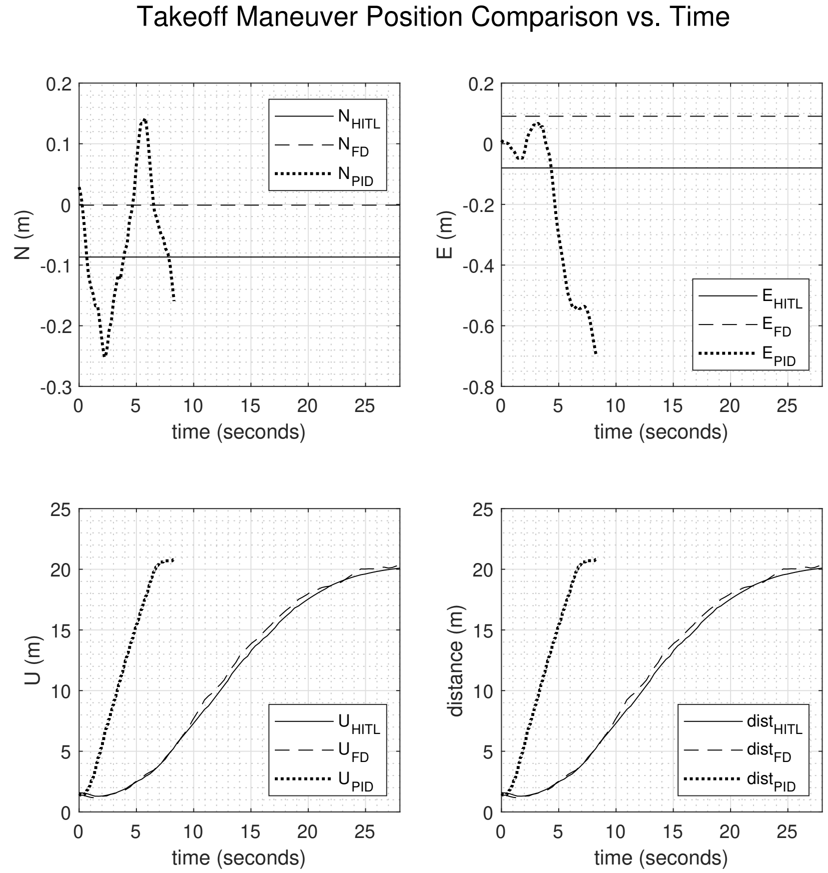


Figure 6.7: Takeoff Maneuver Position Comparison

and experimental flight tests with the final desired altitude being 20 m. As expected, the HITL is the most accurate since it lacks winds, and the experimental results are more accurate than the standard PID.

Table 6.1: Takeoff Maneuver: Final Position Comparison

	U	ΔU	% Error
PID	20.700	0.7	3.5%
HITL	20.095	0.095	0.475%
Exp	20.338	0.338	1.69%

Figure 6.8 compares the velocity profiles of the standard DJI PID controller against the integrated extremal control and explicit guidance HITL simulation and the flight test. The PID controller finishes before the extremal controller (flight test and HITL) because the elapsed time in the DJI OSDK implementation was a counter for computing position commands instead of the real world clock time. Therefore, the PID controller has a higher average velocity of 2.5 m/s as it ascends to 20 m in approximately 8 seconds, while the extremal controller has an average velocity of 0.71 m/s with an ascent time of 28 seconds. Consequently, the PID velocity profile is significantly larger than the integrated extremal control and explicit guidance velocity. The PID velocity profile resembles a classic trapezoidal velocity profile for point to point maneuvers with max acceleration (max force), constant velocity, and then max deceleration [134, 135]. This typical approach is time optimal due to max acceleration and velocity but has discontinuous acceleration profiles [134]. Contrarily, the integrated extremal control and explicit guidance method presented here has continuous acceleration through E Guidance, which yields smoother profiles for velocity and position. There are fluctuations in the integrated extremal control and explicit guidance flight test because of the wind. The MSE of the ascent velocity between the HITL and the flight test is 0.0234 $(m/s)^2$ with a RMS value of 0.1529 m/s.

Weierstrass-Erdmann Corner Conditions. The Weierstrass-Erdmann corner conditions are satisfied if the Hamiltonian and Lagrange multipliers are continuous at the corner point or junction [126]:

$$\begin{gathered} H(t^-_{switch}, \boldsymbol{s}^-, \boldsymbol{\lambda}^-, \boldsymbol{u}^-) = H(t^+_{switch}, \boldsymbol{s}^+, \boldsymbol{\lambda}^+, \boldsymbol{u}^+) \;, \\ \boldsymbol{\lambda}(t^-_{switch}) = \boldsymbol{\lambda}(t^+_{switch}) \;, \end{gathered} \tag{6.17}$$

in which t_{switch} indicates the time when the control switches from maximum to intermediate thrust. The minus superscript indicates coming from the left, i.e., the maximum thrust arc, and the plus superscript indicates coming from the right, i.e., the intermediate thrust arc. The Hamiltonian at t_{switch} for the maximum thrust arc (coming from the left) is:

$$H_{max} = \lambda_3 v_z(t^-_{switch}) \;, \tag{6.18}$$

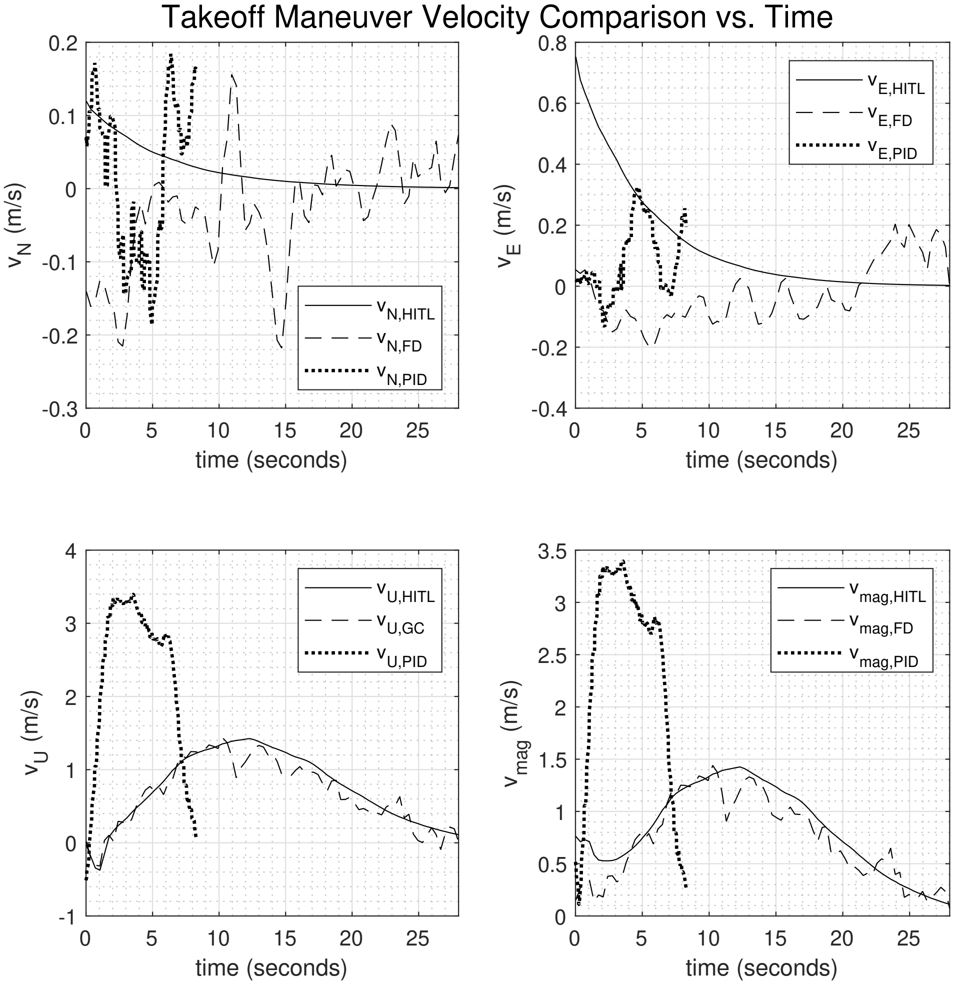


Figure 6.8: Takeoff Maneuver Velocity Comparison

where λ_6 is zero at the switching time from maximum to intermediate thrust. The Hamiltonian for the intermediate thrust arc (coming from the right) is:

$$H_{int} = \lambda_3 v_z(t^+_{switch}). \tag{6.19}$$

The velocity at the switching time for both arcs is 0.6175 m/s, and $\lambda_3 = 43.0899$ is constant for both arcs. Therefore, the Hamiltonian is continuous at the junction. Note that $\lambda_6 = 0$ at the junction and remains zero throughout the entire intermediate thrust arc, and $\lambda_3 = 43.0899$ is continuous throughout the maximum and intermediate thrust arcs. Therefore, the Lagrange multipliers are continuous at the junction, so the takeoff maneuver satisfies the Weierstrass-Erdmann corner conditions.

Fig. 6.9 shows the profiles of the Lagrange multipliers throughout the entire takeoff trajectory with maximum and then intermediate thrust. Note that they are continuous at the junction at $t_{switch} = 0.28$ s, which shows $\boldsymbol{\lambda}(t_{switch})^+ = \boldsymbol{\lambda}(t_{switch})^-$.

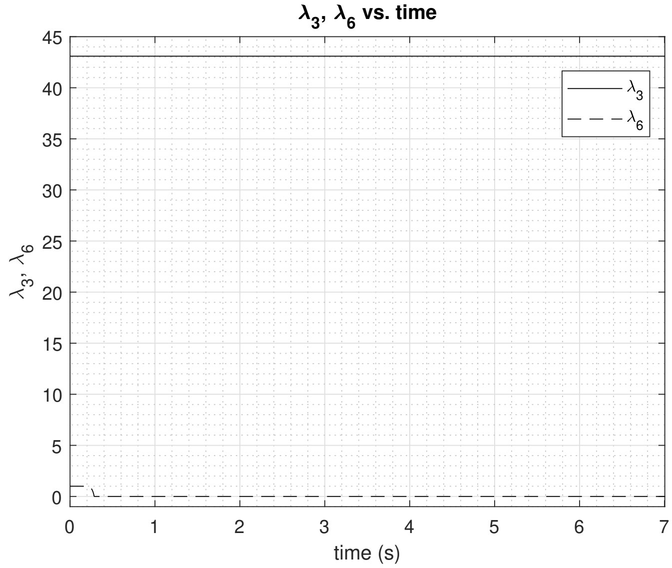


Figure 6.9: Profiles for λ_3, λ_6 of Takeoff Maneuver

Motor Spin Rates. Fig. 6.10 shows the simulated motor spin rates for the maximum thrust and intermediate thrust arcs of the takeoff maneuver. The top half of Fig. 6.10 shows the constant maximum thrust arc for the first 0.28 seconds, while the bottom half shows the intermediate motor thrust arc from 0.28 to 10.28 seconds. Due to poor scaling, it is easier to separate the arcs.

The quadcopter has a small velocity of 0.6175 $m/s \approx 1.4$ mph at the junction, and then, it must accelerate and then decelerate to stop at the desired altitude of 20 m. The quadcopter decelerates by initially slowing down its motors and then speeding up to the hovering motor spin rates. After approximately 4.6 seconds, the motor spin rates speed up and eventually reach $240,880$ $(rad/s)^2$,

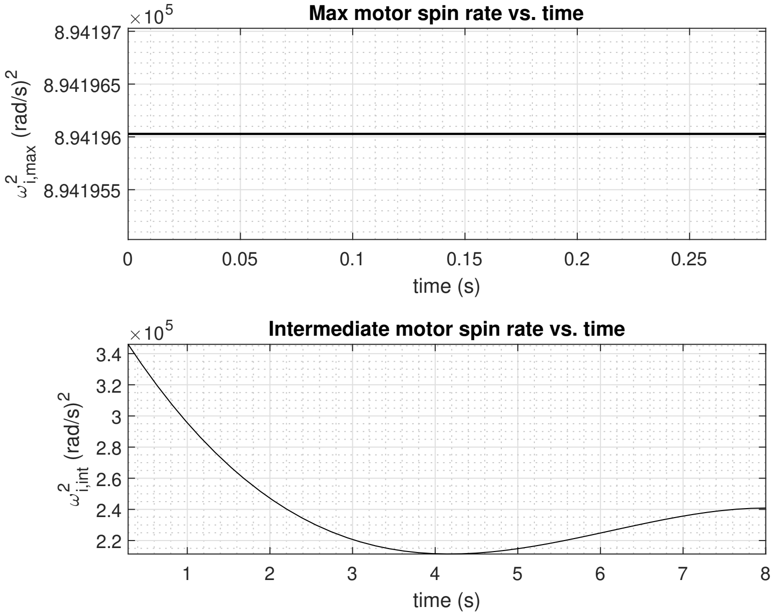


Figure 6.10: Takeoff Maneuver Simulated Motor Spin Rates

which is the motor spin rates required for hovering. The equation for the intermediate motor spin rates comes from solving $\frac{\partial H}{\partial \lambda_6} = a_z$ for ω_i^2 with a_U and v_U from E Guidance, which is denoted by subscript EG:

$$\omega_i^2 = \frac{m}{4k}\left(a_{U,EG} + g + \frac{k_d v_{U,EG}}{m}\right). \tag{6.20}$$

On the other hand, the motor spin rates from the standard PID controller flight test has a similar form but uses the telemetry data:

$$\omega_i^2 = \frac{m}{4k}\left(a_U + g + \frac{k_d v_U}{m}\right). \tag{6.21}$$

Figure 6.11 compares the standard PID controller, simulated HITL, and integrated extremal control and explicit guidance motor spin rates for the takeoff maneuver. The PID controller finishes before the extremal controller (flight test and HITL) because the elapsed time in the DJI OSDK implementation was a counter for computing position commands instead of the real world clock time. Since the takeoff maneuver is one-dimensional, there are no torques or rotational motion. Therefore, the motor spin rates are assumed to be the same, so each of the four motor spin rates has the same profile. The ascent acceleration values in the PID controller are about ten times larger than the acceleration values in the telemetry data from the integrated extremal control and explicit guidance HITL and flight test. Thus, the PID controller has much higher motor spin rates, so it consumes more energy and reduces the remaining battery capacity, which ultimately reduces

Table 6.2: Takeoff Maneuver MSE and RMS Summary

	U	v_U	ω_i^2	Normalized ω_i^2
MSE	0.1398	0.0234	2.5014e+10	0.0702
RMS	0.3739	0.1529	1.5816e+05	0.2650

the flight time. Thus, the motor spin rates based on integrated extremal control and explicit guidance are more efficient, and just like the velocity profiles, the flight test data has fluctuations due to the wind.

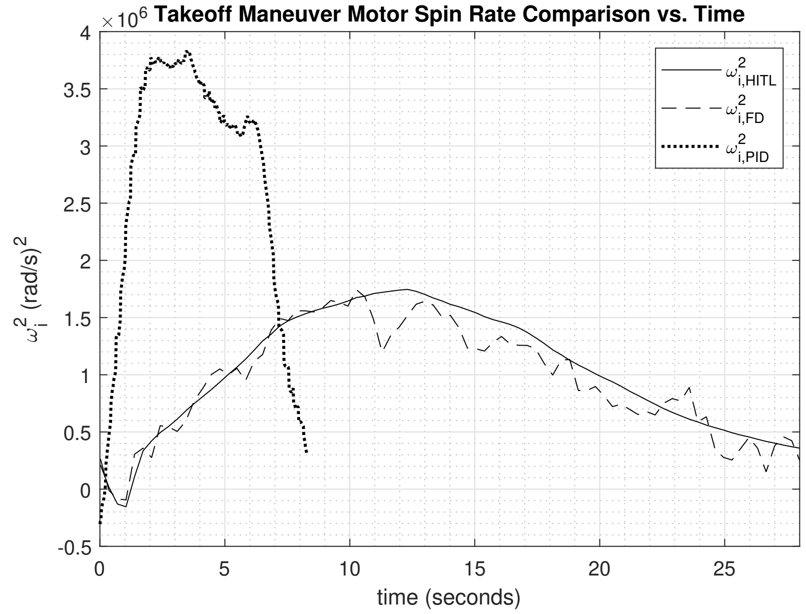


Figure 6.11: Takeoff Maneuver Motor Spin Rate Comparison

Table 6.2 summarizes the MSE and RMS of the altitude, velocity, and motor spin rate profiles of the takeoff maneuver.

Transversality Conditions. Recall from Eq. (5.8) that $dJ = 0$ to fulfill the necessary conditions of optimality. Computing transversality conditions at the initial state is not necessary since the takeoff maneuver has fixed initial states. This means that Γ_{t_0} and $\Gamma_{\boldsymbol{s}_0}$ are not needed, so the first and third terms outside the integral in Eq. (5.8) vanish:

$$\begin{aligned} dJ_{TO} = {} & (\Gamma_{\boldsymbol{s}_{TO,f}} - \boldsymbol{\lambda}_{TO,f}^T)^T d\boldsymbol{s}_f + (\Gamma_{t_{TO,f}} + H_{TO,f})dt_f \\ & + \int_{t_0}^{t_f} \left[(H_{\boldsymbol{s}} + \dot{\boldsymbol{\lambda}}^T)^T \delta \boldsymbol{s} + (H_{\boldsymbol{\lambda}} - \dot{\boldsymbol{s}}^T)^T \delta \boldsymbol{\lambda} + H_{\boldsymbol{u}}^T \delta \boldsymbol{u} + H_{\boldsymbol{\alpha}}^T \delta \alpha \right] dt \, , \end{aligned} \tag{6.22}$$

with TO to denote takeoff. Recall from the intermediate state constraints in Eq. (5.2) that ψ is the only free variable and not constrained. Therefore, the Lagrange multipliers associated with ψ must be considered at the end of the takeoff maneuver.

Recall that λ_7, λ_8, and λ_9 are the relevant Lagrange multipliers for the first three quaternions, and Eq. (2.27) shows that the quaternions depend on ψ. Thus, Eqs. (5.39) and (5.40) provide the closed-form solutions for λ_7, λ_8, and λ_9 even though their state variables are constant throughout the takeoff maneuver, i.e. $q_0 = 1, q_1 = q_2 = 0$.

However, the quadcopter is hovering at 20 m and in the variable motor thrust case at the end of the takeoff maneuver. Recall that $\lambda_7 = \lambda_8 = \lambda_9 = 0$ by algebraically solving the differential equations in Eq. (5.24). Since $\lambda_7 = \lambda_8 = \lambda_9 = 0$, $J = t_f - t_0$, and $\boldsymbol{F}$ does not depend on t_f, the two Γ terms outside the integral in Eq. (6.22) are:

$$\begin{aligned} \Gamma_{t_f} &= \epsilon \frac{\partial J}{\partial t_f} + \boldsymbol{\beta_{TO}}^T \frac{\partial \boldsymbol{F}}{\partial t_f} = \epsilon \,, \\ \Gamma_{\boldsymbol{s}_f} &= \epsilon \frac{\partial J}{\partial \boldsymbol{s}_f} + \boldsymbol{\beta_{TO}}^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{s}_f} = \boldsymbol{\beta}_{TO}^T. \end{aligned} \tag{6.23}$$

Satisfying the First-Order Necessary Conditions of Optimality. The first-order necessary conditions of optimality are satisfied $dJ = 0$. Therefore, all the terms in Eq. (6.22) must be zero:

$$\begin{aligned} &\Gamma_{t_{TO,f}} = \epsilon = -H_{TO,f} \,,\ \Gamma_{\boldsymbol{s}_{TO,f}} = \boldsymbol{\beta_{TO}} = \lambda_{TO,f}^T = \begin{bmatrix} \lambda_7(t_{TO,f}) \\ \lambda_8(t_{TO,f}) \\ \lambda_9(t_{TO,f}) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \,, \\ &H_{\boldsymbol{s}} = -\dot{\boldsymbol{\lambda}}^T \,,\ H_{\boldsymbol{\lambda}} = \dot{\boldsymbol{s}}^T \,,\ H_{\boldsymbol{u}} = 0 \,,\ H_{\boldsymbol{\alpha}} = 0. \end{aligned} \tag{6.24}$$

The Hamiltonian at the end of the takeoff maneuver, $H_{TO,f}$, is zero since $v_U(t_f) = \lambda_6(t_f) = 0$. Therefore, $\epsilon = 0$, so $\Gamma_{t_{TO,f}} = -H_{TO,f} = 0$. Since the quadcopter ends the takeoff maneuver in the variable motor thrust case with $\lambda_7 = \lambda_8 = \lambda_9 = 0$, $\Gamma_{\boldsymbol{s}_{TO,f}} = 0$. The canonical equations and local optimality equations satisfy the remaining four equations in Eq. (6.24), which yield zeros too. Since all the terms in Eq. (6.22) are zero, the takeoff maneuver has an extremal trajectory. Recall that second-order conditions are not considered, so at best, only an extremal trajectory can be currently claimed for the takeoff maneuver.

6.4.2 Waypoint Maneuver

CsvView's Geoplayer app provides a 2D view of the trajectory based on the telemetry data, as shown by the black line in Fig. 6.12.

Acceleration Profiles. Fig. 6.13 shows the accelerations and commanded accelerations. The PID a_U measurements fluctuate around 10 m/s^2 due to gravity, and the a_N and a_E measurements

Figure 6.12: CsvView Geoplayer Waypoint Guidance Trajectory

have smaller fluctuations. On the other hand, the E Guidance accelerations are smoother and lack fluctuations.

Velocity Profiles. Fig. 6.14 compares the PID flight test velocity with the E Guidance velocity. The top half of Fig. 6.14 shows the guided velocity, while the bottom half shows the velocity from the standard PID controller. The quadratic and cubic E Guidance polynomials cause the E Guidance velocity to be nonlinear, specifically a quartic because it integrates a cubic acceleration. Contrarily, the PID velocity is nearly constant throughout the entire flight, but it decreases quickly towards the end at a faster rate than the velocity from E Guidance. Overall, the E Guidance velocity profile is much smoother and gradually slows down towards the target point due to its nonlinear form.

Figure 6.15 compares the experimental, HITL, and PID velocity profiles. The PID controller finishes before the extremal controller (flight test and HITL) because the elapsed time in the DJI OSDK implementation was a counter for computing position commands instead of the real world clock time (68 seconds). Similar phenomenon occurs in the PID velocity profile with the trapezoidal velocity profile, which resembles the takeoff maneuver's PID velocity profile with max acceleration and deceleration shown in Fig. 6.8. Same as before, the extremal control and E Guidance velocities are smoother due to the continuous commanded accelerations from E Guidance. There is an unexpected small jump in $v_{E,HITL}$ at the beginning, which is unusual since it does not occur in the experimental results. Despite that initial jump, the MSE and RMS values are still small but larger than the v_N and v_U MSE and RMS values due to this perturbation. Table 6.3 shows the MSE and RMS between the HITL and experimental velocity results.

Position Profiles. The bottom half of Fig. 6.16 shows the PID positions. The top half shows the E Guidance position profiles, which is nonlinear and smoother than the linear PID position profile. Both reach the desired position at the end of the maneuver, but the p_E and p_U positions in E Guidance converge to the desired values faster than the PID positions.

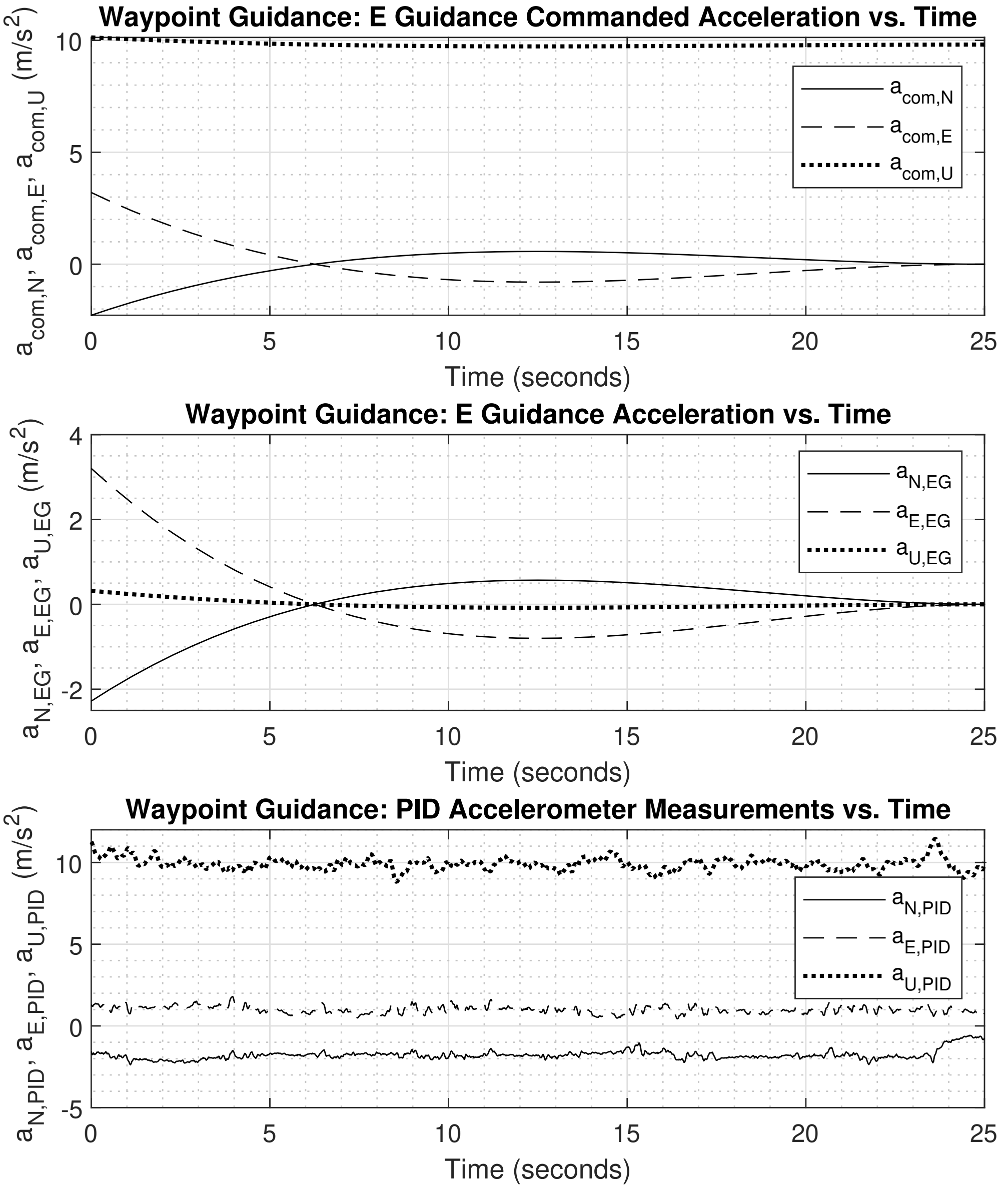


Figure 6.13: E Guidance Acceleration & PID Comparison vs. Time

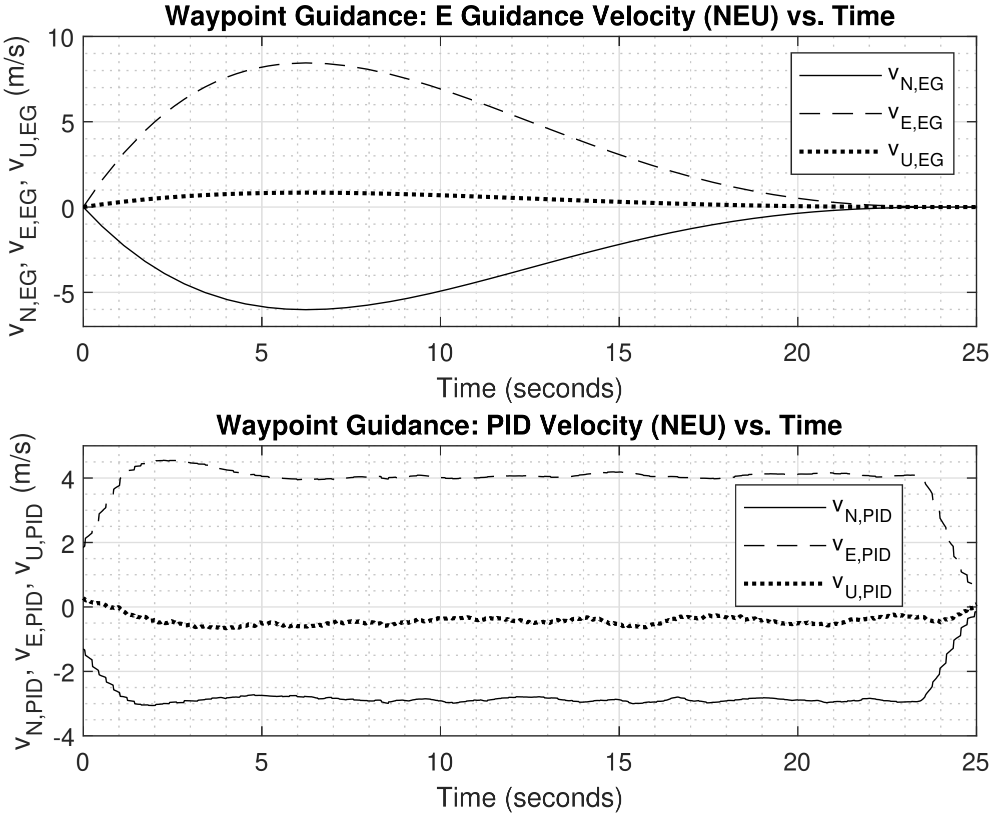


Figure 6.14: E Guidance & PID Velocity Comparison vs. Time

Table 6.3: Experiment and HITL MSE and RMS Velocity Summary

	v_U	v_E	v_U
MSE	0.0778	0.6556	0.0060
RMS	0.2789	0.8097	0.0772

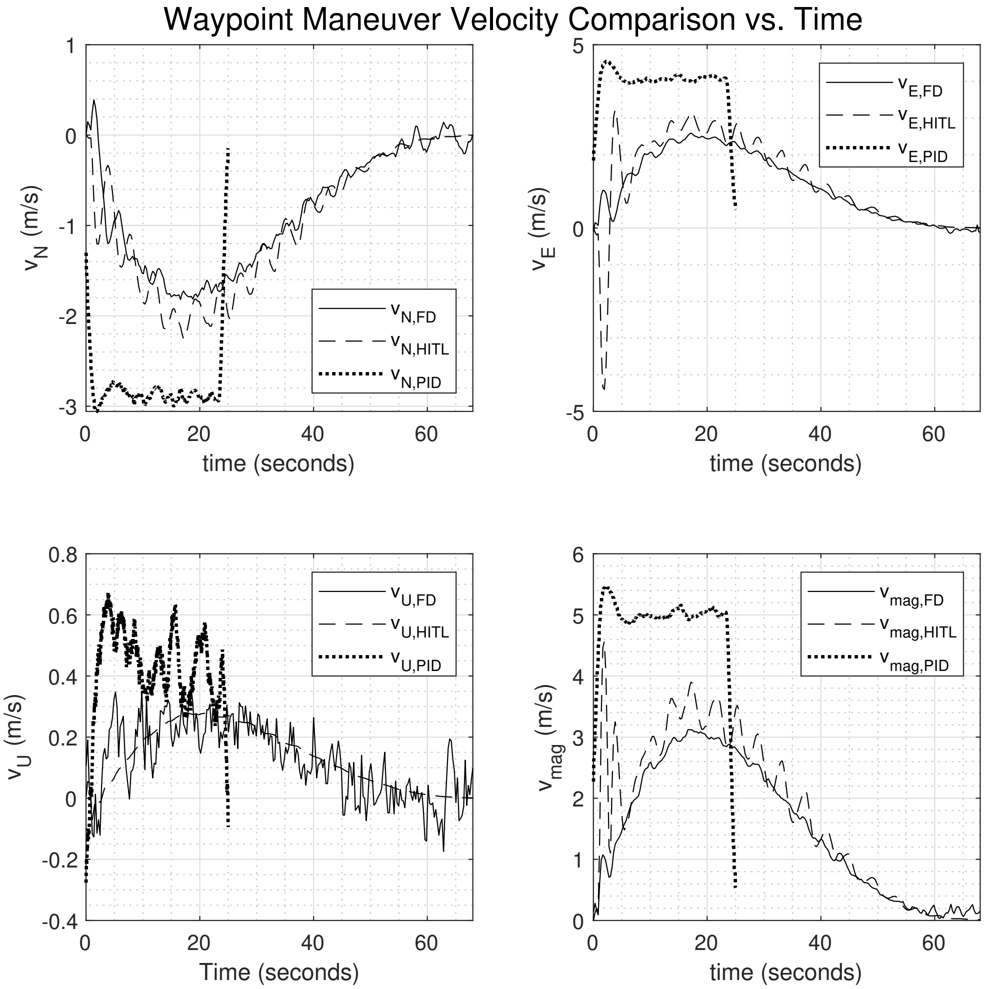


Figure 6.15: Experimental, HITL, & PID Velocity Comparison vs. Time

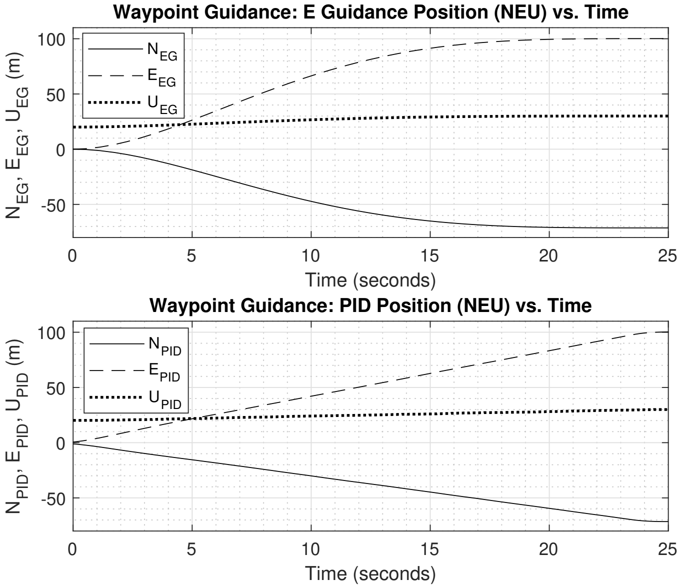
The DJI OSDK function, *positionAndYawCtrl*, uses the p_U commands from E Guidance, but the p_N and p_E commands in *positionAndYawCtrl* require feedback of the remaining distance to travel. Thus, the equations for the north and east position commands are:

$$\begin{aligned} p_{N,cmd} &= p_{N,EG} - (p_{N,dist} + p_{N,target}) \, , \\ p_{E,cmd} &= p_{E,EG} - (p_{E,dist} + p_{E,target}) \, , \end{aligned}$$

where $p_{N,EG}$ and $p_{E,EG}$ are the north and east position commands computed from E Guidance, $p_{N,target}$ and $p_{E,target}$ are the target positions from the start point. The $p_{N,dist}$ and $p_{E,dist}$ are the north and east remaining distances to the target computed by:

$$\begin{aligned} p_{N,dist} &= R_E(lat_{rad} - latOr_{rad}) \, , \\ p_{E,dist} &= R_E \cos lat_{rad}(long_{rad} - longOr_{rad}) \, , \end{aligned}$$

where R_E is the Earth's radius in meters, lat_{rad} and $long_{rad}$ are the current latitude and longitude values in radians from DJI M100's GPS sensor, and $latOr_{rad}$ and $longOr_{rad}$ are the fixed origin latitude and longitude values in radians.


Waypoint Guidance: E Guidance Position (NEU) vs. Time

Waypoint Guidance: PID Position (NEU) vs. Time

Figure 6.16: E Guidance & PID Position Comparison vs. Time

Table 6.4 compares the final NEU coordinates of the PID controller, HITL, and experimental flight tests, and the final NEU coordinates are $(-71.2845, 100.1211, 30)$.

Table 6.5 compares the error of the final NEU coordinates of the PID controller, HITL, and experimental flight tests. As expected, the HITL is the most accurate due to lack of wind, and

Table 6.4: Final Position Comparison

	N	E	U
PID	-71.4577	100.2946	30.2450
HITL	-71.1867	99.9846	29.9968
Exp	-70.9543	99.5425	30.1089

only the experimental final altitude is more accurate than the PID controller. The wind speed was higher on the day of the experimental flight test than the day of the standard DJI PID flight test, which explains the lower error of the standard PID.

Table 6.5: Final Position Error Comparison

	ΔN	ΔE	ΔU	% N Error	% E Error	% U Error
PID	0.1732	0.1735	0.245	0.2430%	0.1733%	0.8167%
HITL	0.09777	0.1365	0.003130	0.1372%	0.1363%	0.01043%
Exp	0.3302	0.5786	0.1089	0.4632%	0.5779%	0.363%

A future consideration involves reducing T_{go} such that the E Guidance trajectory reaches the target sooner and to have T_{go} in the OSDK implementation closely match the real world clock time. Reducing T_{go} causes the acceleration profiles to yield larger values. Consequently, the motors spin faster to generate these larger accelerations, so the limiting factors for reducing T_{go} are the max motor spin rates from Eqs. (5.4)-(5.5) and the DJI M100 max velocity constraints mentioned earlier in subsection 5.1. If T_{go} decreases too much and yields a motor spin rate greater than the maximum motor spin rate or a velocity greater than the DJI M100 max velocities, then this T_{go} is too short and deemed physically impossible. Minimizing time to reach the target point would be necessary for situations such as emergencies, search-and-rescue, or delivering packages as quickly as possible. On the other hand, accuracy is more critical over time optimization for applications such as photography or power line inspections.

Figure 6.17 compares the experimental, HITL, and PID position profiles. As expected, the experimental and HITL position profiles are smoother and nonlinear than the generally linear PID postion profiles. The PID controller finishes before the extremal controller (flight test and HITL) because the elapsed time in the DJI OSDK implementation was a counter for computing position commands instead of the real world clock time. Similar to previous results, even though the HITL lacked winds, the experimental position results closely match the HITL positions. Table 6.6 shows the MSE and RMS between the HITL and experimental position results.

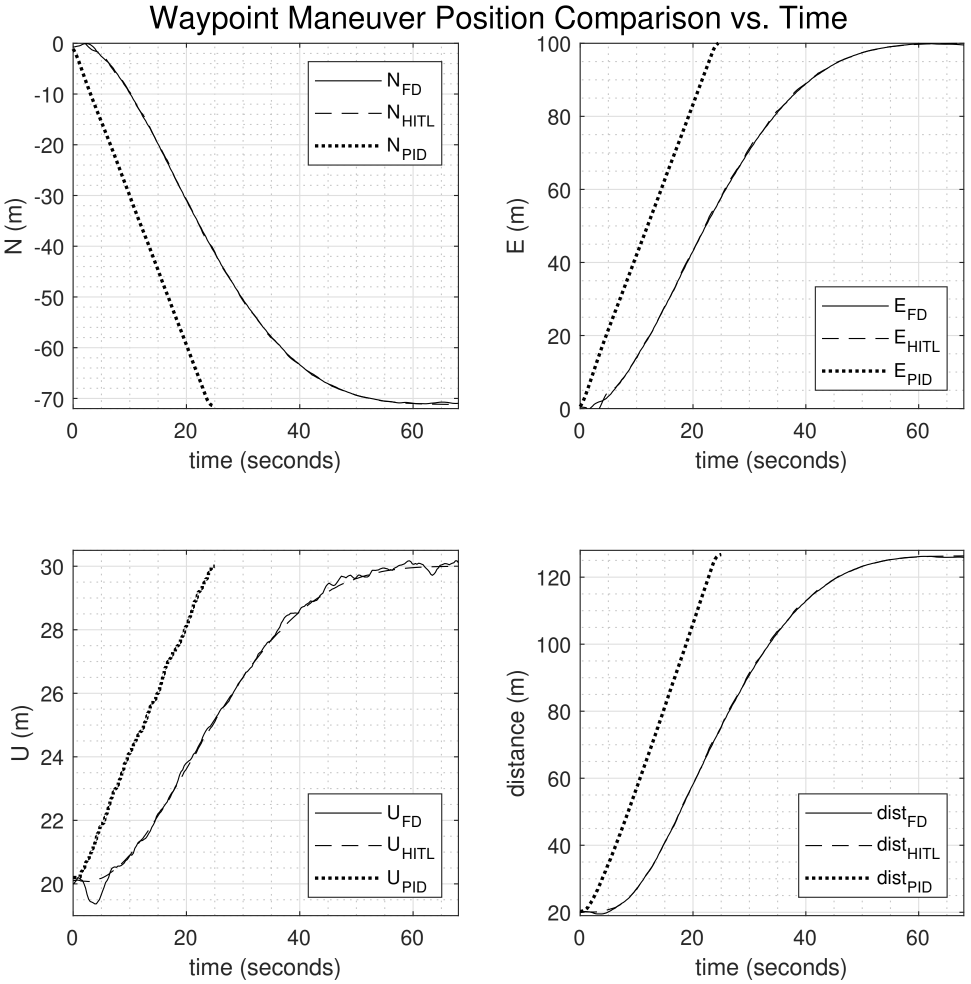


Figure 6.17: Experimental, HITL, & PID Position Comparison vs. Time

Table 6.6: Experiment and HITL MSE and RMS Position Summary

	U	E	U
MSE	0.1047	1.1501	0.0289
RMS	0.3235	1.0724	0.1699

Motor Spin Rates. Figure 6.18 compares the motor spin rates of the experimental, HITL, and PID results. The PID controller finishes before the extremal controller (flight test and HITL) because the elapsed time in the DJI OSDK implementation was a counter for computing position commands instead of the real world clock time. As expected, the PID motor spin rate values are larger and consume more energy. Therefore, the integrated extremal control and E Guidance motor spin rates are more efficient and prolong battery life, which provides longer flight time. The experimental motor spin rates fluctuate due to the wind, while the HITL motor spin rates are smooth due to the lack of wind but with only fluctuations in the beginning. Table 6.7 shows the MSE and RMS based on unnormalized motor spin rates, while Table 6.8 shows the MSE and RMS based on normalized motor spin rates with subscript n to denote "normalized."

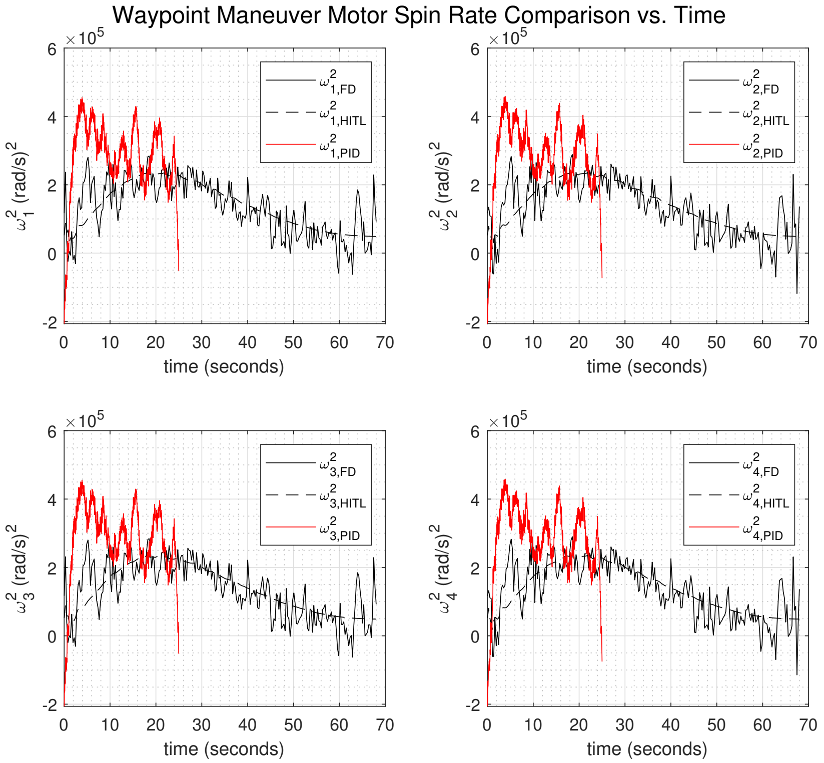


Figure 6.18: Experimental, HITL, & PID Motor Spin Rate Comparison vs. Time

The PID motor spin rates are computed from Eqs. (2.20) and (2.24) based on the quaternion,

Table 6.7: Experiment and HITL MSE and RMS Unnormalized Motor Spin Rate Summary

	ω_1^2	ω_2^2	ω_3^2	ω_4^2
MSE	7.0426e+10	7.1181e+10	7.0011e+10	7.1360e+10
RMS	2.6538e+05	2.6680e+05	2.6460e+05	2.6713e+05

Table 6.8: Experiment and HITL MSE and RMS Normalized Motor Spin Rate Summary

	$\omega_{1,n}^2$	$\omega_{2,n}^2$	$\omega_{3,n}^2$	$\omega_{4,n}^2$
MSE	0.4674	0.4641	0.4649	0.4659
RMS	0.6836	0.6812	0.6819	0.6826

accelerometer, angular velocity, and velocity telemetry data:

$$\begin{aligned}
\omega_1^2 &= \left(\frac{m}{k}\right)\frac{a_U + k_{dz}v_U}{q_0^2 - q_1^2 - q_2^2 + q_3^2} + \frac{\tau_\psi}{4b} + \frac{\tau_\phi}{2\ell k}, \\
\omega_2^2 &= \left(\frac{m}{k}\right)\frac{a_U + k_{dz}v_U}{q_0^2 - q_1^2 - q_2^2 + q_3^2} - \frac{\tau_\psi}{4b} + \frac{\tau_\theta}{2\ell k}, \\
\omega_3^2 &= \left(\frac{m}{k}\right)\frac{a_U + k_{dz}v_U}{q_0^2 - q_1^2 - q_2^2 + q_3^2} + \frac{\tau_\psi}{4b} - \frac{\tau_\phi}{2\ell k}, \\
\omega_4^2 &= \left(\frac{m}{k}\right)\frac{a_U + k_{dz}v_U}{q_0^2 - q_1^2 - q_2^2 + q_3^2} - \frac{\tau_\psi}{4b} - \frac{\tau_\theta}{2\ell k}.
\end{aligned} \tag{6.25}$$

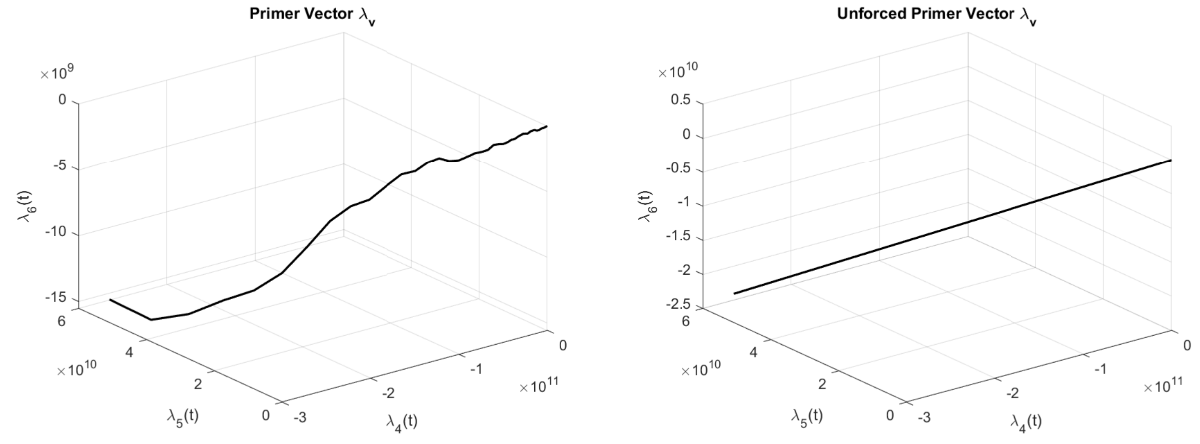
Similar formulas exist for the motor spin rates based on E Guidance.

Primer Vector.

Fig. 6.19a shows $\boldsymbol{\lambda}_v$, the primer vector, for the waypoint guidance maneuver. The profile is generally linear and smooth because λ_6 is solved in terms of the λ_4 and λ_5 terms. Setting λ_6 as a function of λ_4 and λ_5 ensures $\frac{\partial H}{\partial \omega_i} = 0\ \forall t$:

$$\lambda_6(t) = \frac{-\lambda_4(t)(q_0q_2 + q_1q_3) - \lambda_5(t)(q_2q_3 + q_0q_1)}{q_0^2 - q_1^2 - q_2^2 + q_3^2}. \tag{6.26}$$

Ultimately, Eq. (6.26) must be satisfied to generate an extremal trajectory, which makes the guided trajectory an extremal trajectory. Not constraining λ_6 would produce a feasible trajectory instead of an extremal trajectory. Fig. 6.19b shows $\boldsymbol{\lambda}_v$ for this unforced λ_6, which generates a feasible trajectory because $\frac{\partial H}{\partial \omega_i} \neq 0$. The primer vector is a straight line in rocket dynamics for uniform gravity fields [126, 136], so the straight line for the unforced primer vector in Fig. 6.19b validates



(a) Primer Vector

(b) Unforced Primer Vector

Figure 6.19: Comparing Primer Vector: Forced and Unforced

primer vector theory for rocket dynamics in a uniform gravity field.

Transversality Conditions. The waypoint guidance maneuver has free initial and final ψ, so one must consider transversality conditions at the initial and final states. Since the waypoint guidance maneuver occurs after the takeoff maneuver, the initial transversality conditions of the waypoint guidance maneuver become the final transversality conditions of the takeoff maneuver. The waypoint guidance maneuver always remains in the variable motor thrust case, so $\lambda_7 = \lambda_8 = \lambda_9 = 0$. Since $\lambda_7 = \lambda_8 = \lambda_9 = 0$, $J = t_f - t_0$, and $\boldsymbol{G}$ does not depend on t_f, the Γ terms outside the integral in Eq. (6.22) become:

$$\Gamma_{t_{WG,0}} = -\epsilon \;,\; \Gamma_{t_{WG,f}} = \epsilon \;,\; \Gamma_{s_{WG,0}} = \epsilon \;,\; \Gamma_{\boldsymbol{s}_f} = \boldsymbol{\beta^T}_{WG}.$$

Satisfying the First-Order Necessary Conditions of Optimality. The first-order necessary conditions of optimality are satisfied if $dJ = 0$. For the waypoint guidance maneuver, the first differential is:

$$\begin{aligned} dJ_{WG} = {} & (\Gamma_{\boldsymbol{s}_{WG,0}} + \boldsymbol{\lambda}^T_{WG,0})^T d\boldsymbol{s}_0 + (\Gamma_{\boldsymbol{s}_{WG,f}} - \boldsymbol{\lambda}^T_{WG,f})^T d\boldsymbol{s}_f \\ & + (\Gamma_{t_{WG,0}} - H_{WG,0})dt_0 + (\Gamma_{t_{WG,f}} + H_{WG,f})dt_f \\ & + \int_{t_0}^{t_f} \left[(H_{\boldsymbol{s}} + \dot{\boldsymbol{\lambda}}^T)^T \delta\boldsymbol{s} + (H_{\boldsymbol{\lambda}} - \dot{\boldsymbol{s}}^T)^T \delta\boldsymbol{\lambda} + H^T_{\boldsymbol{u}} \delta\boldsymbol{u} + H^T_{\boldsymbol{\alpha}} \delta\alpha \right] dt. \end{aligned}$$

Therefore, all the terms in Eq. (5.8) must be zero:

$$\begin{aligned}
&\Gamma_{t_{WG,0}} = \epsilon = H_{WG,0}\ ,\ \Gamma_{\boldsymbol{s}_{WG,0}} = \boldsymbol{\mu}_{WG} = \boldsymbol{\lambda}^T_{WG,0} = \begin{bmatrix}\lambda_7(t_{WG,0})\\ \lambda_8(t_{WG,0})\\ \lambda_9(t_{WG,0})\end{bmatrix} = \begin{bmatrix}0\\0\\0\end{bmatrix}\ ,\\
&\Gamma_{t_{WG,f}} = \epsilon = -H_{WG,f}\ ,\ \Gamma_{\boldsymbol{s}_{WG,f}} = \boldsymbol{\beta}_{WG} = \boldsymbol{\lambda}^T_{WG,f} = \begin{bmatrix}\lambda_7(t_{WG,f})\\ \lambda_8(t_{WG,f})\\ \lambda_9(t_{WG,f})\end{bmatrix} = \begin{bmatrix}0\\0\\0\end{bmatrix}\ ,\\
&\quad H_{\boldsymbol{s}} = -\dot{\boldsymbol{\lambda}}^T\ ,\ H_{\boldsymbol{\lambda}} = \dot{\boldsymbol{s}}^T\ ,\ H_{\boldsymbol{u}} = 0\ ,\ H_{\boldsymbol{\alpha}} = 0.
\end{aligned}$$

From the takeoff maneuver, $H_{TO,f} = H_{WG,0} = \epsilon = 0$. The Hamiltonian at the end of the waypoint guidance maneuver, $H_{WG,f}$, is also zero. The quadcopter ends the waypoint guidance maneuver with a heading of $\psi = 125°$, which yields $q_0 = 0.462, q_1 = q_2 = 0, q_3 = 0.887$. The velocity components are zero, and $\lambda_6 = 0$ from its dependence on λ_4 and λ_5. Therefore, the Hamiltonian is zero, as expected. The canonical equations and local optimality equations satisfy the remaining four equations in Eq. (6.4.2). Since all the terms in Eq. (6.4.2) are zero, the waypoint guidance maneuver has an extremal trajectory, which means that the takeoff and waypoint guidance maneuvers have extremal trajectories. The extremal control flight test data had an initial battery voltage at 86% and a final battery voltage of 77%, while the default PID flight test data had an initial voltage of 76% and a final battery voltage of 74%. Unfortunately, this is not an accurate apples to apples comparison because battery voltage tends to decrease exponentially, so future flight tests should start with the same battery voltage for a thorough comparison. However, one can assume that the extremal controller would yield smaller voltage decrease since it moves slower and smoother than the PID controller. Future studies may include considering second-order conditions to determine if these maneuvers have optimal trajectories.

6.4.3 Roll Maneuver

This 360° roll maneuver follows the results of Ref. [120, 121] with the FliteTest 270 Chase Quad. Betaflight Blackbox provides real-time flight data for quadcopter and fixed-wing UAVs. A FliteTest 270 Chase Quad flew manually on October 14, 2017 at Ching Field with the Blackbox recording function turned on to record flight data.

Euler Angles. Figure 6.20 shows the original Betaflight Blackbox Euler angle data, where the roll angle switches from -180° to 180° because Blackbox constrains the Euler angles from $-\pi$ to π. To help see this clearer, roll values beyond $-\pi$ decrease by 2π, which allows the roll angle profile to transition smoothly and eliminates a vertical spike at the transition from -180° to 180°.

Figure 6.21 compares the profiles of the simulated E Guidance Euler angles and the PID Betaflight Blackbox flight test data. The top half shows the simulated attitude E Guidance Euler angles, while

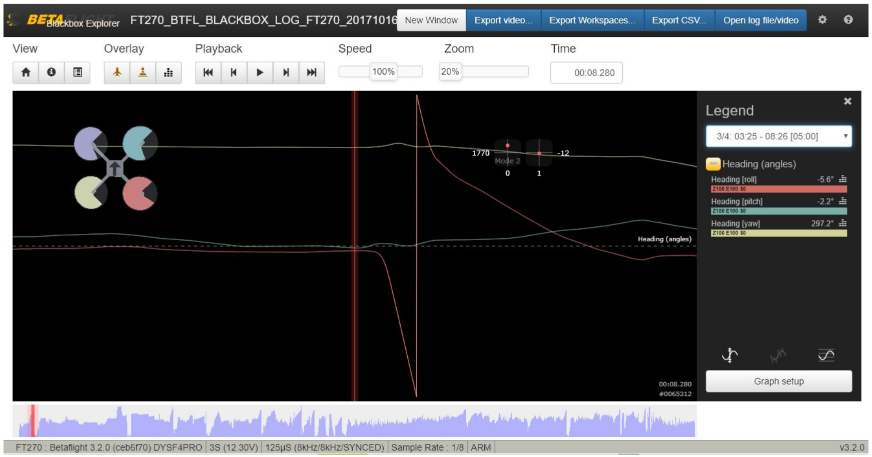


Figure 6.20: Betaflight Blackbox Roll Maneuver Euler Angles

the bottom half shows the Euler angles from the PID flight test telemetry data. The simulated attitude E Guidance Euler angles are smoother than the PID flight test data, but both reach the target point and successfully perform the roll maneuver. The PID flight test data had a final roll angle of $-6.2837\ rad$, while E Guidance had a final roll angle of $-6.2831\ rad$. Even though it is harder to determine in the E guidance roll angle, both seem to have inflection points at around 0.5 seconds.

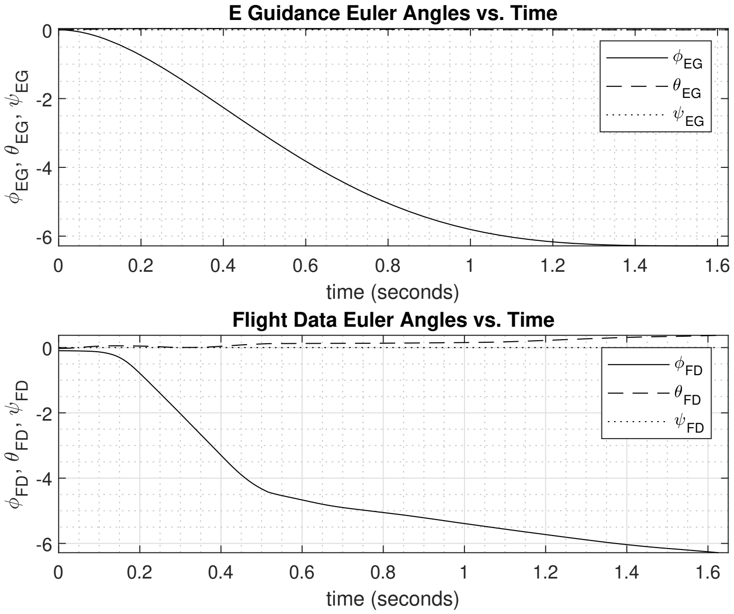


Figure 6.21: Roll Maneuver Euler Angle Comparison

Angular Velocity. Figure 6.22 compares the angular velocity profiles of the simulated E Guidance and the PID flight test data. The E Guidance angular velocity profile is much smoother than the PID flight test data because the pilot has to provide continuous remote controller inputs to keep the quadcopter stabilized while performing the roll maneuver.

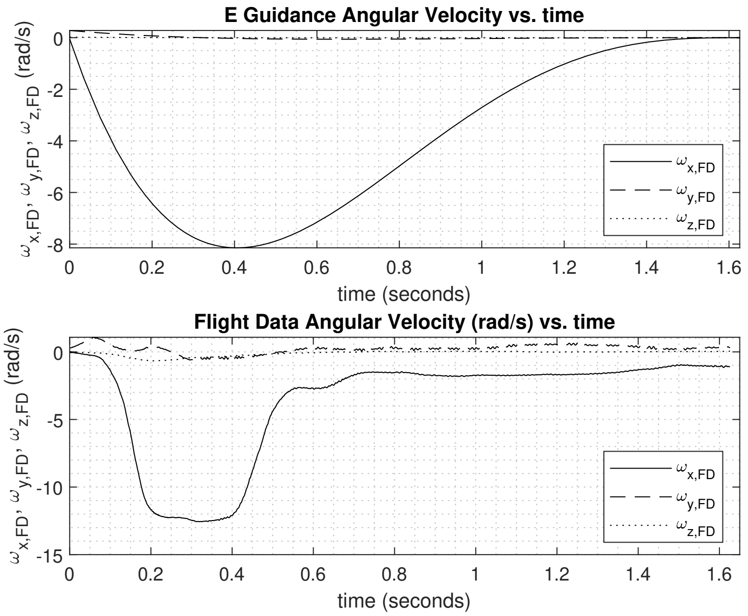


Figure 6.22: Roll Maneuver Angular Velocity Comparison

Torques. Figure 6.23 compares the torques of the simulated E Guidance and the PID flight test data. The E Guidance torques are much smoother than the PID flight test data because the pilot has to provide continuous remote controller inputs to keep the quadcopter stabilized while performing the roll maneuver.

Motor Spin Rates. Figure 6.24 compares the motor spin rates of the simulated E Guidance and the PID flight test data. The motor spin rates in Figure 6.24 mainly depend on the torques and quaternions, which come from the angular acceleration and Euler angles from E guidance:

$$\begin{aligned}
\omega_1^2 &= \frac{mg}{4k(q_0^2 - q_1^2 - q_2^2 + q_3^2)} + \frac{\tau_\psi}{4b} + \frac{\tau_\phi}{2\ell k}\,, \\
\omega_2^2 &= \frac{mg}{4k(q_0^2 - q_1^2 - q_2^2 + q_3^2)} - \frac{\tau_\psi}{4b} + \frac{\tau_\theta}{2\ell k}\,, \\
\omega_3^2 &= \frac{mg}{4k(q_0^2 - q_1^2 - q_2^2 + q_3^2)} + \frac{\tau_\psi}{4b} - \frac{\tau_\phi}{2\ell k}\,, \\
\omega_4^2 &= \frac{mg}{4k(q_0^2 - q_1^2 - q_2^2 + q_3^2)} - \frac{\tau_\psi}{4b} - \frac{\tau_\theta}{2\ell k}.
\end{aligned} \tag{6.27}$$

There are small initial torques that generate a smooth roll maneuver in profiles for the E guidance motor spin rates. Then, the motor spin rates switch to negative thrust to decelerate and then switch to stabilize back to neutral attitude. This switch occurs from the roll maneuver quaternions, which

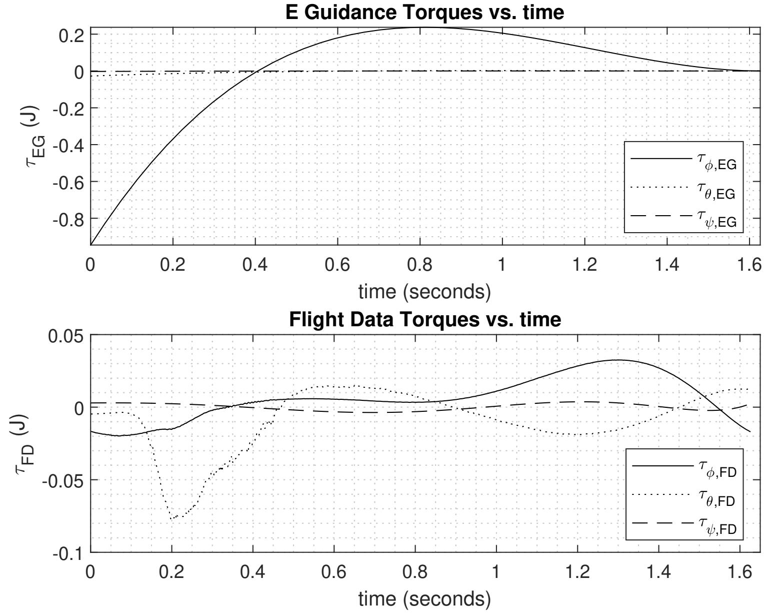


Figure 6.23: Roll Maneuver Torque Comparison

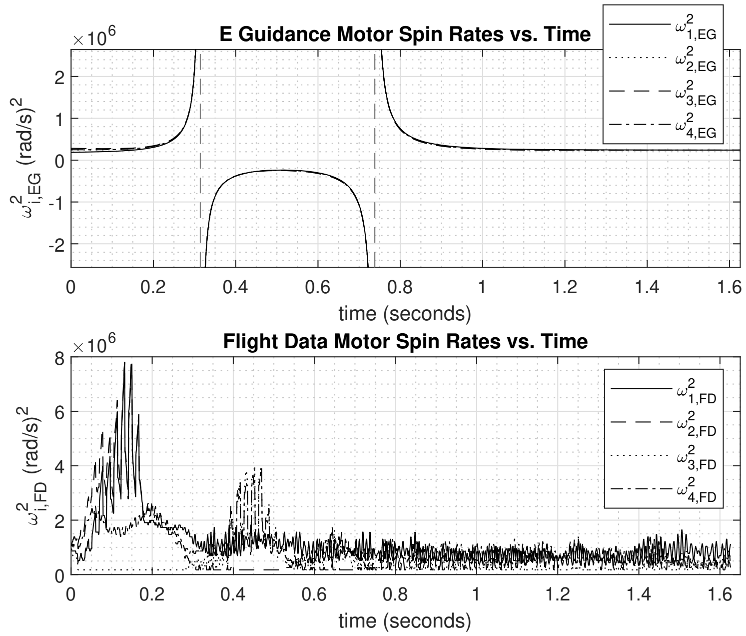


Figure 6.24: Roll Maneuver Motor Spin Rate Comparison

leads to a sinusoidal response with respect to time. The manual PID flight data motor spin rates has numerous throttle fluctuations because the pilot makes continuous adjustments on the remote control sticks to generate the roll maneuver. Even though the guided roll maneuver is smoother, the pilot has faster reflexes but continues to fight to maintain stabilization after performing the 360° roll. Consequently, the manual PID control flight data has several fluctuations due to manual attitude stabilization, which causes the spikes. However, the E Guidance motor spin rates resemble bang-bang arcs, are very smooth, and reach neutral attitude at around 1 second.

CHAPTER 7
HIERARCHICAL MIXTURE OF EXPERTS WITH EXTENDED KALMAN FILTER BANKS

7.1 Hierarchical Mixture of Experts with Extended Kalman Filters Structure

This section uses the dynamic thrust models from section 2.3, and this chapter uses the results of Ref. [122]. The Hierarchical Mixture of Experts architecture involves banks of experts, where each expert is an Extended Kalman Filter. The best estimate is determined among the filters in each bank, and then the best estimates among the experts are compared to determine the optimal estimate by weighing all the models together [1]. Figure 7.1 shows the proposed HME structure with two banks each with a gating network and two EKFs, bank 1 (left), bank 2 (right); inputs are $\mathbf{Q}, \mathbf{K}$, and $\boldsymbol{y}$, the measurement vector. The top-level gating network, GN_3, weighs the outputs of bank 1 and bank 2. The left box comprises bank 1 and filled light-gray to indicate that it uses the Burgers thrust model. Contrarily, bank 2 contains the right box, filled with dark-gray to indicate that it uses the Staples thrust model. The circles indicate the bank outputs. Table 7.1 provides a brief description of the general details of the EKFs in the HME structure. Each of the banks

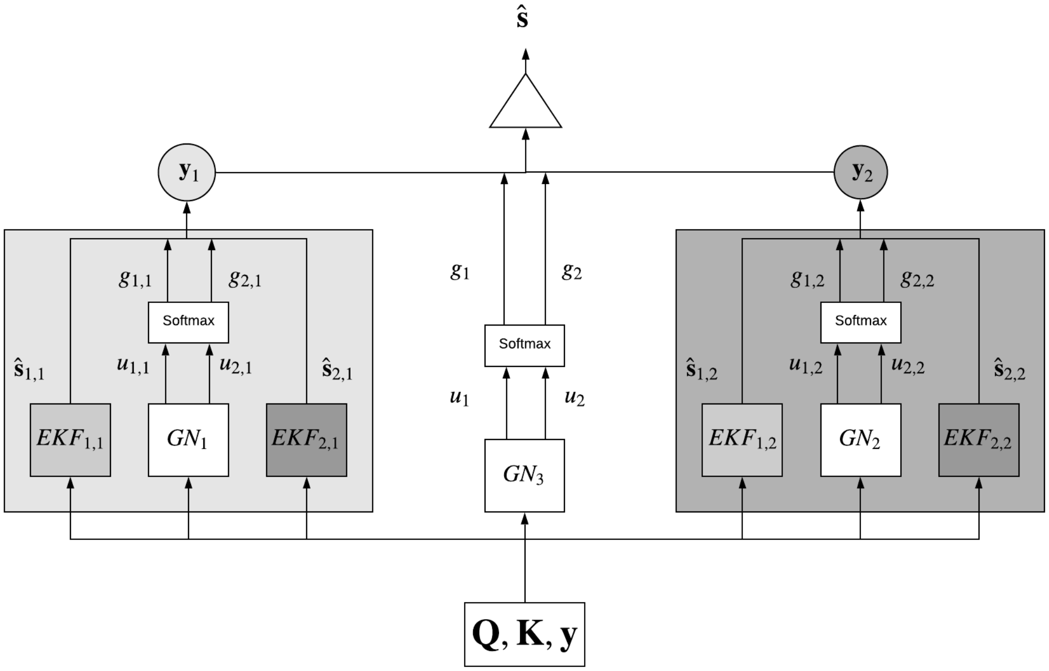


Figure 7.1: Proposed HME Structure

Table 7.1: EKF Description

	Acoustics	Thrust Model	Bank
$EKF_{1,1}$	No	Burgers	1
$EKF_{2,1}$	Yes	Burgers	1
$EKF_{1,2}$	No	Staples	2
$EKF_{2,2}$	Yes	Staples	2

contains two EKFs and a gating network, which uses a modified softmax function to determine the weights. The weights dictate which solution is more accurate and summing the weighted EKF outputs yields the overall output. The bank outputs and overall HME output, weighted best estimate, are defined as:

$$\boldsymbol{y}_1 = g_{1,1}\hat{\boldsymbol{s}}_{1,1} + g_{2,1}\hat{\boldsymbol{s}}_{2,1} \ , \ \boldsymbol{y}_2 = g_{1,2}\hat{\boldsymbol{s}}_{1,2} + g_{2,2}\hat{\boldsymbol{s}}_{2,2} \ , \ \hat{\boldsymbol{s}} = g_1\boldsymbol{y}_1 + g_2\boldsymbol{y}_2. \tag{7.1}$$

The proposed HME structure is general and not constrained to specific types of maneuvers, so it would apply to attitude maneuvers too and other types of aircraft [120].

7.1.1 Gating Networks in the Banks

The first index in the subscript is the expert number. The subscript has a second index, which indicates the bank number for the gating network weights. The top-level gating network, GN_3, is not a bank, so it lacks the second subscript. The gating network weights inside the banks, $g_{j,k}$, use a modified softmax function:

$$g_{j,k} = \frac{exp(-u_{j,k})}{\sum_j exp(-u_{j,k})} \ , \tag{7.2}$$

where j denotes the expert number, and k denotes the bank number. The original formulation of the softmax function lacks a negative sign and uses only positive signs [31, 32, 33, 36]. A softmax function with positive signs favors the larger intermediate gate. However, this study desires the smaller intermediate gate, i.e., smaller error, instead of the larger intermediate gates with larger errors. Thus, including the negative sign allows the softmax function to favor the smaller intermediate gates with smaller errors instead of the larger intermediate gates with larger errors, which is desirable for accurate state estimates. The gating network weights are functions of intermediate gates: $u_{j,k} = Ce_i$, where $C \in \mathbb{R}$ denotes some weighting factor. Alternate forms of the intermediate gates utilize a weight vector multiplied by the measurement vector [31, 32]. The error, e_i, is the root-mean-square error between the dynamical model parameters ($\boldsymbol{s}_{model,i}$) and the state estimate parameters after measurements ($\hat{\boldsymbol{s}}_i(+)$) with i as the iteration in the EKF computation loop:

$$e_i = rms(\boldsymbol{s}_{model,i} - \hat{\boldsymbol{s}}_i(+)).$$

7.1.2 Top-level Gating Network

The top-level gating network, GN_3, sums the outputs from bank 1 and bank 2 as a weighted sum. Its gating network weights are resemble the weights defined in Eq. (7.2), and its intermediate gates have a similar form:

$$u_1 = Ce_1 \ , \ u_2 = Ce_2$$

with the error for the top-level gating network at each iteration defined as:

$$e_1 = rms(\boldsymbol{y}_1 - \boldsymbol{s}_{model}) \ , \ e_2 = rms(\boldsymbol{y}_2 - \boldsymbol{s}_{model}) \ , \tag{7.3}$$

with e_1 as the error of bank 1, and e_2 as the error of bank 2.

7.2 Extended Kalman filters in HME

A quick recap of the HME structure reminds readers that there are four EKFs in the proposed HME structure. The two EKFs in bank 1 use the Burgers thrust model, while the two EKFs in bank 2 use the Staples thrust model. Figure 7.1 shows the EKFs with acoustics are in the dark gray-filled boxes, while the EKFs without acoustics are in the light gray-filled boxes.

Measurement and Process Noise Matrices. The measurement and process noise matrices, **Q** and **K**, respectively, are conventional square matrices with the number of rows and columns equal to the number of state variables. The matrices are diagonal with zeros on the off-diagonal elements with the assumption that cross-correlation errors in the process and measurement noises do not exist. Another assumption is that **Q** and **K** are constant, i.e., invariant with respect to time.

Measurement Models. The global positioning system (GPS) and inertial measurement unit (IMU) are the sensors to provide position and angular velocity measurements, which are the first three and last three state variables in Eq. (2.1). A simple GPS measurement model with subscript, *GPS*, contains bias terms, modeled as white noise: [137]

$$\boldsymbol{p} = \boldsymbol{p}_{GPS} + \begin{bmatrix} \delta_{p_N} \\ \delta_{p_E} \\ \delta_{p_U} \end{bmatrix} \ , \ \begin{bmatrix} \delta_{p_N} \sim N[0, \sigma^2_{GPS,h}] \\ \delta_{p_E} \sim N[0, \sigma^2_{GPS,h}] \\ \delta_{p_U} \sim N[0, \sigma^2_{GPS,v}] \end{bmatrix} . \tag{7.4}$$

The angular velocity measurement model has a similar form to the GPS measurement model and uses subscript, *gyro*, to denote gyroscope measurements:

$$\boldsymbol{\Omega} = \boldsymbol{\Omega}_{gyro} + \begin{bmatrix} \delta_{\Omega_x} \\ \delta_{\Omega_y} \\ \delta_{\Omega_z} \end{bmatrix} \ , \ \begin{bmatrix} \delta_{\Omega_x} \sim N[0, \sigma^2_{\Omega_x}] \\ \delta_{\Omega_y} \sim N[0, \sigma^2_{\Omega_y}] \\ \delta_{\Omega_z} \sim N[0, \sigma^2_{\Omega_z}] \end{bmatrix} . \tag{7.5}$$

The SPL measurements in dB(A) from the MicW i437L Omnidirectional Measurement Microphone give the calibrated sound pressure level, which is similar to an average of the sound level relative to the standard pressure level: [138]

$$L_{pr} = 20 \log_{10} \left(\frac{pr}{pr_{ref}} \right) , \tag{7.6}$$

where pr is the root-mean square sound pressure level, and $pr_{ref} = 20 \ \mu Pa$ is the reference pressure level. Faster motor angular velocities produce more noise, and consequently, higher SPL values occur. Directly relating SPL to motor angular velocities keeps the model as simple as possible, and section 7.5 provides more details. The measurement matrix in extended Kalman filters is defined as: [118]

$$\mathbf{H}_k(\hat{\boldsymbol{s}}_k(-)) = \left. \frac{\partial \boldsymbol{h}(\boldsymbol{s}(t_k))}{\partial \boldsymbol{s}(t_k)} \right|_{\boldsymbol{s}_k(t) = \hat{\boldsymbol{s}}_k(t)(-)} , \tag{7.7}$$

Applying Eq. (7.7) to Eqs. (7.4)-(7.5) yields:

$$\mathbf{H}_k(\hat{\boldsymbol{s}}_k(-)) = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} . \tag{7.8}$$

Utilizing non-Gaussian noise instead of Gaussian noise may increase the accuracy of the measurement models, which is potential future work.

Uncertainty Covariance Matrix. Since **Q** and **K** are diagonal without any cross-correlation errors, the covariance matrix, **P**, is also assumed to lack cross-correlation errors and has a similar form:

$$\mathbf{P} = diag(p_{ii}) \ , \ i = 1, ..., 12. \tag{7.9}$$

Unlike **Q** and **K**, **P** is not invariant with respect to time because the error covariance propagation depends on time and **F**, which is defined as: [118]

$$\mathbf{F}(\hat{\boldsymbol{s}}(t), t) = \left. \frac{\partial \boldsymbol{f}(\boldsymbol{s}(t), t)}{\partial \boldsymbol{s}(t)} \right|_{\boldsymbol{s}(t) = \hat{\boldsymbol{s}}(t)} , \tag{7.10}$$

where $\boldsymbol{f}(\boldsymbol{s}(t), t)$ represents the right-hand side of the dynamical model equations. Putting **F** into a matrix formulation yields: [118]

$$\mathbf{F}(\hat{\boldsymbol{s}}(t), t) = \left[\frac{\partial f_i}{\partial s_j} \right] \ , \ i, j = 1, ..., 12. \tag{7.11}$$

Initial Conditions for the State Vector and Uncertainty Covariance Matrix. The initial state vector is defined as:

$$\boldsymbol{s}_0 = \begin{bmatrix} p_{N,0} & p_{E,0} & p_{U,0} & v_{N,0} & v_{E,0} & v_{U,0} & q_{0,0} & q_{1,0} & q_{2,0} & \omega_{x,0} & \omega_{y,0} & \omega_{z,0} \, , \end{bmatrix}^T . \tag{7.12}$$

and the initial covariance matrix is defined as:

$$\mathbf{P}_0 = diag\left[\sigma^2_{GPS,h} \quad \sigma^2_{GPS,h} \quad \sigma^2_{GPS,v} \quad \sigma^2_{v_N} \quad \sigma^2_{v_E} \quad \sigma^2_{v_U} \quad \sigma^2_{q_0} \quad \sigma^2_{q_1} \quad \sigma^2_{q_2} \quad \sigma^2_{\Omega_x} \quad \sigma^2_{\Omega_y} \quad \sigma^2_{\Omega_z}\right], \tag{7.13}$$

where $diag[\cdot]$ denotes the diagonal elements of the matrix, $\sigma^2_{GPS,h}$ is the GPS measurement variance for in the horizontal plane, $\sigma^2_{GPS,v}$ is the GPS measurement variance for altitude, $\sigma^2_{v,N}, \sigma^2_{v,E}$, and $\sigma^2_{v,U}$ are the variances for velocity, $\sigma^2_{q_0}, \sigma^2_{q_1}$, and $\sigma^2_{q_2}$ are the variances for quaternions, and $\sigma^2_{\Omega_x}, \sigma^2_{\Omega_y}$, and $\sigma^2_{\Omega_z}$ are the variances for angular velocity.

State Vector Propagation. UAV pilots typically prepare trajectories before flights by setting waypoints at various heights with latitude and longitude coordinates. Thus, $\boldsymbol{p}$ and $\boldsymbol{q}$ are known a priori and become the nominal trajectory and attitude through telemetry data from previous flights, assuming the telemetry data provides the true states of the maneuver even though the true trajectory and states will never be known in practice. Taking GPS measurements with noise provides position updates, and differentiating $\boldsymbol{p}$ with respect to time yields velocity. Similarly, taking gyroscope measurements with noise provides angular velocity measurements. Converting angular velocity to the time derivatives of quaternions and integrating them yields the quaternions. Utilizing cubic spline curve fits of the a priori $\boldsymbol{p}$ trajectory gives $\boldsymbol{p}(t)$ and $\boldsymbol{v}(t)$ as functions of time. Incorporating GPS noise deviates $\boldsymbol{p}(t)$ and consequently, $\boldsymbol{v}(t)$ from the nominal trajectory and velocity. A similar case exists for $\boldsymbol{q}(t)$ with gyroscope noise measurements affecting the quaternion estimations. Having $\boldsymbol{p}, \boldsymbol{v}$, and $\boldsymbol{q}(t)$ as explicit functions of time allows one to integrate the error covariance with respect to time.

Integrating Uncertainty Covariance Equations. Integrating the uncertainty covariance involves these time-dependent functions:

$$\dot{\mathbf{P}} = \mathbf{F}(\hat{\boldsymbol{s}}(t), t)\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}^T(\hat{\boldsymbol{s}}(t), t) + \mathbf{Q}(t). \tag{7.14}$$

To simplify the integration of Eq. (7.14), one can assume the error covariance is in steady-state such that $\dot{\mathbf{P}} = 0$, but this assumption is assumed to be invalid due to the quadcopter moving constantly, which contributes to variable uncertainty as time marches on, i.e., covariances are not constant. An alternative integration approach uses linear Kalman filters for a linearized system instead of EKFs for nonlinear systems. Then, utilizing the state transition matrix allows the error covariance to propagate forward [118]. However, $\mathbf{F}$ may be invariant with respect to time for some systems, which also makes it very convenient for integrating the error covariance propagation [99]. Conveniently, $\mathbf{P}$ is diagonal and lacks cross-correlation errors. Using a change of variables simplifies $\dot{\mathbf{P}}(t)$:

$$\begin{aligned}\dot{\mathbf{P}}(t) &= \mathbf{A}(t) + \mathbf{B}(t) + \mathbf{Q}(t)\ ,\ \mathbf{Q}(t) = [\mathbf{q}_{ij}]\ ,\ t_{k-1} \le t \le t_k\ , \\ \mathbf{A} &= \mathbf{F}(\hat{\mathbf{s}}(t), t)\mathbf{P}(t) \quad = [\mathbf{a}_{ij}]\ \ ,\ a_{ij} = \sum_{k=1}^{n} \frac{\partial f_i}{\partial s_k} p_{kj}\ , \\ \mathbf{B} &= \mathbf{P}(t)\mathbf{F}(\hat{\mathbf{s}}(t), t)^T = [\mathbf{b}_{ij}]\ ,\ b_{ij} = \sum_{k=1}^{n} p_{ik} \frac{\partial f_k}{\partial s_j}.\end{aligned} \tag{7.15}$$

Without correlations between state vector components, $\dot{P}(t)$ becomes:

$$\dot{p}_{ii} = \ a_{ii} + b_{ii} + q_{ii}\ ,\ i = 1, ..., n. \tag{7.16}$$

Integrating both sides of $\dot{p}_{ii}$ gives:

$$\int_{t_{k-1}}^{t_k} \dot{p}_{ii}\ dt = \int_{t_{k-1}}^{t_k} \bigg(a_{ii} + b_{ii} + q_{ii}\ \bigg) dt. \tag{7.17}$$

Expanding the left hand side gives:

$$\mathbf{P}_k(-) - \mathbf{P}_{k-1}(+) = \int_{t_{k-1}}^{t_k} \bigg(a_{ii} + b_{ii} + q_{ii}\bigg) dt. \tag{7.18}$$

Finally, moving $\mathbf{P}_{k-1}(+)$ to the right hand side gives:

$$\mathbf{P}_k(-) = \mathbf{P}_{k-1}(+) + \int_{t_{k-1}}^{t_k} \bigg(a_{ii} + b_{ii} + q_{ii}\bigg) dt. \tag{7.19}$$

Computing the partial derivatives to determine the **F** matrix yields some non-zero terms, but most of the terms are conveniently zero. Ultimately, this leaves only the partials for the velocity covariance components on the diagonal of **F**, which are on the diagonal of **F** in the $44, 55$, and 66 entries. Conveniently, the remaining nine diagonal elements are zero, which simplifies the integration of Eq. (7.14). See Appendix to view the equations for the velocity partials for the Burgers model and Staples model. Overall, there are three differential equations and nine linear equations per thrust model to solve to propagate the error covariance:

$$\begin{aligned}
p_k(-)_{11} &= p_{k-1}(+)_{11} + q_{11}\Delta t\,, & p_k(-)_{22} &= p_{k-1}(+)_{22} + q_{22}\Delta t \\
p_k(-)_{33} &= p_{k-1}(+)_{33} + q_{33}\Delta t\,, & \dot{p}_k(-)_{44} &= p_{k-1}(-)_{44} \cdot 2\frac{\partial \dot{v}_N}{\partial v_N} + q_{44} \\
\dot{p}_k(-)_{55} &= p_{k-1}(-)_{55} \cdot 2\frac{\partial \dot{v}_E}{\partial v_E} + q_{55}\,, & \dot{p}_k(-)_{66} &= p_{k-1}(-)_{66} \cdot 2\frac{\partial \dot{v}_U}{\partial v_U} + q_{66} \\
p_k(-)_{77} &= p_{k-1}(+)_{77} + q_{77}\Delta t\,, & p_k(-)_{88} &= p_{k-1}(+)_{88} + q_{88}\Delta t \\
p_k(-)_{99} &= p_{k-1}(+)_{99} + q_{99}\Delta t\,, & p_k(-)_{1010} &= p_{k-1}(+)_{1010} + q_{1010}\Delta t \\
p_k(-)_{1111} &= p_{k-1}(+)_{1111} + q_{1111}\Delta t\,, & p_k(-)_{1212} &= p_{k-1}(+)_{1212} + q_{1212}\Delta t
\end{aligned} \tag{7.20}$$

where Δt denotes the time step between iterations, $(-)$ denotes before measurements, $(+)$ denotes after measurements, and the factor of 2 comes from having a symmetric $\mathbf{F}$ matrix. Utilizing the Runge-Kutta (4,5) formula with the Dormand-Prince pair solves the three differential equations at each iteration with $\mathbf{P}_{k-1}(+)$ as the initial condition.

7.3 Acoustic Data Collection

Obtaining acoustic measurements was initially thought to aid in estimation accuracy, which has not been done before in the UAV acoustic community. The Rbotics Databot sensor was installed on the top of the Raspberry Pi case on the DJI M100 for acoustic sensor data collection. The Databot sensor comes with a mic and various sensors: air pressure, altitude, humidity, accelerometer, gyroscope, magnetometer, ultraviolet, light (lux), carbon dioxide, and volatile organic compounds [139]. The initial plan was to use the Databot's mic to collect acoustic measurements of the DJI M100 as it flew various maneuvers.

However, there were some issues using the Databot sensor to collect acoustic measurements. One issue was a remote controller connection issue when connecting the Databot sensor via Bluetooth to an Android smartphone. A possible solution involves connecting the Databot sensor to the Raspberry Pi. Another issue was that the Arduino program for gathering microphone data has units in the range of 16 or 17 dB in a quiet room, which is much lower than the typical range of a quiet room, i.e., 40 dB. Thus, an alternative acoustic route with different sensors took precedence.

The Tascam DR-10L provides uncalibrated waveform data through the AudioTools app's Recorder feature, while the MicW i437L Omnidirectional Microphone collects calibrated SPL measurements in dB(A). Analyzing the numerous data from flight tests led to attempts in determining relationships between state variables and acoustic measurements. It is possible to find relationships among the motor angular velocities, velocity, acceleration, jerk, and snap, but this is outside the scope of this work.

7.4 Attempts for Deriving Acoustic Models for Quadcopters

This section will highlight the attempts in deriving an acoustic quadcopter thrust model, which were unfortunately not successful even after following a similar approach [92]. Eqs. 51-52 (p.1127) of Ref. [92] apply to hovering or static rotors, i.e., a hovering quadcopter. The idea was that if dB is known, then working backwards allows one to compute p'_{Ln}, which is a function of propeller angular velocity, speed of sound, observer distance from the rotation center, force per unit area on the medium, local Mach number in the radial direction, period of the sound, and harmonic number. Of these variables, the motor frequency is important and the most relevant, so the other terms can be lumped together. Algebraic manipulations allow us to solve for the angular velocity of the propeller blade.

It is common knowledge that 1 Hz = 60 rpm because there are 60 seconds in a minute. If we have the motor rpm, then, we can get the motor frequency. Contrarily, if we have the motor frequency, we can get the motor rpm. Therefore, knowing the dB values leads to estimating or calculating the motor spin rates, which are the control variables.

One of the restrictions of this method (p.1128 of Ref. [92]) is "far field position of the observer." Assuming the Mic i437L would be the observer, asks the question if the Mic i437L is for far field. If so, this method applies to the hover maneuver because the second restriction is "no forward motion of the rotor is allowed." For the other maneuvers with translational motion, other models need to be considered.

An attempted example of determining a UAV acoustic model started with a sample of motor spin rates and dB during the hover maneuver at 15.193 seconds: 86.5316 dB with an average motor spin rate of 499.6441 rad/s.

Using Eq. (52) of Ref. [92] with 86.5316 dB to solve for $|p'_{Ln}|$ gives $p'_{Ln} = 0.29998186$. Computing the integral in Eq. (51) of Ref. [92] in three ways does not provide close enough results to $p'_{Ln} = 0.29998186$, assuming l_r equals weight/area. The three different integration approaches were:

1. Assumed the loading force per unit area (l_r) and the local Mach number in the radial direction were not constant (integrate with respect to r)
2. Assumed the loading force per unit area (l_r) and the local Mach number in the radial direction were constant throughout the radius of the propeller
3. Assumed only the loading force per unit area (l_r) was constant

Overall, current explanations are either of the following two options:

1. not integrating correctly
2. the Hawkings and Lowson's method (p.1127-1128 of Ref. [92]) for subsonic/supersonic rotors for hovering or static conditions does not apply to quadcopter propellers

To avoid complicated derivations and to keep the problem simple, the approach was to directly relate SPL with motor angular velocities. Previous quadcopter dynamical model equations used static thrust, which would probably be inaccurate [100]. Applying relevant dynamical propeller thrust models to quadcopters would theoretically increase the dynamical model accuracy by following the approaches in [104] and [102]. Thus, rederiving the quadcopter dynamical model with dynamic propeller thrust would be necessary but possibly tedious. Then, the last steps would be to create EKFs and then insert into HME.

7.5 Hierarchical Mixture of Experts Methodology

7.5.1 Baseline Indoor Data Collection

The MicW i437L has a frequency range from 20 Hz to 20 kHz and a sensitivity of -27.5 dBFS at 1 kHz, and a signal-to-noise ratio above 62 dB [140]. Connecting the MicW i437L via lightning cable to an iPhone 5S allows users to collect acoustic data through the NoiseLab-Lite and AudioTools IOS apps. The NoiseLab-Lite app provides sound pressure level (SPL) measurements in dB(A), and it can produce calibrated SPL measurements when attached with the MicW i437L omnidirectional measurement microphone, which attaches to an iPhone 5S via lightning connector. The Recorder tool in the AudioTools app provides uncalibrated waveform data. Two tests indoors provides baseline acoustic data of the quadcopter with and without propellers.

1. No propeller test - spinning the motors from 0% to 100% without any propellers
2. Propeller test with takeoff - takeoff indoors at low altitude with propellers spinning

Both tests were manual and in a quiet, indoors environment with an average baseline SPL of approximately 40 dB(A).

7.5.2 Description of Maneuvers

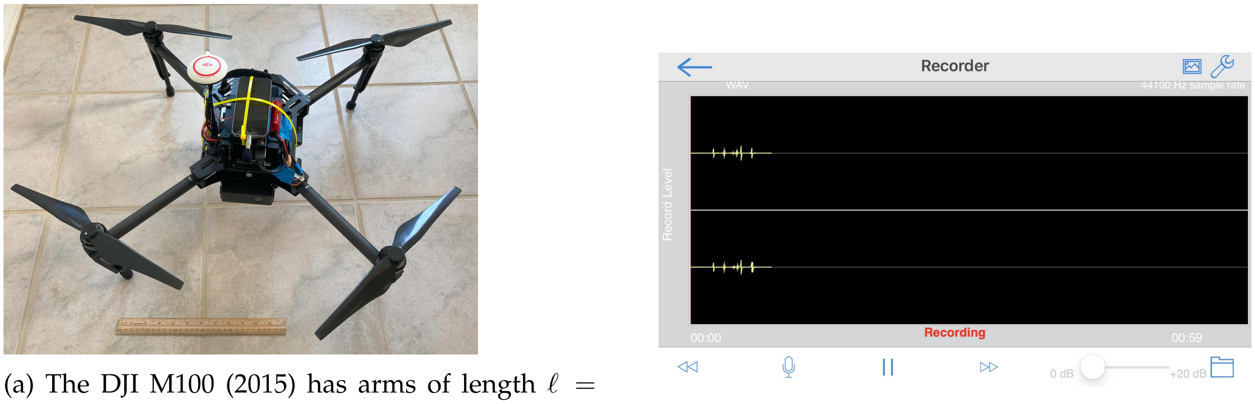
Five maneuvers were conducted for collecting acoustic data:

1. Hover - maintain a position with small drifts due to the wind
2. Waypoint - fly to pre-determined GPS coordinates
3. Triangle - fly a series of waypoints in a triangle pattern, where the flight plan is to stop at each waypoint, turn (yaw) towards the next waypoint, and then proceed to the next waypoint
4. Shoelace loop - fly a series of waypoints in a shoelace loop pattern, where the flight plan involves mostly smooth transition from waypoint to waypoint without the stop and turn approach in the triangle maneuver
5. Max throttle - apply max thrust in the NEU directions for a short amount of time

Each maneuver started with a manual takeoff and then shifted into autonomous mode to fly through the waypoints. The hover and max throttle maneuvers remained completely in manual mode.

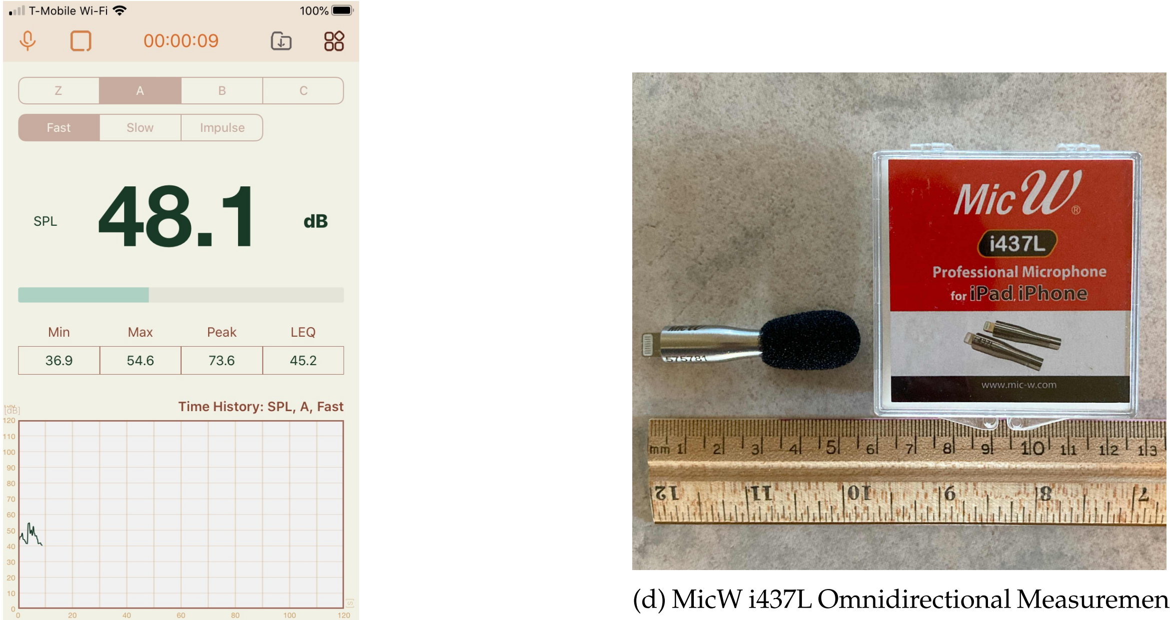
7.5.3 DJI M100 Description with Acoustic Setup

The authors performed the waypoint, triangle, and shoelace loop maneuvers on a DJI M100 quadcopter, the same platform used for the takeoff and waypoint guidance maneuvers in Ref. [124, 123, 121].



(a) The DJI M100 (2015) has arms of length $\ell = 0.31\ m$, 13" propellers with pitch of 4.5", and a total mass of $m = 3.133\ kg$ with one TB48D 6S (5700 mAhr) battery

(b) AudioTools recorder screenshot with a sample rate of 44.1 kHz to generate wav files, which are opened through MATLAB's *audioread* function



SPL 48.1 dB

Min	Max	Peak	LEQ
36.9	54.6	73.6	45.2

(c) NoiseLab-Lite recording screenshot with a sample rate of 44100, buffer size of 16384, calibrated at 0.0 dB, measuring SPL in dB(A) with the fast option

(d) MicW i437L Omnidirectional Measurement Microphone (Lightning Connector) with a frequency band from 20 Hz to 20 kHz, sensitivity of -27.5 dBFS (94dB SPL at 1 kHz), a signal-to-noise ratio over 62 dB, connects directly to iPhone or iPad via light, and weighs 10 g (User's manual)

Figure 7.2: DJI M100 and Acoustic Equipment

7.5.4 SPL and Motor Angular Velocity Curve Fit

Combining the telemetry data, numerous SPL measurements from the five maneuvers, and the propeller test data provides a sufficient sample size of 789 data points, which are used to generate a curve fit model between SPL and motor angular velocity. The Least Absolute Residuals (LAR) method provides good polynomial fits of SPL vs. motor angular velocities. It takes on the form with $\alpha_j \in \mathbb{R}$, $j = 1, ..., 5$:

$$SPL_k = \alpha_1 \omega_{i,k}^4 + \alpha_2 \omega_{i,k}^3 + \alpha_3 \omega_{i,k}^2 + \alpha_4 \omega_{i,k} + \alpha_5 \ , i = 1, ..., 4. \tag{7.21}$$

Table 7.2 shows the coefficients of the curve fit. Figure 7.3 shows the curve fits of SPL to ω_i.

Table 7.2: SPL and Motor Angular Velocity Curve Fit Parameters

	R^2	RMSE	α_1	α_2	α_3	α_4	α_5	μ	σ
ω_1	0.9951	0.7601	0	0	-4.781	0.8472	87.58	429.7	148
ω_2	0.9939	0.8508	0	0	-3.979	2.89	86.66	437	157.1
ω_3	0.9931	0.9033	0	0.188	-3.115	3.535	85.93	439.1	159.3
ω_4	0.9947	0.7926	-1.207	-2.485	0.9231	9.556	83.01	447.1	161.3

7.6 Flight Test Results

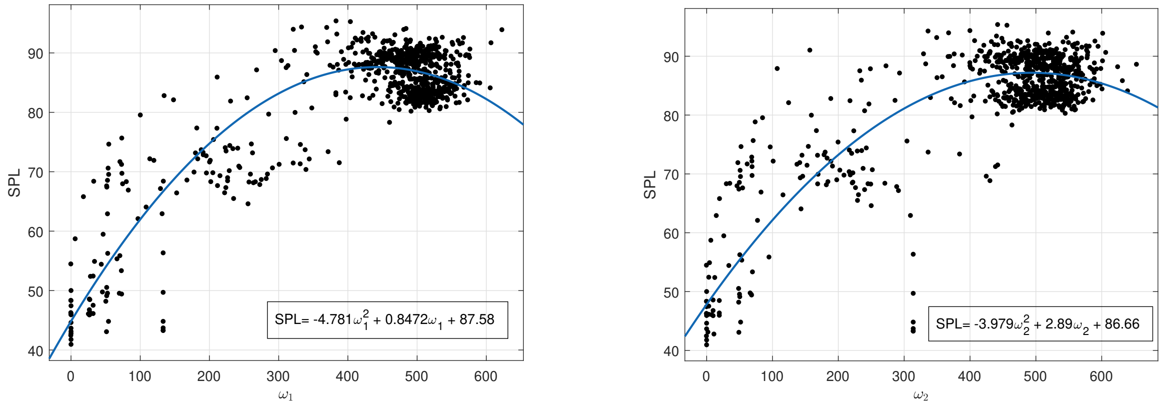
This section follows the work of Ref. [122]. The author conducted flight tests of the five maneuvers using the DJI M100, shown in Figure 7.2a. The waypoint, triangle, and shoelace loop maneuvers occurred at Mililani Ravine Park, while the propeller test and hover maneuver were performed at the author's home. The DJI Ground Station Pro app set the pre-determined waypoints and paths for the maneuvers. Each maneuver had different types of data: the DJI M100 telemetry data extracted post-flight, the uncalibrated AudioTools waveform data, and the recorded SPL data from NoiseLab-Lite with the calibrated MicW i437L Omnidirectional Measurement Mic.

Maneuver Trajectories. The NE coordinates in the NEU frame used the latitude and longitude GPS data in radians:

$$N = R_E(lat - lat_0) \ , \tag{7.22}$$

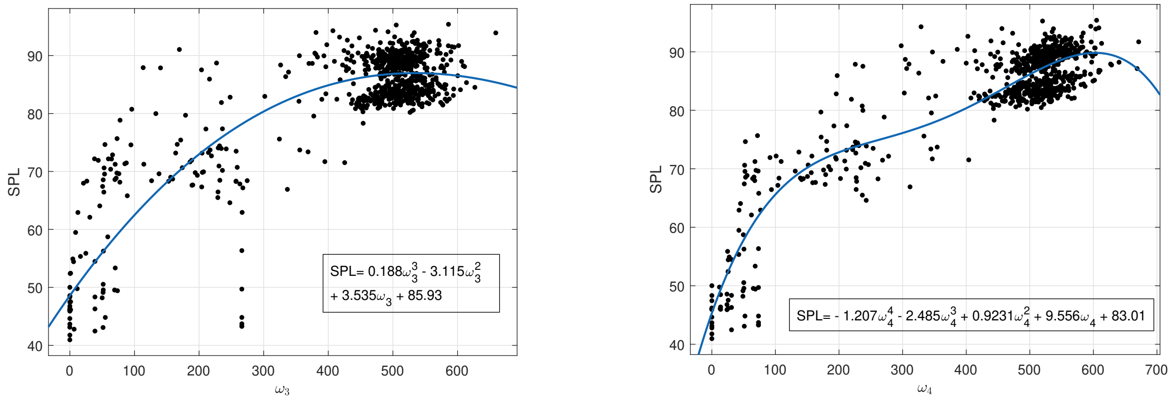
$$E = R_E \cos(lat)(lon - lon_0) \ , \tag{7.23}$$

with R_E as the radius of the Earth in meters, lat as the current latitude in radians, lon as the current longitude in radians, lat_0 as the initial latitude in radians, and lon_0 as the initial longitude in radians. The U coordinate in the NEU frame came from the relative altitude measurements in the telemetry data.



(a) Curve Fit of SPL to ω_1

(b) Curve Fit of SPL to ω_2



(c) Curve Fit of SPL to ω_3

(d) Curve Fit of SPL to ω_4

Figure 7.3: Curve Fit of SPL to ω_i

Trajectories in NEU Frame. Figure 7.4 shows the 3D trajectories of the hover, waypoint, triangle, shoelace loop, propeller test, and max throttle maneuvers in the NEU frame. The hover maneuver involves hovering without any pilot inputs for approximately 6 seconds with several small fluctuations due to the wind, but it did not deviate drastically from its takeoff location, i.e., landed less than 0.5 away from the original takeoff location. After takeoff, the waypoint maneuver had the quadcopter ascend to the desired altitude of approximately 15 m, move horizontally towards the waypoint, and then land after reaching the waypoint.

After taking off, the triangle maneuver also ascended to the desired altitude of approximately 15 m, went to each of the prescribed waypoints to create a triangle path in the air, and then returned to land at its takeoff location. The shoelace loop maneuver involved following waypoints to trace a shoelace loop path in the air. The triangle maneuver had sharp corners because the pilot used the DJI GS Pro *straight* setting for the cornering parameter. Using the *straight* setting caused the drone to stop and then turn at each waypoint towards the next waypoint before resuming. On the other hand, the shoelace loop maneuver had smoother edges due to the *curved* setting for cornering, which provides smoother and continuous flight. The no propeller test's NEU coordinates are not displayed because the drone did not move throughout the test.

The propeller test was indoors, but the GPS signal was just strong enough to get a GPS lock. Similar to the hover maneuver, not much horizontal translation occurred to avoid hitting anything indoors. After takeoff, the max throttle maneuver applied max thrust to ascend vertically, paused for approximately 0.5 seconds, applied max pitch forward, had another short pause also of about 0.5 seconds, drifted slightly due to the wind, applied max pitch backward, applied max thrust to descend and then quickly ascend, and then landed. The small loop-like component in the path occurred when the pilot rapidly switched from max descent to max ascent. The total 3D distance traveled for the maneuvers is as follows: 3.57 m for hover, 56.12 m for waypoint, 122.80 m for triangle, 115.63 m for shoelace loop, 1.61 m for propeller test with takeoff and 29.39 m for max throttle.

2D Trajectories. Since the hover maneuver did not deviate much from the start point, its 2D trajectory is not shown. However, the waypoint, triangle, and shoelace loop maneuvers significantly moved more than the hover maneuver. Their 2D trajectories are shown in Figure 7.5 through Google Earth Pro.

7.7 Acoustic Data

Sound Pressure Level Data. Figure 7.6 shows the MicW i437L Omnidirectional Measurement Microphone calibrated SPL in dB(A) using the NoiseLab-Lite app for all the maneuvers. The hover maneuver had SPL values of approximately 90 dB, fluctuating at approximately 90 dB throughout the maneuver. The waypoint maneuver and shoelace loop maneuvers were generally quieter with SPL values ranging from approximately 80 to 85 dB. The baseline SPL values in at Mililani Ravine

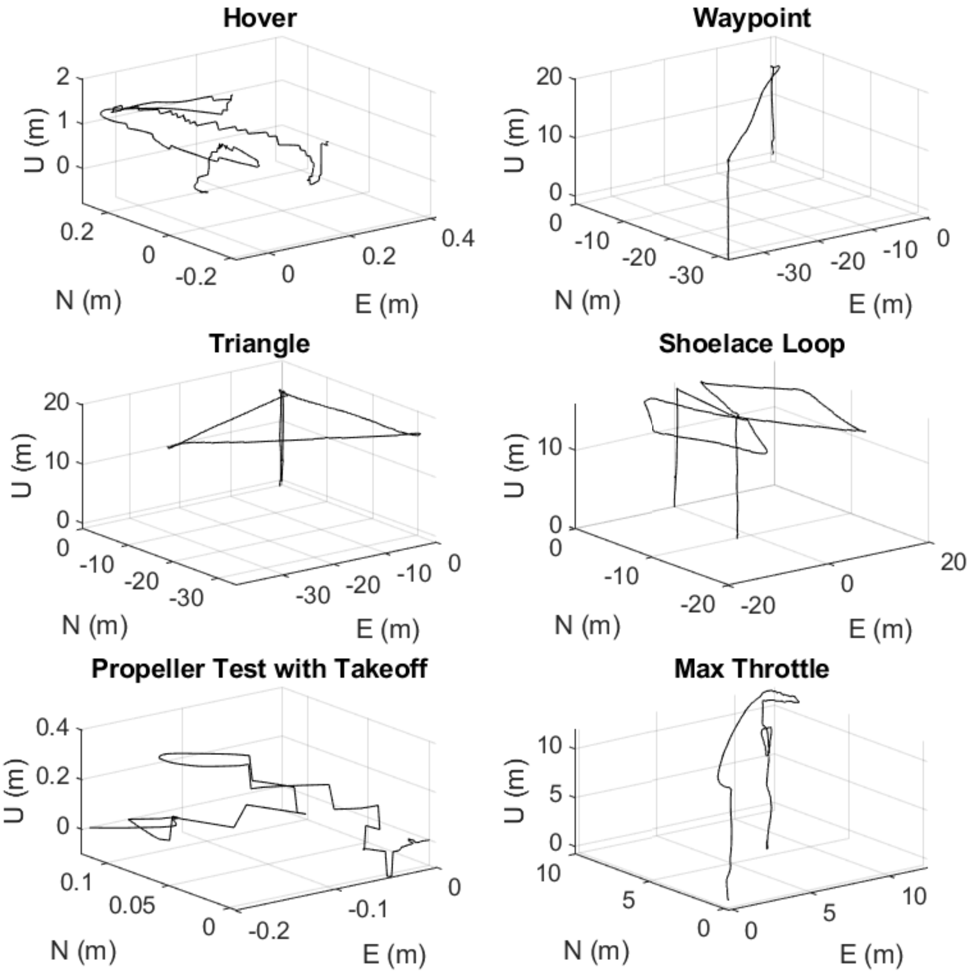
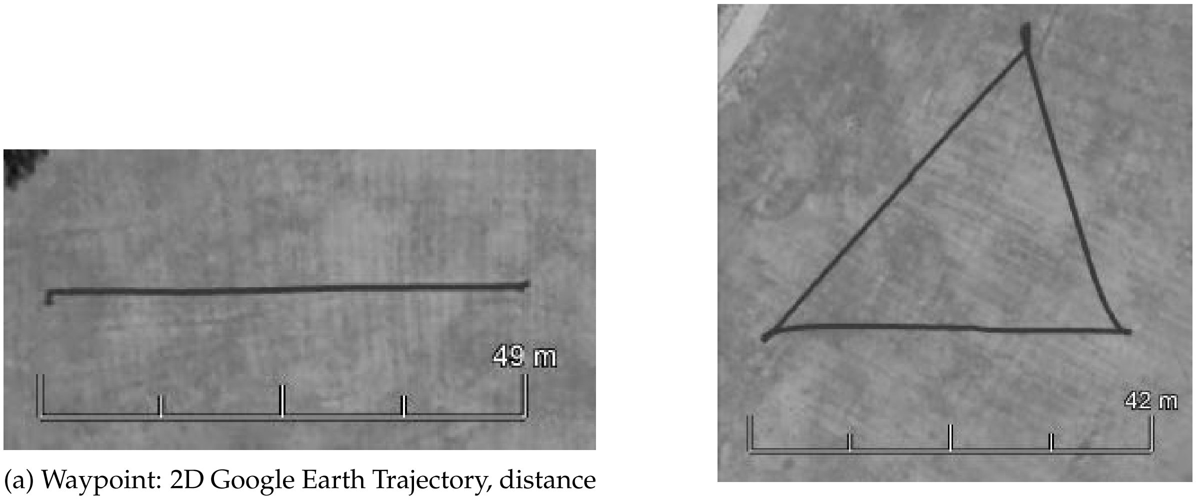
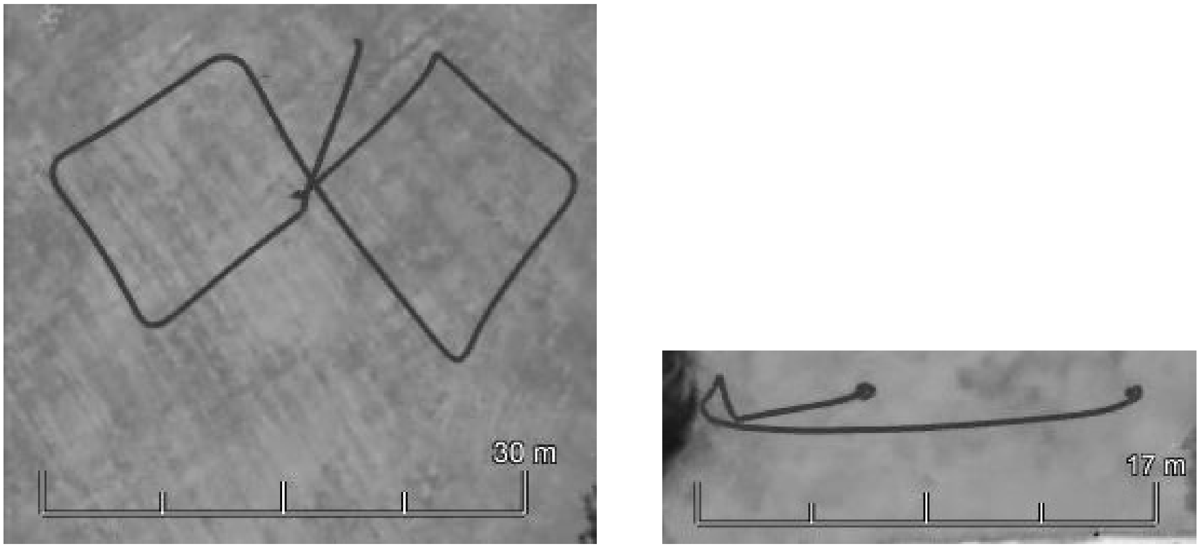


Figure 7.4: North, East, and Up (NEU) Coordinates of the Maneuvers



(a) Waypoint: 2D Google Earth Trajectory, distance from takeoff is 48.6 m

(b) Triangle: 2D Google Earth Trajectory



(c) Shoelace loop: 2D Google Earth Trajectory

(d) Max Throttle: 2D Google Earth Trajectory

Figure 7.5: Mililani Ravine Park, 95-1100 Kaapeha St, Mililani, HI 96789

Park were approximately 45 dB, which fluctuated slightly depending on the wind. The right side of Figure 7.6 shows SPL in dB(A) for the no propeller test, propeller test with takeoff, and max throttle maneuver. The motors generated SPL values up to approximately 70 dB without having propellers. The max throttle test had the largest SPL values as high as 95 dB, which occurred when the pilot sent commands for max throttle during ascent, descent, or max pitch.

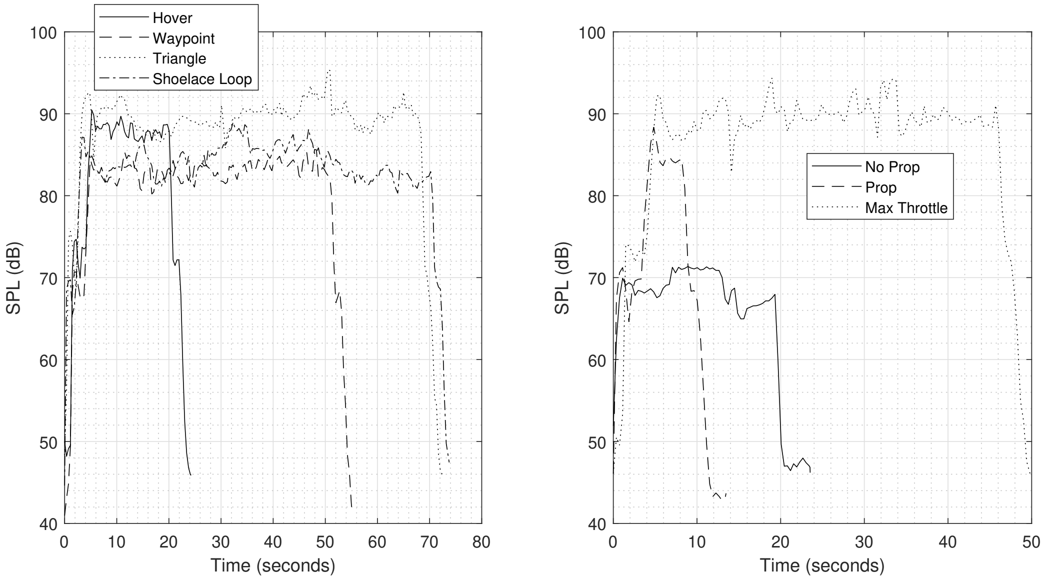


Figure 7.6: NoiseLab-Lite dB(A) for all Maneuvers and Tests

SPL and Motor Spin Rate Data. Figure 7.7 shows the SPL in dB(A) vs. the motor spin rates in rad/s for the hover, waypoint, triangle, and shoelace loop maneuvers. The SPL levels generally reached approximately 85 to 90 dB above 400 rad/s. Starting up the motors during takeoff or slowing them down during landing produced an approximate SPL range between 50-80 dB.

7.8 HME-EKF Simulation Setup and Simulation Results

This section follows Ref. [122] and describes the setup for the HME-EKF simulation. The DJI M100 manual provides GPS measurement errors [127], and the **K** matrix components came from observing the GPS and IMU data in the hover maneuver's telemetry data since the hover maneuver lacked significant remote controller commands. The Q matrix components are chosen to tune the EKFs, and using these values gives **Q** and **K** as:

Table 7.3: **Q** Matrix Components

	σ_{p_N}	σ_{p_E}	σ_{p_U}	σ_{v_N}	σ_{v_E}	σ_{v_U}	σ_{q_0}	σ_{q_1}	σ_{q_2}	σ_{Ω_x}	σ_{Ω_y}	σ_{Ω_z}
Q	2.5	2.5	0.5	0.001	0.001	0.001	0.005	0.005	0.005	0.05	0.05	0.05

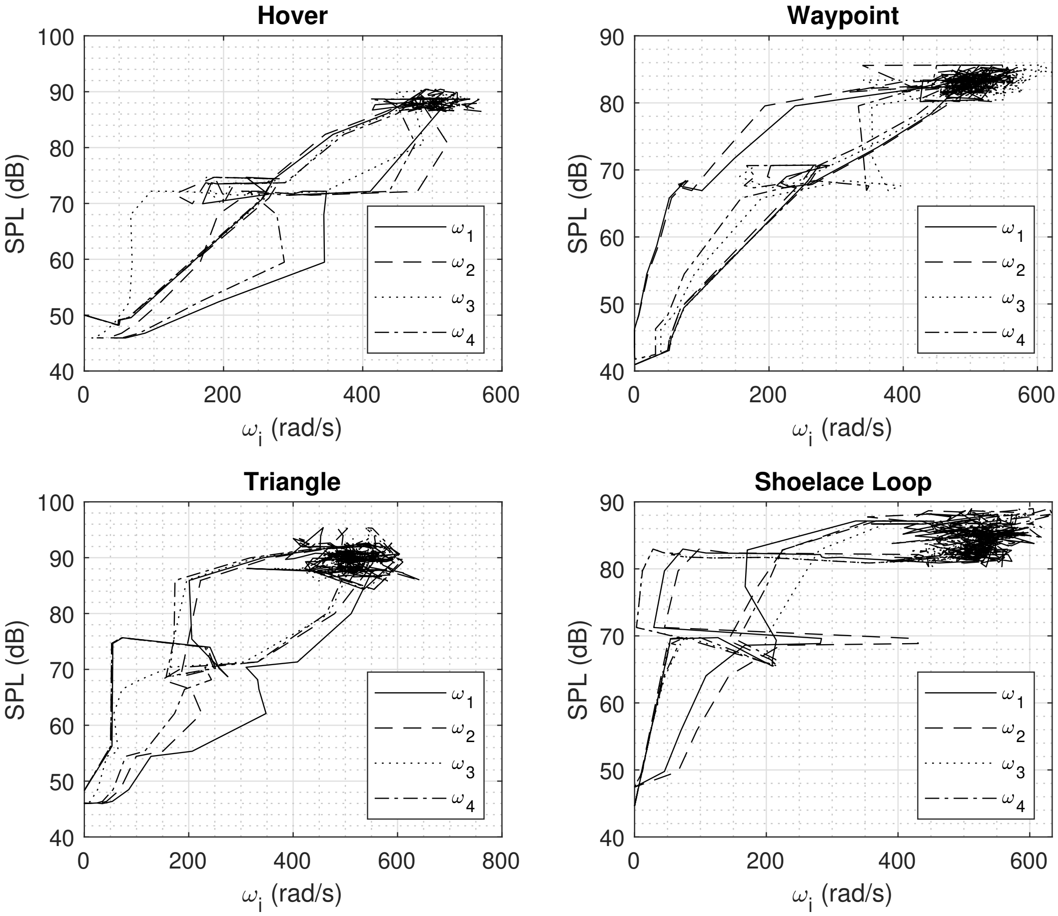


Figure 7.7: SPL vs. Motor Spin Rates for Maneuvers

Table 7.4: **K** Matrix Components

	$\sigma_{GPS,h}$	$\sigma_{GPS,h}$	$\sigma_{GPS,v}$	σ_{a_x}	σ_{a_y}	σ_{a_z}	σ_{gyro_x}	σ_{gyro_y}	σ_{gyro_z}
K	2.5	2.5	0.5	0.02	0.02	0.02	0.0707	0.0707	0.0707

$$\begin{aligned} \mathbf{Q} &= diag\begin{bmatrix}\sigma^2_{p_N} & \sigma^2_{p_E} & \sigma^2_{p_U} & \sigma^2_{v_N} & \sigma^2_{v_E} & \sigma^2_{v_U} & \sigma^2_{q_0} & \sigma^2_{q_1} & \sigma^2_{q_2} & \sigma^2_{\Omega_x} & \sigma^2_{\Omega_y} & \sigma^2_{\Omega_z}\end{bmatrix}, \\ \mathbf{K} &= diag\begin{bmatrix} \sigma^2_{GPS,h} & \sigma^2_{GPS,h} & \sigma^2_{GPS,v} & \sigma^2_{a_x}\Delta t & \sigma^2_{a_y}\Delta t & \sigma^2_{a_z}\Delta t\ ... \\ \sigma^2_{gyro_x}\Delta t & \sigma^2_{gyro_y}\Delta t & \sigma^2_{gyro_z}\Delta t & \sigma^2_{gyro_x} & \sigma^2_{gyro_y} & \sigma^2_{gyro_z}. \end{bmatrix} \end{aligned} \tag{7.24}$$

There were not any sensors to measure velocity or quaternions, so the **K** matrix components for velocity and quaternions use the accelerometer and gyroscope error multiplied by Δt. The Staples thrust model is more accurate than the Burgers thrust model after conducting preliminary calculations of the motor angular velocities. Thus, the Burgers thrust model's process noise includes a factor of two: $\mathbf{Q}_B = 2\mathbf{Q}$, while the EKFs with the Staples thrust model use just **Q**.

This section follows the simulation results of Ref. [122], which applies HME-EKF to the waypoint and triangle maneuvers. Overall, the runtime per cycle is approximately 1.4 seconds, which demonstrates that implementing HME would yield near real-time capabilities. Most of the computation time involves solving the SPL and motor angular velocity curve fit equations and integrating the error covariance propagation equations. The HME approach is not constrained to solely quadcopters, so fixed-wing aircraft can use HME too but with different dynamical model equations. [17, 141]

Accuracy. The EKF state estimation accuracy uses mean square error (MSE) and root mean square (RMS). Some values repeat because they have the same Q components and measurements. Both thrust models are inaccurate after comparing the max motor spin rate values with the telemetry and acoustic motor spin rates. Modeling inaccuracies or lack of measurements lead to increasing covariances, which the next subsection discusses.

For the triangle maneuver, the motor spin rates from Burgers model produce values as high as $2.2328 \cdot 10^{11}\ (rad/s)^2$. The Staples model yields motor spin rates as high as $1.9318 \cdot 10^{11}\ (rad/s)^2$. The estimated motor spin rates from acoustics give values as high as only $2.1692 \cdot 10^{4}\ (rad/s)^2$, while the telemetry data yields values as high as only $3.3209 \cdot 10^{5}\ (rad/s)^2$. The max acoustic and telemetry motor spin rates are considerably smaller than the values from the Burgers and Staples models. Thus, the motor spin rates from the EKFs with acoustics produce more accurate estimations.

Similarly, the waypoint motor spin rates from the Burgers thrust model has values as high as $8.3154 \cdot 10^{10}\ (rad/s)^2$. The Staples thrust model gives values up to $7.1941 \cdot 10^{10}\ (rad/s)^2$. The telemetry data yields motor spin rate values as high as only $3.2286 \cdot 10^{5}\ (rad/s)^2$, and the motor spin rates from acoustics give values as high as $2.1691 \cdot 10^{4}\ (rad/s)^2$. Like the triangle maneuver, the Burgers and Staples models are inaccurate, and the the motor spin rates from the EKFs with acoustics produce more accurate estimations.

Overall, both dynamic thrust models are highly inaccurate compared to the telemetry data. The assumptions simplify the quadcopter dynamic thrust models excessively, which means they do not model the real-world quadcopter dynamic thrust very accurately. Future work includes utilizing other quadcopter thrust models with relative velocity over the motor, the motor's angular velocity, and thrust coefficients [142]. Another option is to tune the dynamic thrust model parameters by tailoring the parameters to the DJI M100 through more flight tests.

Table 7.5: Triangle & Waypoint MSE and RMS Summary of Position and Velocity

	N	E	U	v_N	v_E	v_U
Triangle MSE	4.074	3.010	0.129	0.0087	0.0082	0.0324
Waypoint MSE	2.870	4.267	0.151	0.0062	0.0049	0.0446
Triangle RMS	2.018	1.735	0.360	0.0931	0.0907	0.180
Waypoint RMS	1.694	2.066	0.389	0.0788	0.0703	0.211

Table 7.6: Triangle & Waypoint MSE and RMS Summary of Quaternions and Angular Velocity

	q_0	q_1	q_1	ω_x	ω_y	ω_z
Triangle MSE	0.231	0.0020	0.093	0.0014	0.0013	0.0014
Waypoint MSE	0.0890	0.0274	0.0967	0.0011	0.0014	0.0015
Triangle RMS	0.480	0.0448	0.0966	0.0368	0.0346	0.0372
Waypoint RMS	0.298	0.166	0.311	0.0339	0.0379	0.0381

Gating Networks' Weights. Figure 7.8 shows the profiles of the gating network weights in bank 1 and bank 2 for the triangle maneuver, and the weights sum to one, which provides a symmetric or reflective result. Since $EKF_{2,1}$ always has smaller error than $EKF_{1,1}$, $EKF_{2,1}$ is clearly the best performer because $EKF_{2,1}$ includes acoustics. Consequently, it yields more accurate state and motor angular velocity estimates than $EKF_{1,1}$, which does not include acoustics. Similar results occur in bank 2, where the gating network prefers $EKF_{2,2}$, which includes acoustics. As expected, the EKFs with acoustics provide more accurate estimations, so the gating networks tend to favor those EKFs. Figure 7.9 shows the gating networks of the top-level gating network in the triangle maneuver, and the weights sum to one, which provides a symmetric or reflective result. The top-level gating network prefers bank 2 over bank 1, but there is not a clear preference in choosing between EKFs in banks 1 and 2. The gating network leans towards bank 2, which uses the Staples thrust model, and produces motor angular velocity estimates closer to the telemetry motor angular velocities. Figure 7.10 shows the profiles of the gating network weights of bank 1 and bank 2 for the waypoint maneuver, and the weights sum to one, which provides a symmetric or reflective result. The gating network in bank 1 prefers $EKF_{2,1}$ due to its smaller error than $EKF_{1,1}$. The gating network never changes its selection, so $EKF_{2,1}$ is clearly the better performer, which is expected since $EKF_{2,1}$ uses acoustics. As a result, it yields more accurate motor angular velocity estimates than $EKF_{1,1}$, which does not include acoustics. Once again, the EKFs with acoustics provide more accurate estimations. Figure 7.11 shows the profiles of the gating network weights of the top-level gating network in the waypoint maneuver, and the weights sum to one, which provides a sym-

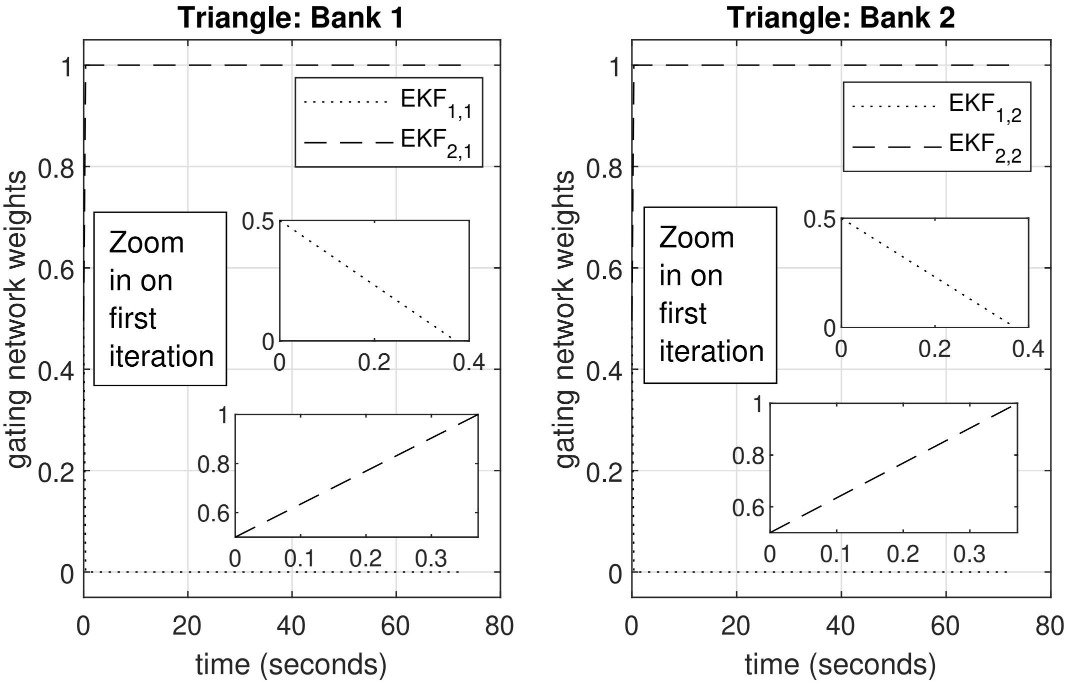


Figure 7.8: Triangle: Bank Comparison of Gating Network Weights

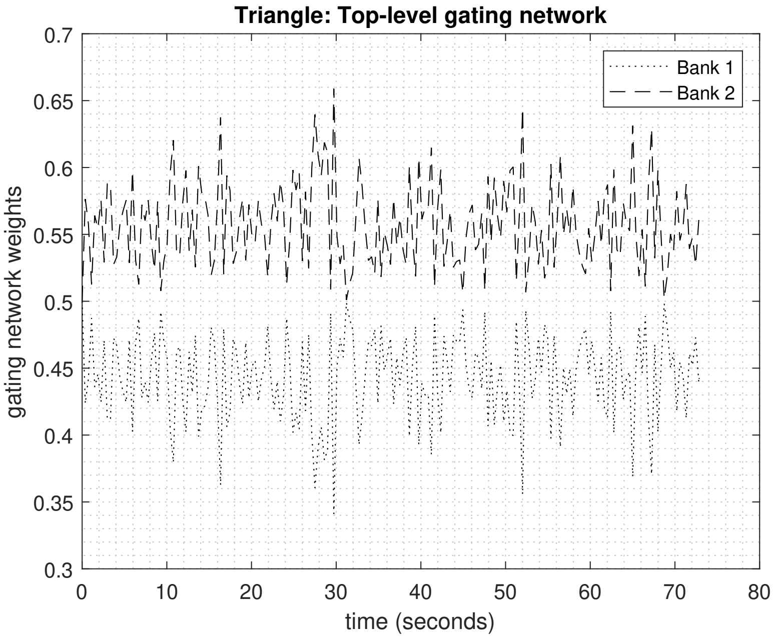


Figure 7.9: Triangle Top-Level Gating Network Weights

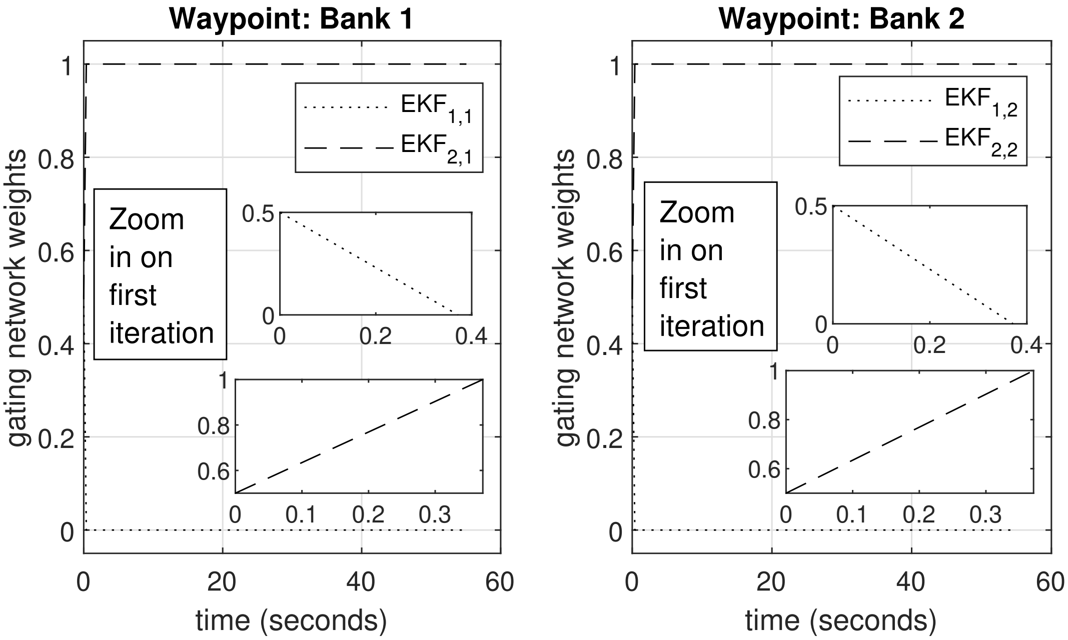


Figure 7.10: Waypoint Bank Comparison of Gating Network Weights

metric or reflective result. The results are similar to the triangle maneuver, where the top-level gating network prefers bank 2 over bank 1. Bank 2 uses the Staples thrust model, which yields motor angular velocity estimates closer to the telemetry motor angular velocities than the Burgers thrust model. Overall, the EKFs with acoustics yield more accurate estimates for the motor angular velocities, and the Staples thrust model is slightly more accurate than the Burgers thrust model. It is unfortunate that both thrust models are inaccurate when compared against the acoustic and telemetry motor angular velocities. If one of the non-acoustic thrust models provided state and motor angular velocity estimations close to the model, then there would be a best performer with one of the banks converging to 1 and the other converging to 0.

Uncertainty Covariances. This subsection shows the uncertainty covariances of the waypoint maneuver's second extended Kalman filter, $EKF_{2,1}$, which is in bank 1 and uses the Burgers thrust model. Figure 7.12 shows the profiles of the position variance vs. time. They converge quickly to fixed values because $EKF_{2,1}$ uses GPS measurements, which allows the uncertainty to be bounded. The blue line represents $\pm 2\sigma$, while the red line represents $\pm 3\sigma$. Adding the mean error to the sigma bounds centers them around the position variances. The position covariances lie within the $\pm 2\sigma$ and $\pm 3\sigma$ bounds, which shows high confidence in the position estimations. Figure 7.13 shows the profile of the velocity variance vs. time, and a quick glance may seem like the uncertainty covariances converge. However, Figure 7.14 shows a zoomed in version of the velocity variances, and the variances for v_N and v_E diverge. Thus, true divergence occurs for the variances for v_N and v_E [118]. The lack of velocity measurements, inaccurate dynamics thrust models, and

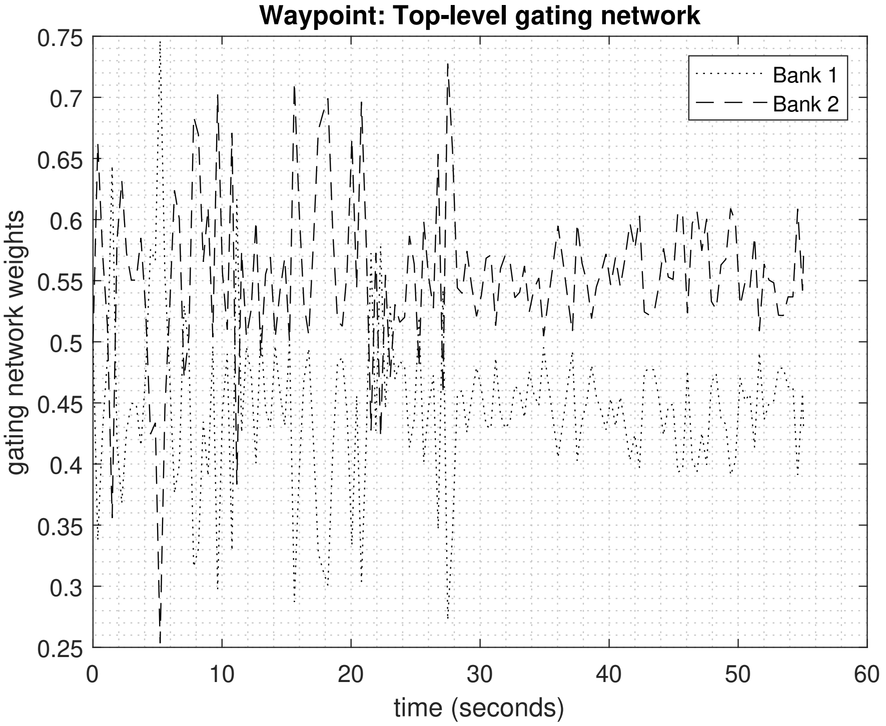


Figure 7.11: Waypoint Top-Level Gating Network Weights

neglecting the cross-correlation covariances are the three main reasons that cause the uncertainties and inaccuracies to grow in time.

Figure 7.15 shows the variance of quaternion vs. time, which diverge due to the lack of quaternion measurements. The q_1 and q_2 variances are the same due to having the same process error variances, so the error covariance propagation equations yield the same values. They stay within the $\pm 2\sigma$ and $\pm 3\sigma$ bounds but gradually diverge. Overall, the quadcopter attitude models are less accurate than the translation models because they approach the 2σ bound closer than the velocity variances. Like the velocity covariances, the quaternion covarainces also demonstrate true divergence due to lack of quaternion measurements, inaccurate thrust models, and neglecting cross-correlation covariances [118]. Figure 7.16 shows the variance of angular velocity vs. time. The angular velocity variances converge quickly due to gyroscope measurements, which is similar to the position variance convergence. Both the position and angular velocity uncertainties converge and remain within the $\pm 2\sigma$ and $\pm 3\sigma$ bounds, which demonstrates high confidence in the estimations. The other EKFs in the triangle and waypoint maneuvers demonstrate very similar results, so they are not shown to conserve space.

Root Mean Square Error vs. Time. Figure 7.17 shows the RMS profiles of $EKF_{2,1}$ of the triangle maneuver with respect to time. The RMS values converge for position and angular velocity, while the RMS values for q_0 and v_U tend to diverge.

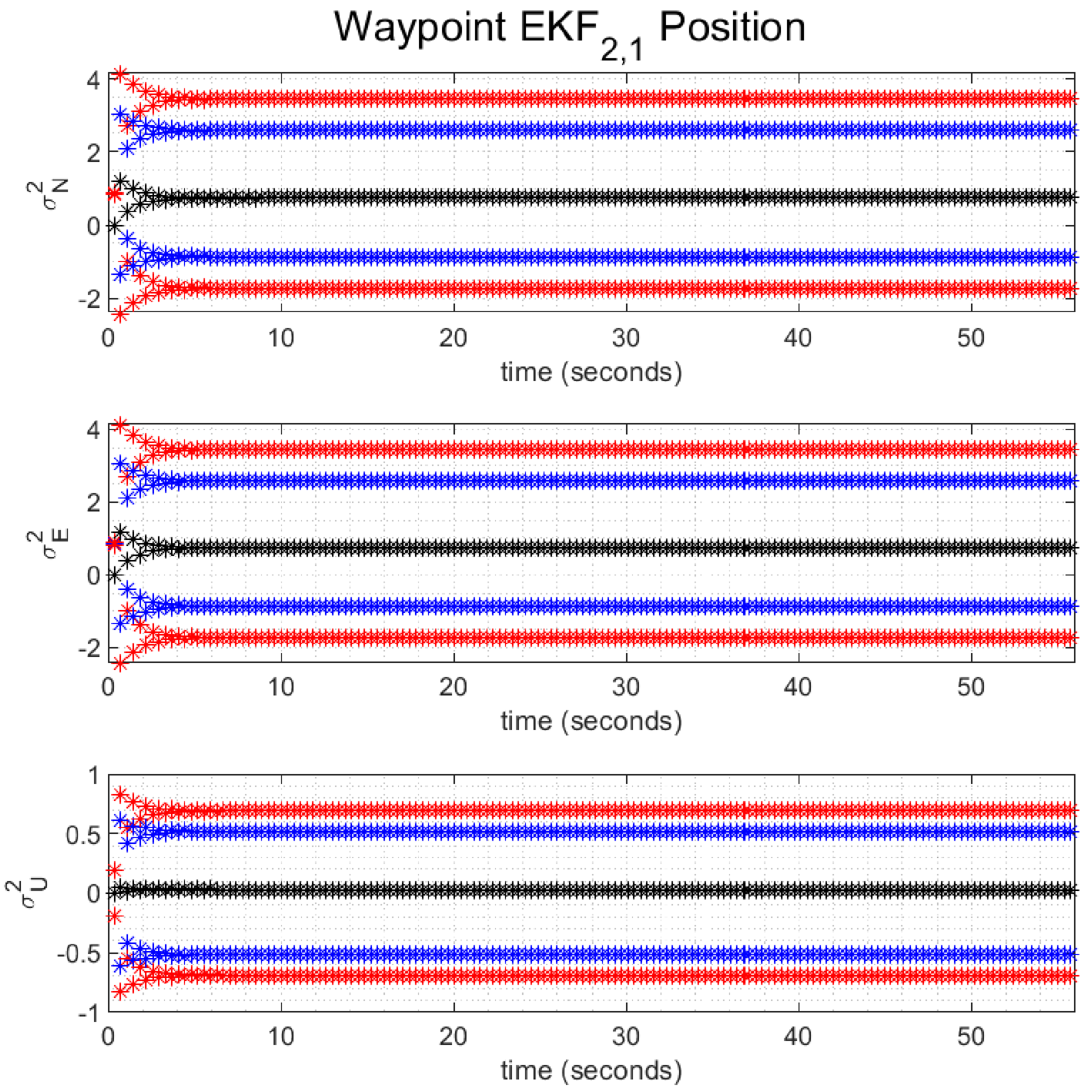


Figure 7.12: Waypoint $EKF_{2,1}$ Position Variance

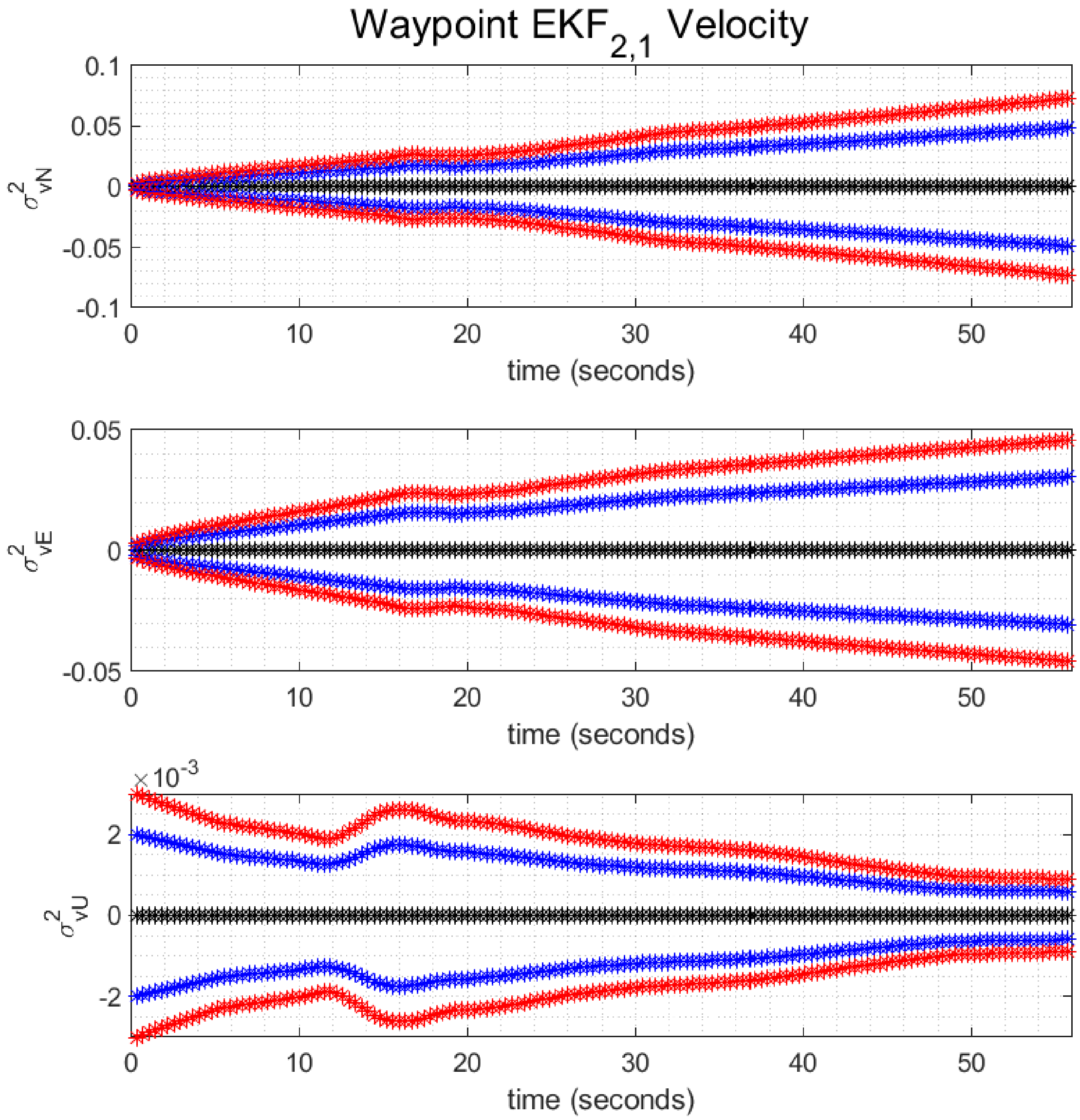


Figure 7.13: Waypoint $EKF_{2,1}$ Velocity Variance

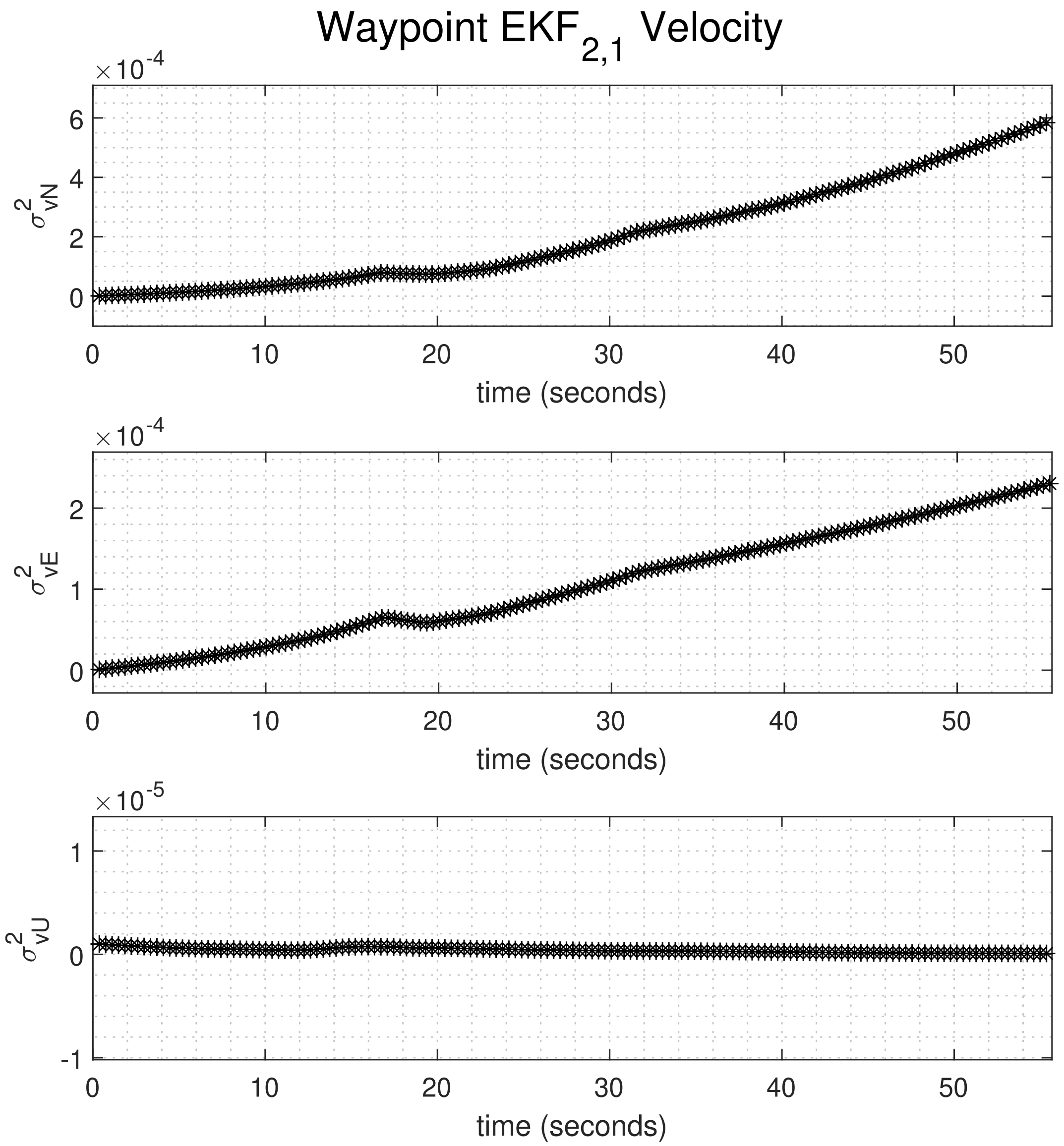


Figure 7.14: Waypoint $EKF_{2,1}$ Velocity Variance Zoomed In

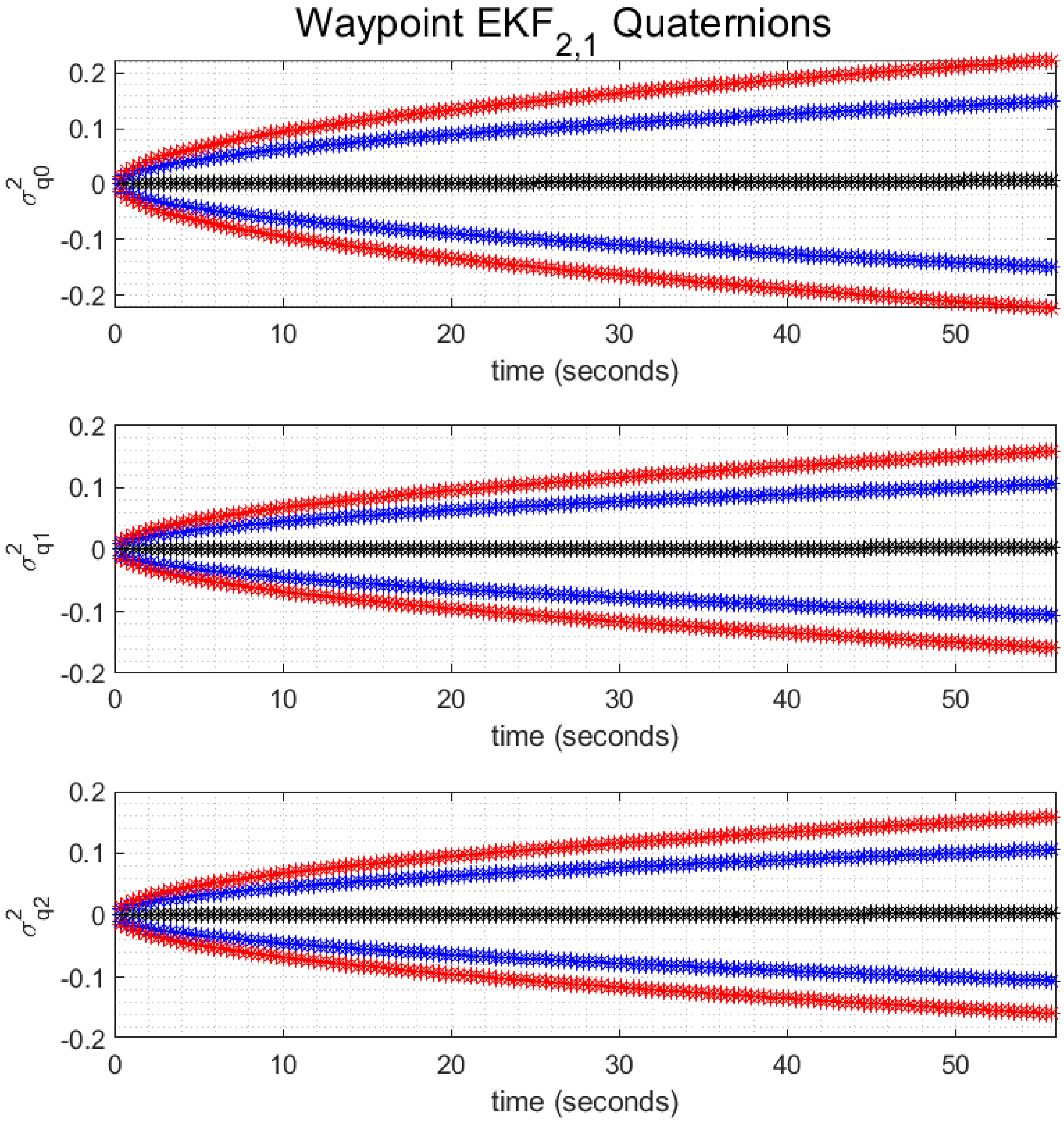


Figure 7.15: Waypoint $EKF_{2,1}$ Quaternion Variance

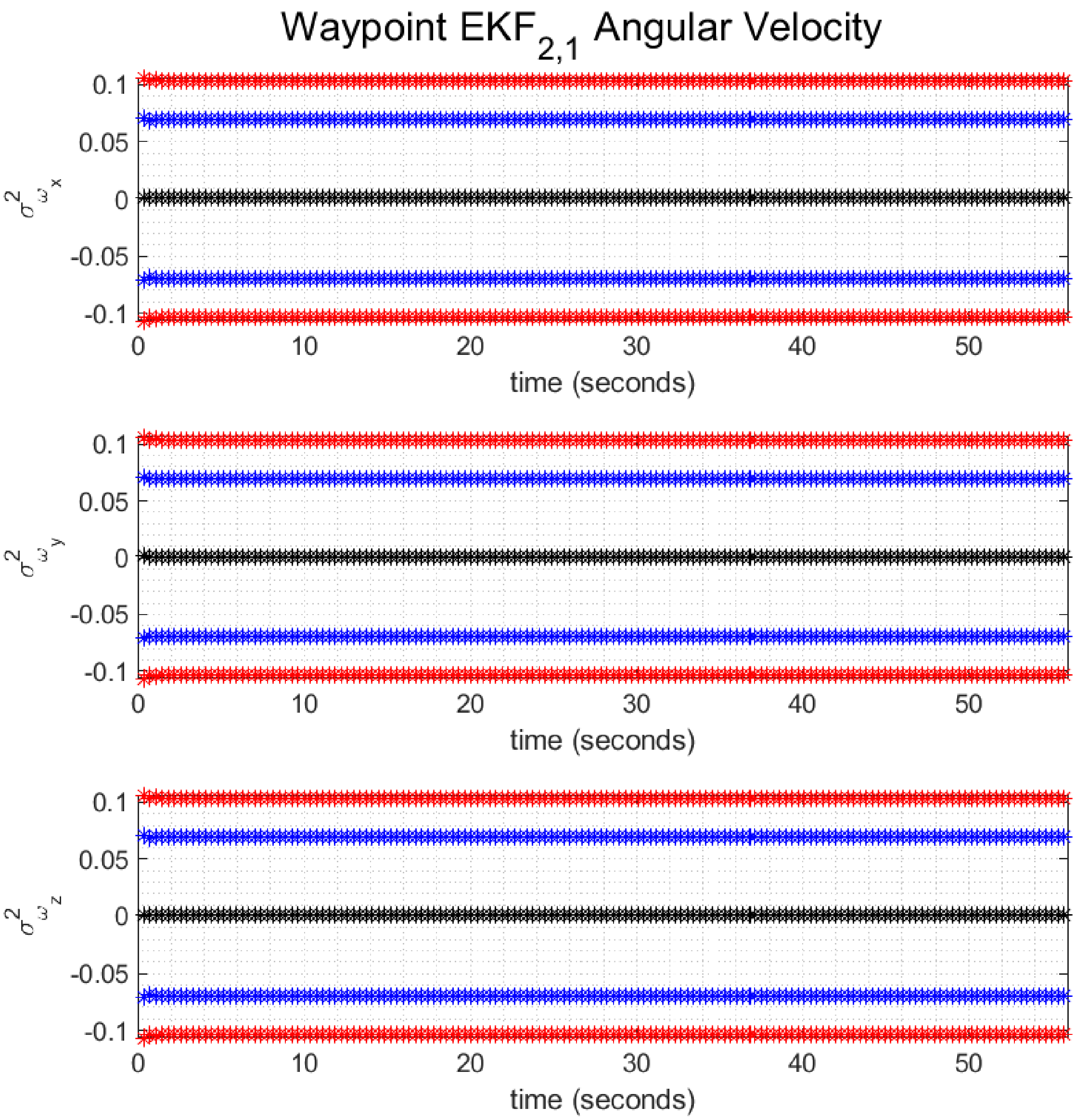


Figure 7.16: Waypoint $EKF_{2,1}$ Angular Velocity Variance

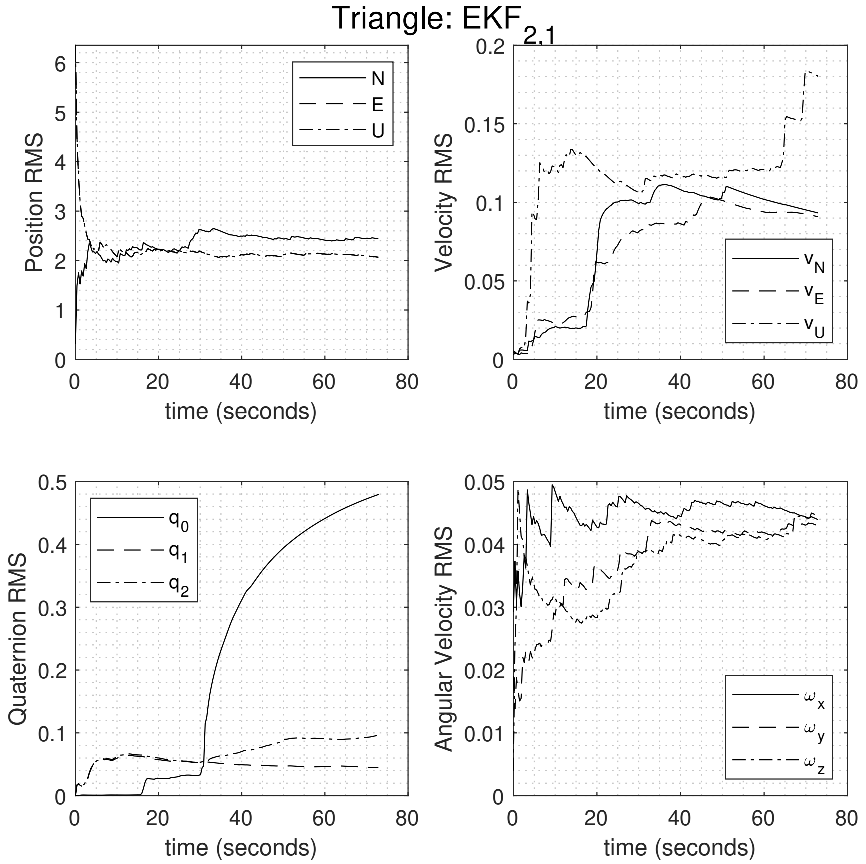


Figure 7.17: Triangle $EKF_{2,1}$ RMS

Figure 7.18 shows the RMS profiles of $EKF_{2,1}$ of the waypoint maneuver with respect to time. Just like the triangle maneuver, the RMS values converge for position and angular velocity. The RMS values for v_U tend to diverge. The RMS values for q_0 seem to decrease, but they do not converge sufficiently fast in the allotted time.

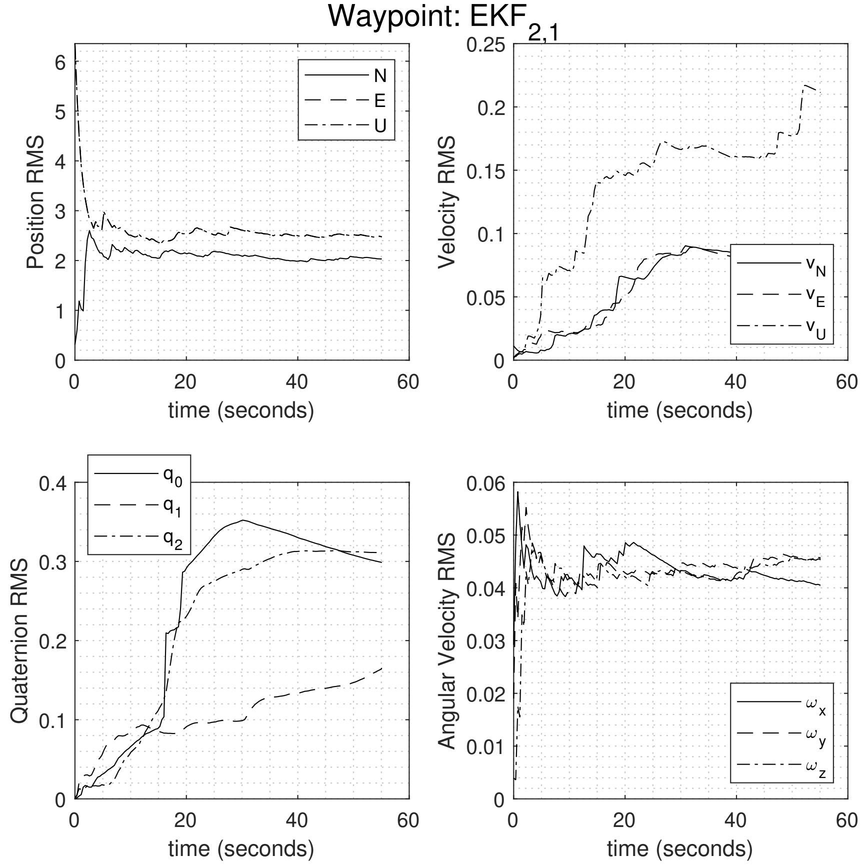


Figure 7.18: Waypoint $EKF_{2,1}$ RMS

7.9 Concluding Remarks

Overall, the HME framework with acoustics demonstrates a viable navigation solution for UAVs. The Burgers and Staples dynamic thrust models are not accurate, but the HME simulation provides a broad approach by considering different models and parameters. This leads to increased accuracy, which affects control and guidance. Thus, the next chapter will tie extremal control and explicit guidance together with HME to create an integrated TGNC system for quadcopters.

CHAPTER 8
INTEGRATED TARGETING, GUIDANCE, NAVIGATION, AND CONTROL

This chapter follows the dynamical model and TGC approach in Ref. [124], which uses the Hamiltonian formalism to determine extremal control following the indirect (adjoint) method and translational E Guidance to satisfy the boundary conditions. The navigation solution uses the same HME approach in Ref. [122]. Of the three maneuvers presented, the waypoint guidance maneuver demonstrates integrated targeting, guidance, navigation, and control.

8.1 Targeting

For simplicity of the maneuvers and integration, pilots usually determine and load waypoints before the mission. During the mission, the remaining distance to the waypoint based on GPS measurements feeds into E Guidance, and Lin says that in targeting, "position sensor data must be available...to locate the pursuer with respect to the target" [114]. The DJI OSDK function, *positionAndYawCtrl*, requires feedback of the remaining distance to the waypoint, so it uses GPS measurements to determine the remaining distance to the waypoint. The remaining distance to the waypoint feeds into the position command through *positionAndYawCtrl*, which also uses the position from E Guidance. However, numerous intermediate target points along the nominal trajectory can be autonomously determined onboard in real-time.

The target states are determined based on the nominal trajectory, which is pre-computed on the ground using the dynamical model, and it is based on the maneuver characteristics, dynamic environment, and the convenience to the mission (maneuver) designer to be highly accurate. The onboard computer computes each target state as target points for the guidance function to steer the UAV towards the target. The target state computation repeats after reaching the next target point and stops until the UAV reaches the final target. The navigation function provides state estimation at every instance of time and feeds into the initial boundary conditions for E Guidance at each target point.

Figure 8.1 shows an example of this targeting strategy. The yellow dots indicate the initial and final points, while the red dots indicate intermediate target points. The quadcopter starts the waypoint maneuver at point 0 and computes the first target state on the nominal trajectory (green line) at point 1 (first red dot), which occurs approximately at the one-third mark of the trajectory. The target state's position and velocity feed into E Guidance as the desired target with the position and velocity at point 1 as the initial conditions for E Guidance. After reaching the first target point, it determines another target state, TS_2 (second red dot) and feeds this target state's position and velocity into E Guidance. The navigation function continuously estimates the state vector at each instance of time throughout the maneuver, and the state estimation at each target point feeds into the initial boundary conditions for the next E Guidance arc. The red arrows indicate how E Guidance would steer the UAV towards each target point, where slight deviations exist from the nominal trajectory. This process repeats until the UAV reaches the last target point (last yellow dot). Overall, increasing the number of target points increases the UAV's accuracy in following the nominal trajectory. The targeting approach is general and can apply to other maneuvers such as the closed-loop triangle maneuver (see Fig. 8.2).

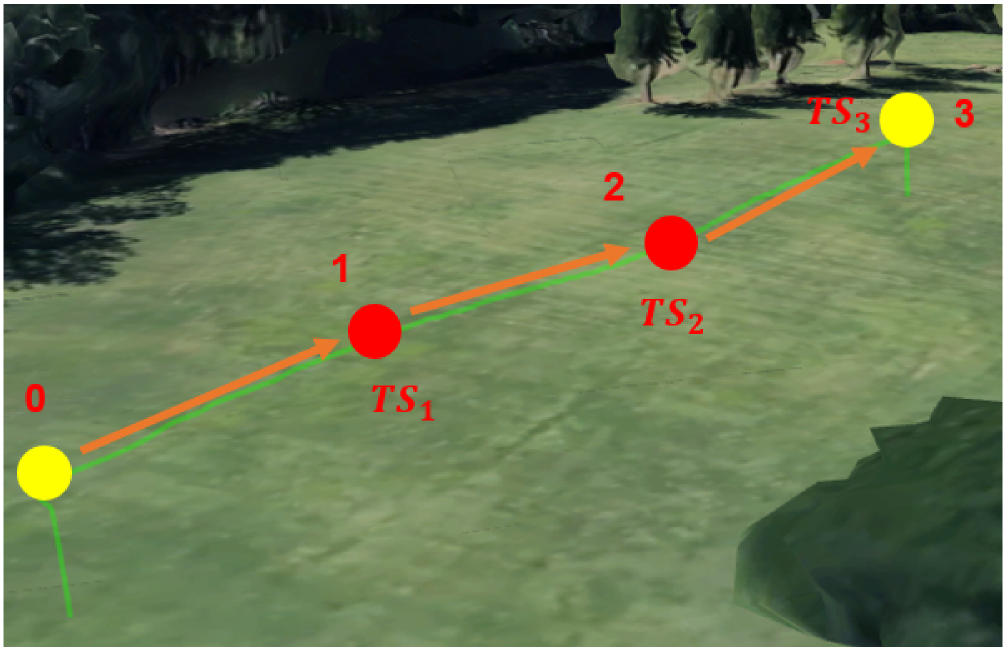


Figure 8.1: Waypoint Targeting Example

However, a potential re-targeting procedure for completeness of targeting has the UAV compute targets onboard based on the nominal trajectory and in environments with dynamic obstacles. In this case, the UAV must determine new target states based on sensors that detects obstacles, so it must compute new targets to avoid potential collisions. This provides a framework for real-time re-targeting through sensor data and guidance as part of the integrated TGNC. Overall, increasing the number of target points increases the UAV's chances of avoiding dynamics obstacles. Simulations of the integrated targeting and re-targeting procedures requires a set of studies outside the scope of the proposed research, so it can be considered as a potential future research topic.

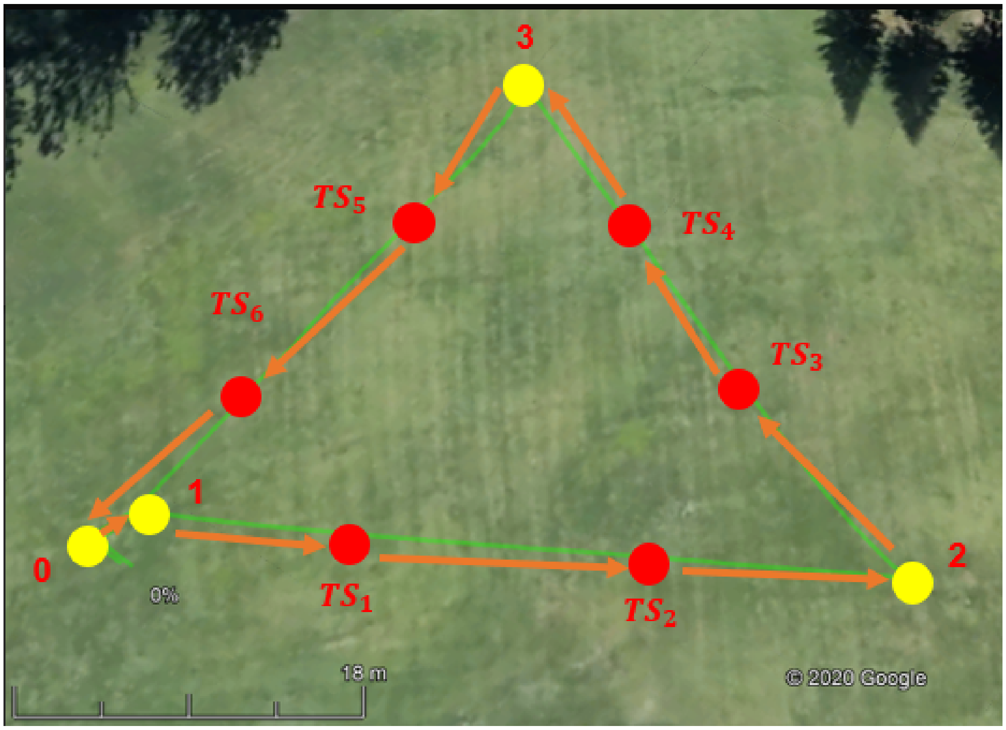


Figure 8.2: Triangle Targeting Example

8.2 Guidance

The and guidance for the waypoint guidance maneuver follows the translational explicit guidance approach in Ref. [124]. For the reader's convenience, this section summarizes and highlights the main targeting and guidance results for the waypoint guidance manevuer in Ref. [124]. The target point is located approximately -71 m south, 100 m east, and 10 m up from the takeoff maneuver, which ends at an altitude of 20 m. Thus, the waypoint guidance maneuver follows the takeoff maneuver and has these boundary conditions:

$$\boldsymbol{p}_0 = \begin{bmatrix} 0 \\ 0 \\ 20 \end{bmatrix}, \ \boldsymbol{v}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \ \boldsymbol{p}_D = \begin{bmatrix} -71 \\ 100 \\ 30 \end{bmatrix}, \ \boldsymbol{v}_D = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{8.1}$$

The waypoint guidance maneuver uses these polynomials for translational E Guidance:

$$p_1(t) = p_3(t) = p_5(t) = \tau^2 \, , \ p_2(t) = p_4(t) = p_6(t) = \tau^3.$$

This selection of polynomials yields accelerations of:

$$a_x(t) = c_1\tau^2 + c_2\tau^3 - g_x(t) \, , \ a_y(t) = c_3\tau^2 + c_4\tau^3 - g_y(t) \, , \ a_z(t) = c_5\tau^2 + c_6\tau^3 - g_z(t). \tag{8.2}$$

Using $T_{go} = 25$ seconds and the boundary conditions from Eq. (8.1) gives the coefficients for the accelerations with subscript WG to denote waypoint guidance:

$$c_{1,WG} = 0.0109\,,\ c_{2,WG} = -5.8396 \cdot 10^{-4}\,,\ c_{3,WG} = -0.0154$$
$$,\ c_{4,WG} = 8.2019 \cdot 10^{-4}\,,\ c_{5,WG} = -0.0015\,,\ c_{6,WG} = 8.1920 \cdot 10^{-5}.$$

8.3 Control

The control for the waypoint guidance manevuer follows the approach in Ref. [122] with integrated extremal control and translational explicit guidance. For the reader's convenience, this section highlights the main extremal control results of the waypoint guidance from Ref. [124].

Motor Spin Rates. The closed-form solutions of the controls (ω_i) are:

$$\begin{aligned}
\omega_1^2 &= \left(\frac{m}{k}\right)\frac{a_U + k_{dz}v_U}{q_0^2 - q_1^2 - q_2^2 + q_3^2} + \frac{\tau_\psi}{4b} + \frac{\tau_\phi}{2\ell k}\,,\ \omega_2^2 = \left(\frac{m}{k}\right)\frac{a_U + k_{dz}v_U}{q_0^2 - q_1^2 - q_2^2 + q_3^2} - \frac{\tau_\psi}{4b} + \frac{\tau_\theta}{2\ell k}\,,\\
\omega_3^2 &= \left(\frac{m}{k}\right)\frac{a_U + k_{dz}v_U}{q_0^2 - q_1^2 - q_2^2 + q_3^2} + \frac{\tau_\psi}{4b} - \frac{\tau_\phi}{2\ell k}\,,\ \omega_4^2 = \left(\frac{m}{k}\right)\frac{a_U + k_{dz}v_U}{q_0^2 - q_1^2 - q_2^2 + q_3^2} - \frac{\tau_\psi}{4b} - \frac{\tau_\theta}{2\ell k}.
\end{aligned} \tag{8.3}$$

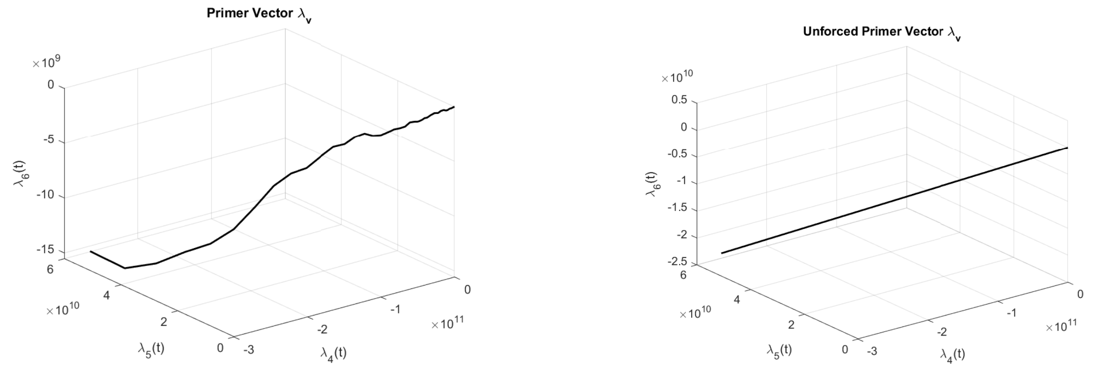
It takes only approximately $7.3215 \cdot 10^{-2}$ milliseconds to compute each extremal control value, so onboard implementation of this extremal controller is feasible for real-time applications.

8.3.1 Extremality and Lagrange Multipliers

The integrated TGNC waypoint guidance maneuver uses the nontrivial variable motor thrust case from Ref. [124], which generates a field of extremals with six non-zero Lagrange multipliers ($\lambda_1, ..., \lambda_6$) and six Lagrange multipliers ($\lambda_7, ..., \lambda_{12}$) equal to zero as shown in Fig. 5.1. Extremality occurs by setting λ_6 as a function of λ_4 and λ_5, which ensures $\frac{\partial H}{\partial \omega_i} = 0\ \forall t$ with H being the Pontryagin function:

$$\lambda_6(t) = \frac{-\lambda_4(t)(q_0q_2 + q_1q_3) - \lambda_5(t)(q_2q_3 + q_0q_1)}{q_0^2 - q_1^2 - q_2^2 + q_3^2}. \tag{8.4}$$

Thus, the guided trajectory is an extremal trajectory. If λ_6 was not forced in this manner, a feasible trajectory would exist instead of an extremal trajectory. Having $\lambda_6(t)$ unforced gives a straight line for the primer vector, which validates primer vector theory for rocket dynamics in a uniform gravity field.



(a) Primer Vector

(b) Unforced Primer Vector

Figure 8.3: Comparing Forced and Unforced Primer Vectors

8.3.2 Satisfying the First-Order Necessary Conditions of Optimality

The general form of the first differential of the functional J is [76]:

$$\begin{aligned} dJ &= (\Gamma_{\boldsymbol{s}_0} + \boldsymbol{\lambda}_0^T)^T d\boldsymbol{s}_0 + (\Gamma_{\boldsymbol{s}_f} - \boldsymbol{\lambda}_f^T)^T d\boldsymbol{s}_f + (\Gamma_{t_0} - H_0)dt_0 + (\Gamma_{t_f} + H_f)dt_f \\ &+ \int_{t_0}^{t_f} \left[(H_{\boldsymbol{s}} + \dot{\boldsymbol{\lambda}}^T)^T \delta\boldsymbol{s} + (H_{\boldsymbol{\lambda}} - \dot{\boldsymbol{s}}^T)^T \delta\boldsymbol{\lambda} + H_{\boldsymbol{u}}^T \delta\boldsymbol{u} + H_{\boldsymbol{\alpha}}^T \delta\alpha \right] dt \,, \end{aligned} \tag{8.5}$$

where 0 and f denote initial and final respectively. The Γ terms are defined as:

$$\begin{aligned} \Gamma_{t_0} &= \epsilon \frac{\partial J}{\partial t_0} + \boldsymbol{\mu}^T \frac{\partial \boldsymbol{E}}{\partial t_0} \,, \qquad \Gamma_{t_f} = \epsilon \frac{\partial J}{\partial t_f} + \boldsymbol{\beta}^T \frac{\partial \boldsymbol{F}}{\partial t_f} \,, \\ \Gamma_{s_0} &= \epsilon \frac{\partial J}{\partial \boldsymbol{s}_0} + \boldsymbol{\mu}^T \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{s}_0} \,, \qquad \Gamma_{s_f} = \epsilon \frac{\partial J}{\partial \boldsymbol{s}_f} + \boldsymbol{\beta}^T \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{s}_f} \end{aligned} \tag{8.6}$$

with $\boldsymbol{E}$ to denote the initial constraint equations and $\boldsymbol{F}$ to denote the final constraint equations. Satisfying the first-order necessary conditions of optimality means that $dJ = 0$. The first differential for the waypoint guidance maneuver is:

$$\begin{aligned} dJ_{WG} &= (\Gamma_{\boldsymbol{s}_{WG,0}} + \boldsymbol{\lambda}_{WG,0}^T)^T d\boldsymbol{s}_0 + (\Gamma_{\boldsymbol{s}_{WG,f}} - \boldsymbol{\lambda}_{WG,f}^T)^T d\boldsymbol{s}_f \\ &+ (\Gamma_{t_{WG,0}} - H_{WG,0})dt_0 + (\Gamma_{t_{WG,f}} + H_{WG,f})dt_f \\ &+ \int_{t_0}^{t_f} \left[(H_{\boldsymbol{s}} + \dot{\boldsymbol{\lambda}}^T)^T \delta\boldsymbol{s} + (H_{\boldsymbol{\lambda}} - \dot{\boldsymbol{s}}^T)^T \delta\boldsymbol{\lambda} + H_{\boldsymbol{u}}^T \delta\boldsymbol{u} + H_{\boldsymbol{\alpha}}^T \delta\alpha \right] dt. \end{aligned} \tag{8.7}$$

All the terms in Eq. (8.5) must be zero for $dJ = 0$:

$$\begin{aligned}
&\Gamma_{t_{WG,0}} = \epsilon = H_{WG,0} \text{ , } \Gamma_{\boldsymbol{s}_{WG,0}} = \boldsymbol{\mu}_{WG} = \boldsymbol{\lambda}_{WG,0}^{T} = \begin{bmatrix} \lambda_7(t_{WG,0}) \\ \lambda_8(t_{WG,0}) \\ \lambda_9(t_{WG,0}) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \text{ ,} \\
&\Gamma_{t_{WG,f}} = \epsilon = -H_{WG,f} \text{ , } \Gamma_{\boldsymbol{s}_{WG,f}} = \boldsymbol{\beta}_{WG} = \boldsymbol{\lambda}_{WG,f}^{T} = \begin{bmatrix} \lambda_7(t_{WG,f}) \\ \lambda_8(t_{WG,f}) \\ \lambda_9(t_{WG,f}) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \text{ ,} \\
&\quad H_{\boldsymbol{s}} = -\dot{\boldsymbol{\lambda}}^{T} \text{ , } H_{\boldsymbol{\lambda}} = \dot{\boldsymbol{s}}^{T} \text{ , } H_{\boldsymbol{u}} = 0 \text{ , } H_{\boldsymbol{\alpha}} = 0.
\end{aligned} \tag{8.8}$$

From the takeoff maneuver, $H_{TO,f} = H_{WG,0} = \epsilon = 0$, and the Pontryagin function at the end of the waypoint guidance maneuver, $H_{WG,f}$, is also zero. The quadcopter ends the waypoint guidance maneuver with a heading of $\psi = 125°$, which yields $q_0 = 0.462, q_1 = q_2 = 0, q_3 = 0.887$. The velocity components are zero, and $\lambda_6 = 0$ from its dependence on λ_4 and λ_5,. Therefore, the Hamiltonian is zero, as expected. The canonical equations and local optimality equations cause the remaining equations in the integrand of Eq. (8.8) to be zero. Therefore, all the terms in Eq. (8.7) are zero, so the waypoint guidance maneuver has an extremal trajectory. This dissertation does not consider second-order conditions, so an optimal trajectory for the waypoint guidance maneuver cannot be claimed, i.e., at best, only extremal. Future studies may include second-order conditions to determine if the waypoint guidance maneuver yields optimal control with a corresponding optimal trajectory.

8.4 Navigation

For convenience and simplicity, the integrated TGNC waypoint guidance maneuver uses the same navigation approach and HME structure diagram from Ref. [122] but also compares the performance of the error covariance with off-diagonal components. The Burgers model yields 78 non-zero uncertainty covariances, while the Staples model yields 72 non-zero uncertainty covariances. The EKFs with acoustics utilize simulated SPL measurements based on the motor angular velocities from Ref. [124] with a factor of $\frac{1}{\sqrt{5}}$ to yield similar values based on the flight test data with experimental SPL measurements. This fudge factor acts as a tuning parameter because the motor spin rates from Ref. [124] utilize a static thrust model, which does not capture all the dynamics throughout maneuvers. As shown earlier, the Burgers and Staples dynamic thrust models do not accurately capture the dynamics either. Future studies may encompass researching dynamic thrust models that closely model quadcopter dynamics for various maneuvers such as the waypoint guidance maneuver, which is very common. The integrated TGNC software in this chapter uses the same HME structure shown in Fig. 7.1.

Gating Network Weights. Figure 8.4 shows the gating networks weights of bank 1 and bank

2, and the weights sum to one, which provides a symmetric or reflective result. Zooming in on the first iteration shows that it is a tie at 50-50. However, the gating network quickly converges and chooses the EKFs with acoustics ($EKF_{2,1}$ & $EKF_{2,2}$) as the best performers due to the smaller errors.

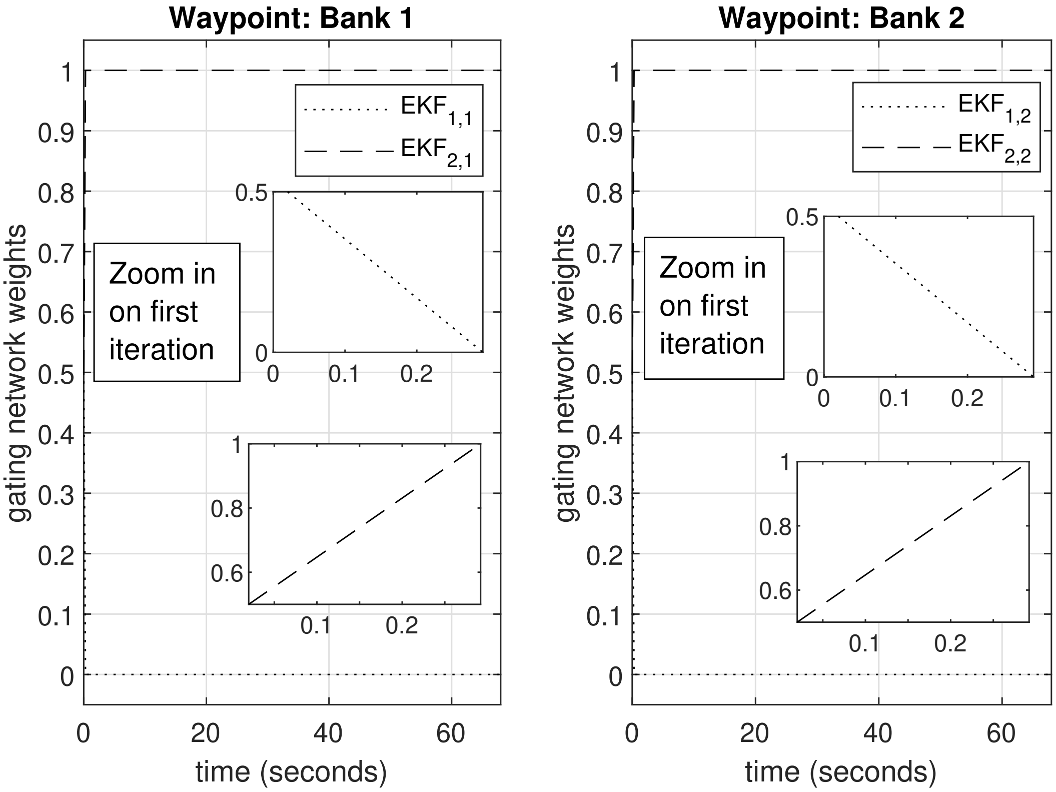


Figure 8.4: Waypoint Bank Comparison of Gating Network Weights

Figures 8.6 and 8.7 include the off-diagonal error covariance components of **P**. The gating network still prefers the filters with acoustics, but there is a divergence problem in the 95 element of **P**, which is a cross-correlation between q_2 and v_E. Unfortunately, it is unknown why this divergence occurs, so the plot stops at approximately 42 seconds. Even if this $\mathbf{P}_{95}$ divergence was resolved, the runtime per cycle ranges from 10-16 seconds. Thus, near real-time implementation is not feasible when including the off-diagonal components due to numerous integrations per iteration.

Accuracy. The EKF state estimation accuracy uses mean square error (MSE) and root mean square (RMS) in the tables in this subsection. Some values repeat due to the same Q components and measurements, and the next subsection discusses covariances.

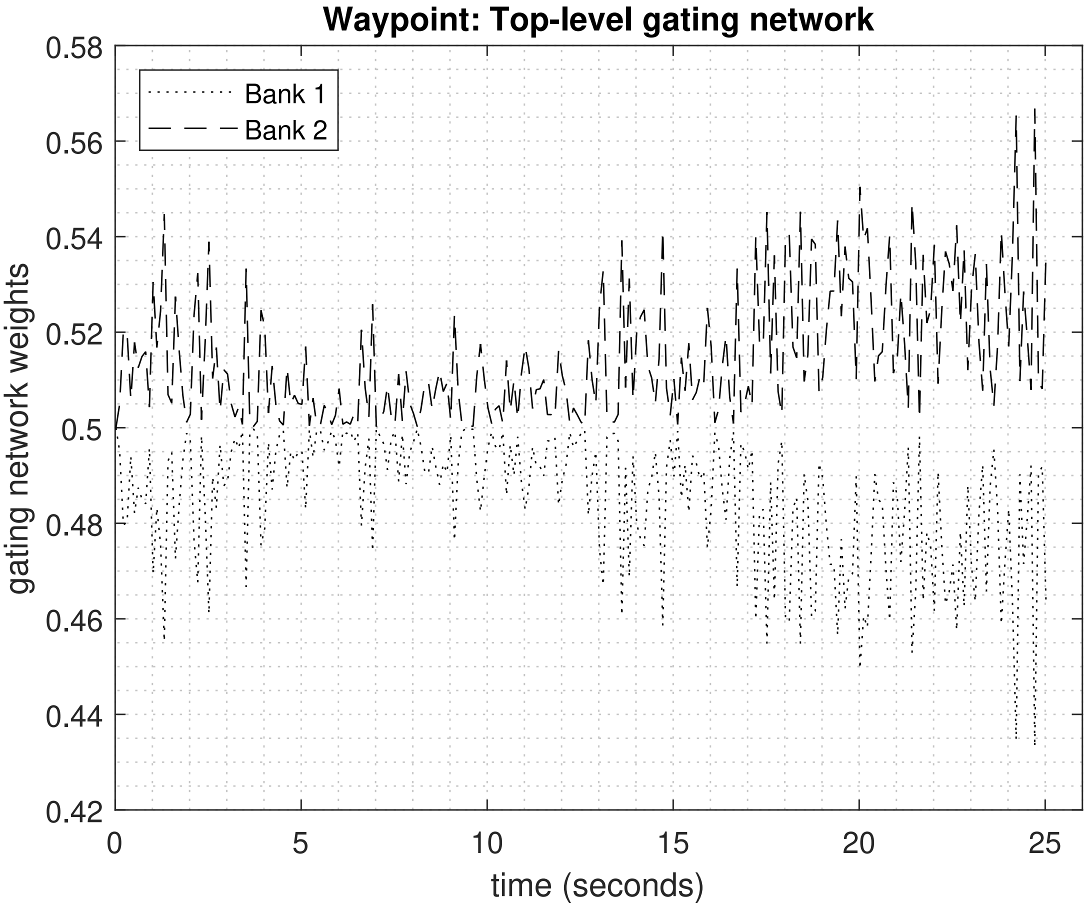


Figure 8.5: Waypoint Top-Level Gating Network Weights

Table 8.1: MSE and RMS Summary of Position and Velocity

	N	E	U	v_N	v_E	v_U
MSE	0.7972	1.1216	0.0442	6.3545	12.6050	0.2416
RMS	0.8929	1.0591	0.2101	2.5208	3.5503	0.4915

Table 8.2: MSE and RMS Summary of Quaternions and Angular Velocity

	q_0	q_1	q_1	ω_x	ω_y	ω_z
MSE	0.0011	$5.4219 \cdot 10^{-4}$	$4.4787 \cdot 10^{-4}$	$3.3543 \cdot 10^{-4}$	$3.6359 \cdot 10^{-4}$	$3.8518 \cdot 10^{-4}$
RMS	0.0334	0.0233	0.0212	0.0183	0.0191	0.0196

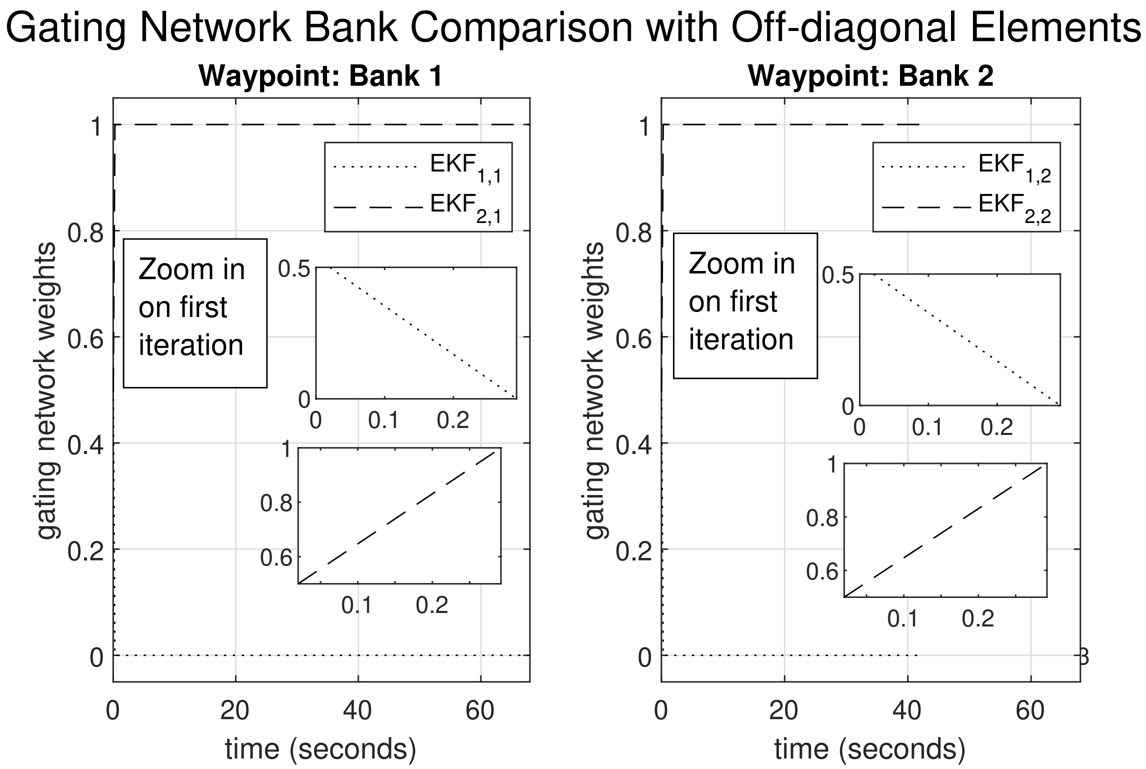


Figure 8.6: Waypoint Bank Comparison of Gating Network Weights with Cross-correlation

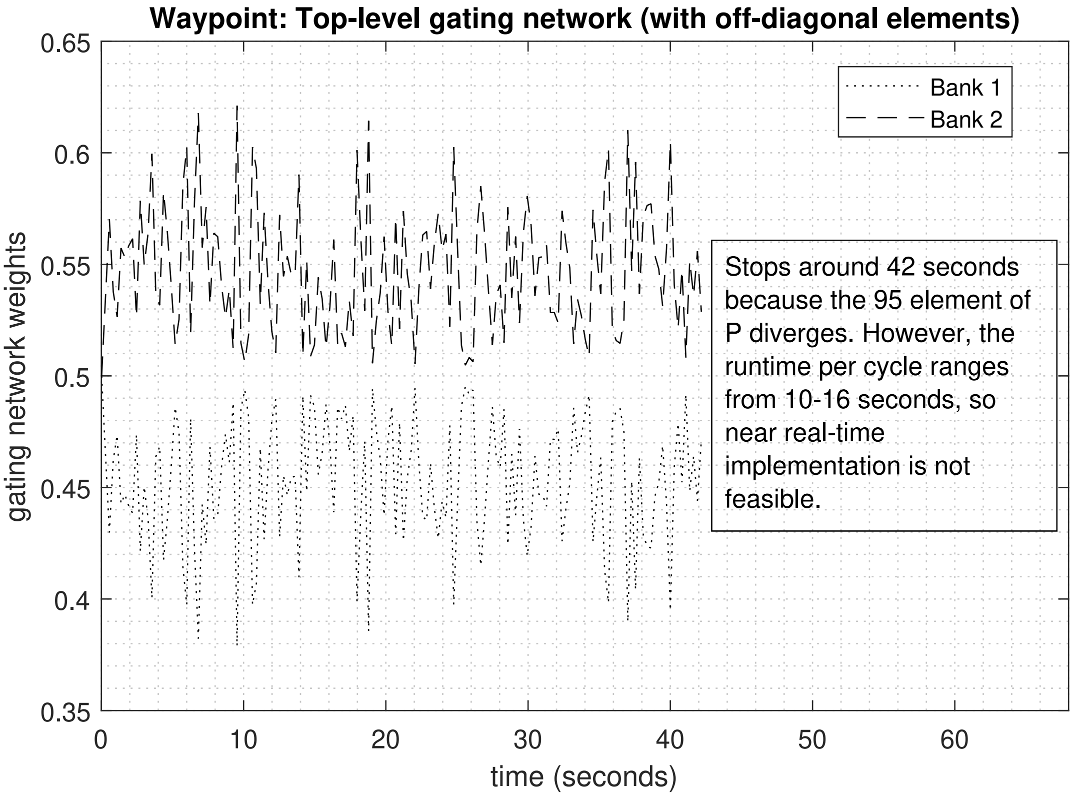
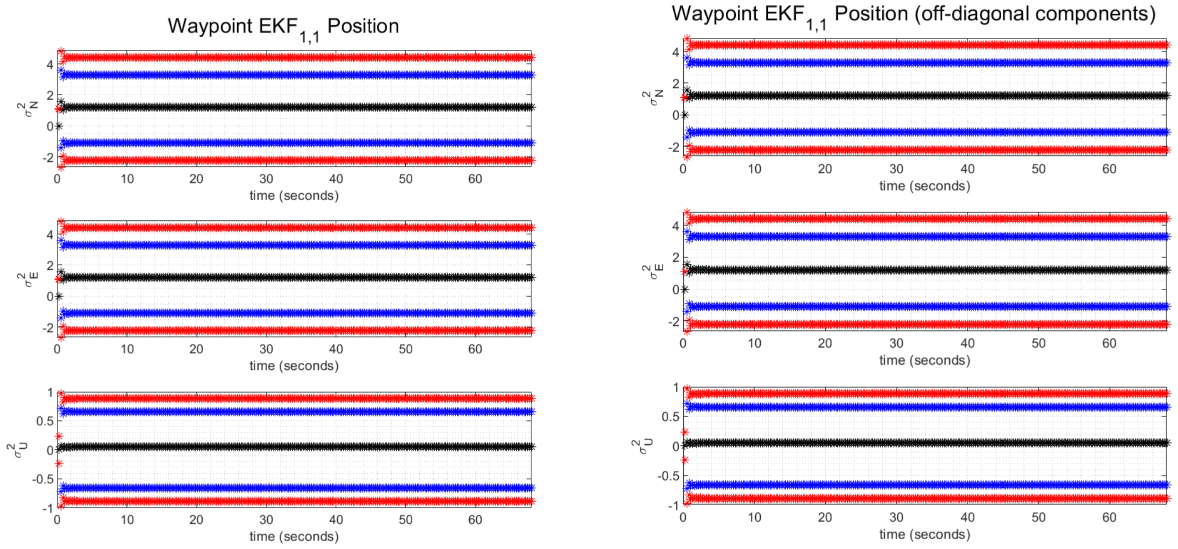


Figure 8.7: Waypoint Top-Level Gating Network Weights with Cross-correlation

8.4.1 Hierarchical Mixture of Experts Extended Kalman Filter Covariances

First Extended Kalman Filter. Figure 8.8a shows the variance of position vs. time for the waypoint maneuver's first extended Kalman filter, $EKF_{1,1}$, which is in bank 1 and uses the Burgers thrust model. The position variance quickly converges to a fixed value due to GPS measurements, so uncertainty in position is bounded by staying within the $\pm 2\sigma$ (blue lines) and $\pm 3\sigma$ (red lines) bounds, which center around the position variance by adding the mean error.



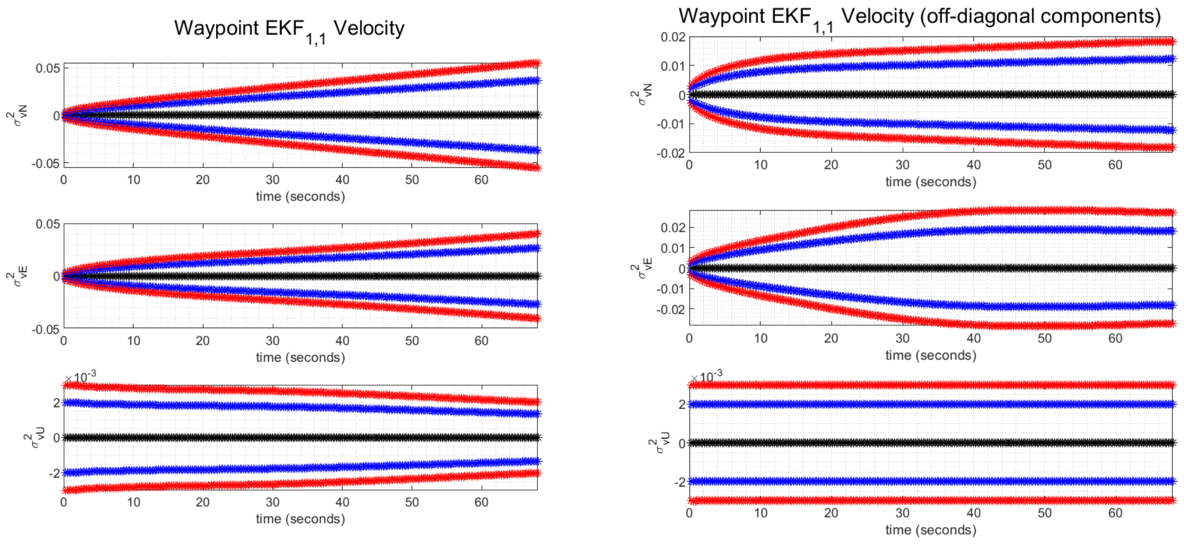
(a) Position Variance (b) Position Variance with Cross-correlation

Figure 8.8: Waypoint $EKF_{1,1}$ Position Variance

Figure 8.8b shows the position covariances with the cross-correlation covariances. Comparing Figures 8.8a- reffig:tgnc wpt EKF1 position variance diagonal shows no significant differences, so including cross-correlation covariances does not increase the accuracy or confidence in the position estimations. Figure 8.9a shows the variance of velocity vs. time for the waypoint maneuver's first EKF, $EKF_{1,1}$. The velocity variance for v_U converges, while the velocity variances for v_N and v_E diverge. Figure 8.10a shows a zoomed-in version of the velocity variances, which demonstrates true divergence for v_N and v_E [118]. Even though the velocity variances lie within the $\pm 2\sigma$ and $\pm 3\sigma$ bounds, the lack of velocity measurements, having simple dynamics thrust models with simplifying assumptions, and neglecting cross-correlation covariances are the three main reasons that cause the uncertainties and inaccuracies to grow in time.

Including the cross-correlation terms affects the velocity covariances, as shown in Figures 8.9b-8.10b. The covariance for v_N with cross-correlation increases but slows down and levels off, while the covariance for v_N without cross-correlation increases faster with time. The covariance for v_E with cross-correlation increases but seems to plateau and decrease later, while the covariance for

v_E without cross-correlation only increases and exhibits true divergence.The covariance for v_U with cross-correlation converges, while the covariance for v_U without cross-correlation trended downwards. They remain within the $\pm 2\sigma$ and $\pm 3\sigma$ bounds, which diverge.



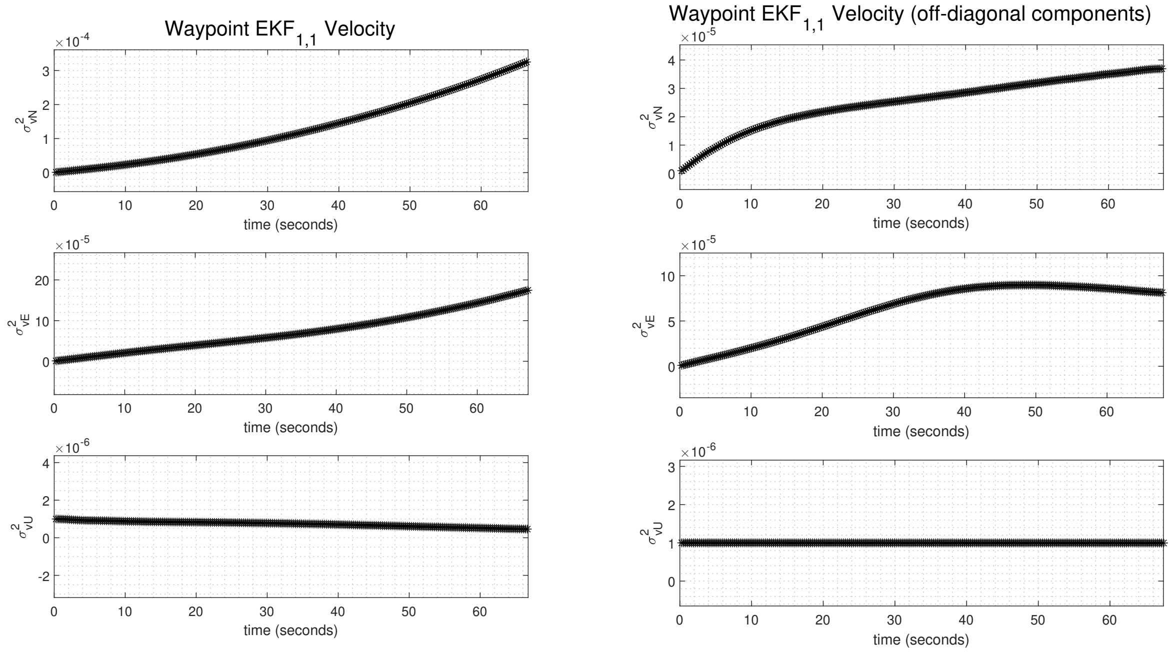
(a) Velocity Variance

(b) Velocity Variance with Cross-correlation

Figure 8.9: Waypoint $EKF_{1,1}$ Velocity Variance

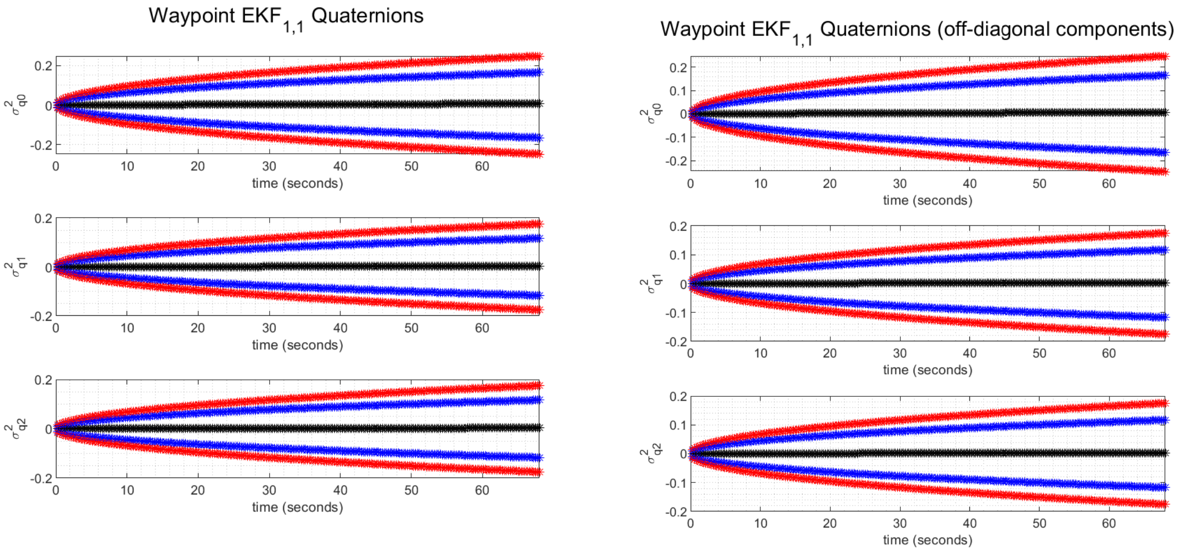
Figure 8.11a shows the variance of quaternion vs. time for the waypoint maneuver's first EKF, $EKF_{1,1}$. The quaternion variances diverge due to the lack of quaternion measurements and exceed the $\pm 2\sigma$ and $\pm 3\sigma$ bounds, which demonstrates true divergence and a lack of confidence in the quaternion estimations. The q_1 and q_2 variances are the same due to having the same process error variances, so the error covariance propagation equations produce the same values. Figure 8.11b shows the quaternion covariances with the cross-correlation terms, which shows a significant improvement by staying within the $\pm 2\sigma$ and $\pm 3\sigma$ bounds. Figures 8.12a - 8.12b show the quaternion variances zoomed in, and both diverge. Neglecting the cross-correlation terms causes the quaternion variances to exceed the initial covariances, while including including the cross-correlation terms causes the quaternion variances to stay within the initial covariances. Overall, the quadcopter attitude models are less accurate than the translation models, which leads to true divergence for quaternions.

Figure 8.13a shows the variance of angular velocity vs. time for the waypoint maneuver's second EKF, $EKF_{1,1}$. Figure 8.13b shows the angular velocity covariances with cross-correlation, which does not yield any significant increases in performance. Both angular velocity variances converge quickly, just like the position variances due to gyroscope measurements. Just as the position uncertainty was bounded within the $\pm 2\sigma$ and $\pm 3\sigma$ bounds, the uncertainty in angular



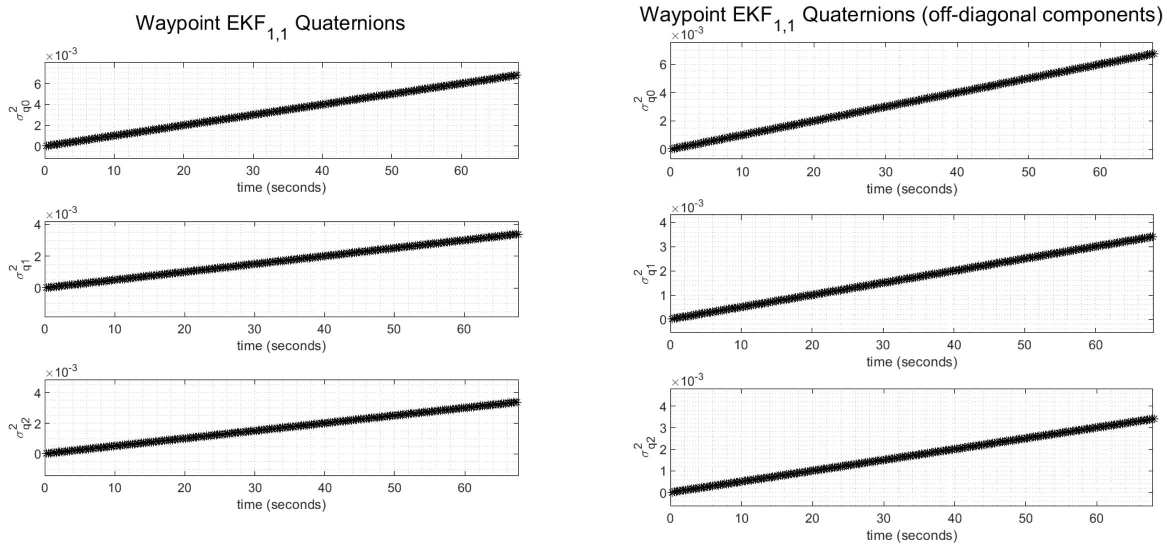
(a) Velocity Variance (b) Velocity Variance with Cross-correlation

Figure 8.10: Waypoint $EKF_{1,1}$ Velocity Variance Zoomed In



(a) Quaternion Variance (b) Quaternion Variance with Cross-correlation

Figure 8.11: Waypoint $EKF_{1,1}$ Quaternion Variance


Waypoint EKF$_{1,1}$ Quaternions

Waypoint EKF$_{1,1}$ Quaternions (off-diagonal components)

(a) Quaternion Variance (b) Velocity Quaternion with Cross-correlation

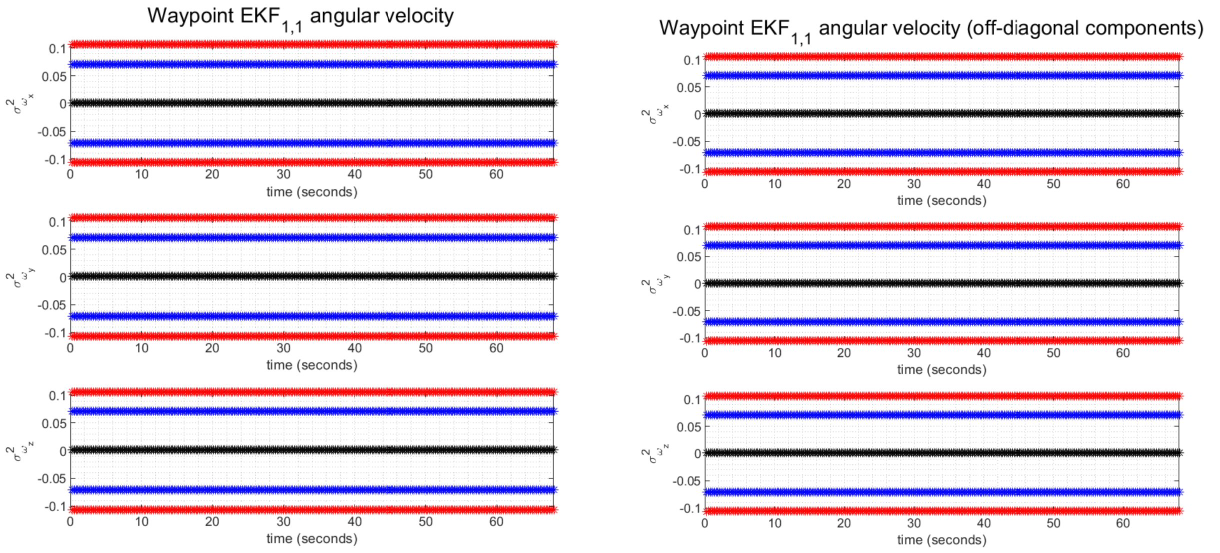
Figure 8.12: Waypoint $EKF_{1,1}$ Quaternion Variance Zoomed In

velocity is also bounded. Overall, there is a tradeoff between accuracy and computation time between including or neglecting the cross-correlation terms. There is a significant gain in performance for the quaternion estimations because the quaternion covariances with cross-correlation are bounded within $\pm 2\sigma$ and $\pm 3\sigma$. The velocity estimations are also more accurate due to slower divergence and convergence in v_U. However, it takes approximately 10 to 16 seconds per cycle with cross-correlation. Neglecting cross-correlation terms gives a much faster runtime per cycle at approximately 1.2 to 1.4 seconds, which demonstrates near real-time implementation. Since the position, velocity, and angular velocity estimations are within the $\pm 2\sigma$ and $\pm 3\sigma$ bounds without cross-correlation terms, neglecting the cross-correlation terms provides adequate estimation results for near real-time implementation. The other three EKFs demonstrate similar profiles and are not included to conserve space.

8.4.2 State Variable Estimation by Hierarchical Mixture of Experts

Position. Figure 8.14a shows the differences of the position profiles of the experimental flight test data, simulated HITL data, experimental PID flight test data, and the simulated HME estimation. The HME estimates are very accurate and closely follow the nominal trajectories. Figure 8.14b shows there is not a significant difference with cross-correlation terms in which the only difference is that HME stops abruptly due to the $\mathbf{P}_{95}$ divergence issue.

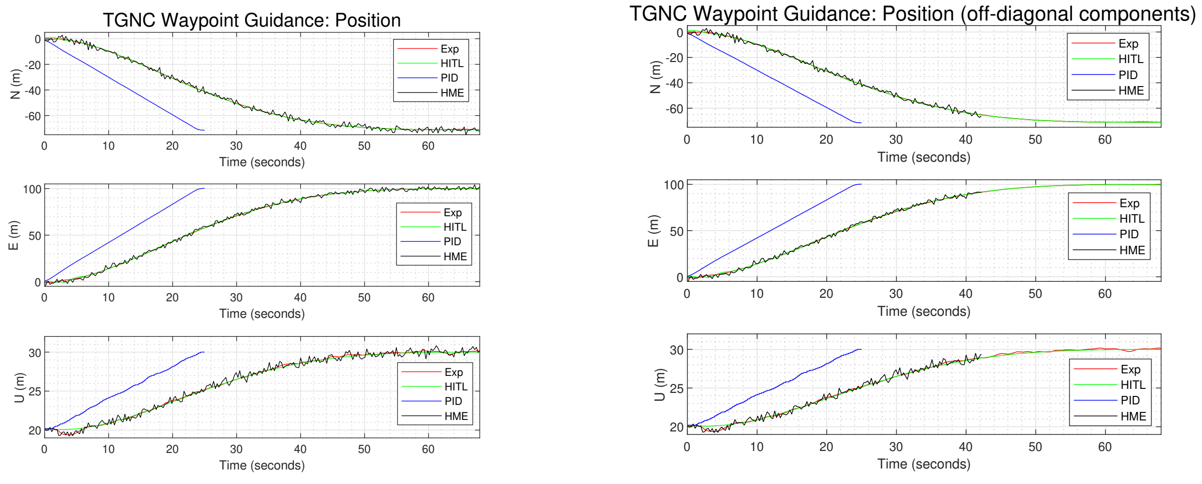
Velocity. Figure 8.15a shows the differences of the velocity profiles of the experimental flight test data, simulated HITL data, experimental PID flight test data, and the simulated HME esti-



(a) Angular Velocity Variance

(b) Angular Velocity Variance with Cross-correlation

Figure 8.13: Waypoint $EKF_{1,1}$ Angular Velocity Variance

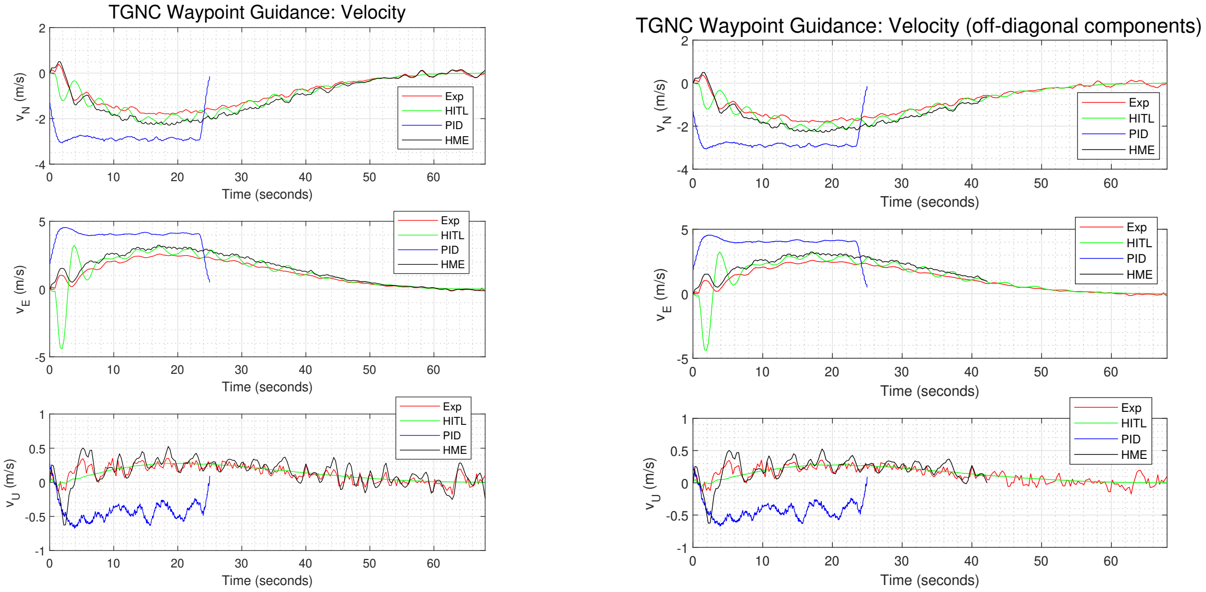


(a) Position Comparison

(b) Position Comparison with Cross-correlation

Figure 8.14: TGNC Position Comparison

mation. The HME estimates are fairly accurate and closely follow the experimental velocities. Figure 8.15b shows the velocity estimations with corss-correlation. It shows there is not a big difference between including or neglecting cross-correlation in velocity estimations despite the $\mathbf{P}_{95}$ divergence issue when including cross-correlation.

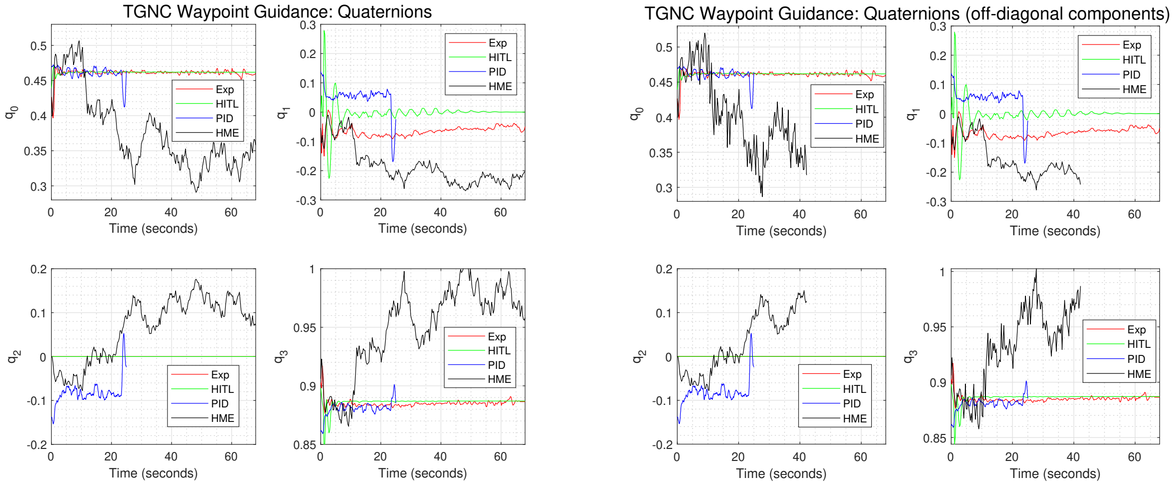


(a) Velocity Comparison (b) Velocity Comparison with Cross-correlation

Figure 8.15: TGNC Velocity Comparison

Quaternions. Figure 8.16a shows the differences of the quaternion profiles of the experimental flight test data, simulated HITL data, experimental PID flight test data, and the simulated HME estimation. The HME estimates are very accurate and closely follow the nominal trajectories, and q_2 in the experimental and HITL data remains zero. Even though including the cross-correlation covariances increases the confidence in the quaternion estimations, the overall quaternion estimation profiles are similar when comparing Figure 8.16a and 8.16b. The HME quaternion estimation stops due to the $\mathbf{P}_{95}$ element diverging.

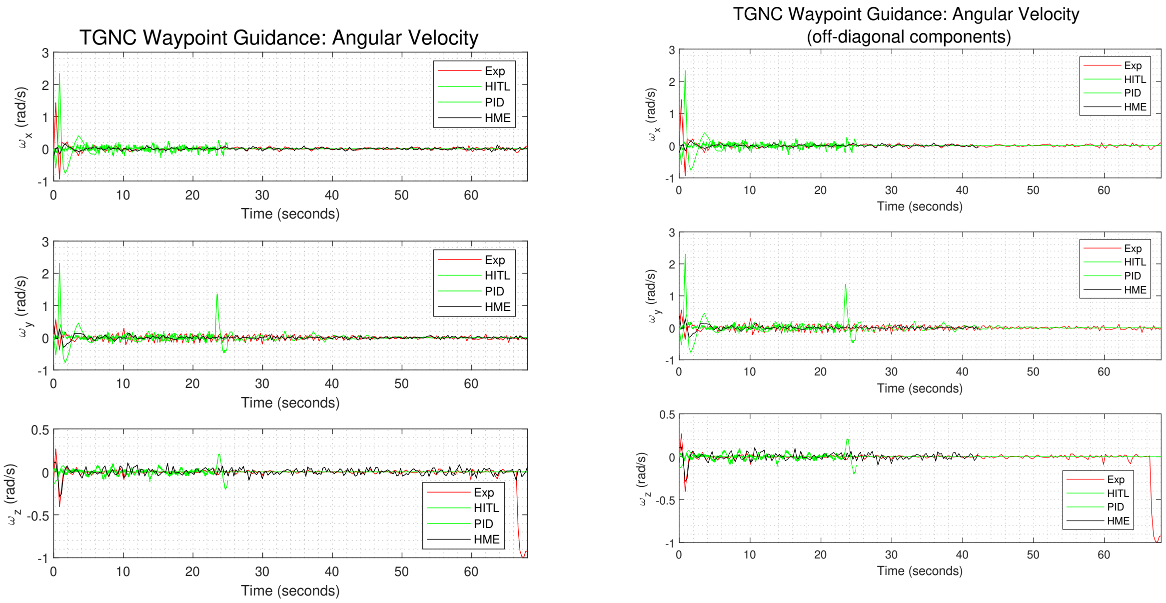
Angular Velocity. Figure 8.17a shows the differences of the angular velocity profiles of the experimental flight test data, simulated HITL data, experimental PID flight test data, and the simulated HME estimation. There is not a significant difference with cross-correlation covariances, as shown in Figure 8.17b. The HME estimates are very accurate and closely follow the nominal trajectories. In the experimental ω_z, there is a spike due to a slight twist upon landing on the uneven grass.



(a) Quaternion Comparison

(b) Quaternion Comparison with Cross-correlation

Figure 8.16: TGNC Quaternion Comparison



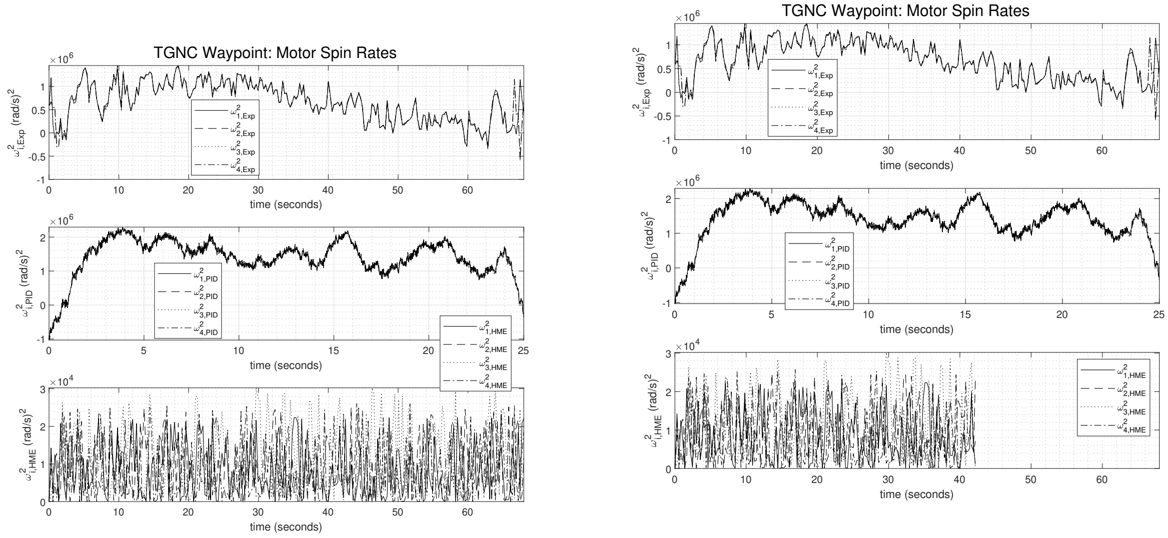
(a) Angular Velocity Comparison

(b) Angular Velocity Comparison with Cross-correlation

Figure 8.17: TGNC Angular Velocity Comparison

8.5 Comparison of Control Variables

Motor Spin Rate Comparison. Figure 8.18a shows the differences among the motor spin rate profiles of the experimental flight test data, experimental PID flight test data, and the simulated HME estimates. The PID motor spin rates are the highest and cause the quadcopter consume the most energy, which leads to shorter flight time. The HME acoustic motor spin rates tend to be slower by a factor of approximately 100 $(rad/s)^2$, which conserves the most energy. Overall, the acoustic model estimates lower values and does not yield any negative values because there are no negative SPL measurements. This leads to some modeling inaccuracies because it implies that the quadcopter motors will never spin in the opposite direction to decelerate. Contrarily, spinning in the opposite direction occurs in the E Guidance and PID motor spin rates as shown by the negative values at the beginning and end of the maneuver. The simulated HME estimates choose the motor spin rates based on acoustics. The motor spin rates with E Guidance are generally slower than the PID motor spin rates. Figure 8.18b shows the profiles of the motor spin rates with the cross-correlation terms of **P**, which does not yield any increase in accuracy or performance without the cross-correlation terms. Overall, there is no significant effect on the motor spin rates because the cross-correlation terms do not impact the motor spin rates computations.



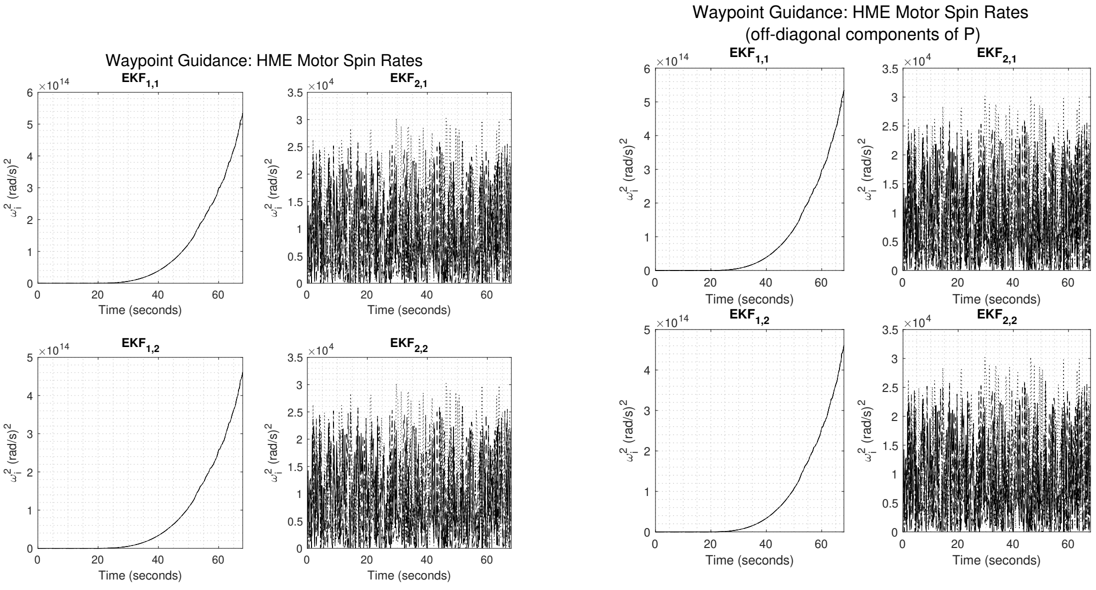
(a) TGNC Motor Spin Rate Comparison

(b) TGNC Motor Spin Rate Comparison with Cross-correlation

Figure 8.18: TGNC Motor Spin Rate Comparison

HME Motor Spin Rate Comparison. Figure 8.19a shows the differences among the HME motor spin rate profiles. The motor spin rates for $EKF_{2,1}$ and $EKF_{2,2}$ are the same because they use the same acoustic measurements to estimate the motor angular velocities. The motor spin rates in $EKF_{1,1}$ uses the Burgers thrust model and diverges to a high value of $1.1328 \cdot 10^{17}$ $(rad/s)^2$.

The Staples thrust model also diverges to a slightly smaller value of $9.7723 \cdot 10^{16}$ $(rad/s)^2$. For the acoustic EKFs, the motor spin rates in $EKF_{2,1}$ and $EKF_{2,2}$ peak at $2.2434 \cdot 10^4$ $(rad/s)^2$. The Burgers and Staples models do not model the quadcopter thrust accurately because they diverge and significantly exceed the maximum motor spin rate values.



(a) HME Motor Spin Rates

(b) HME Motor Spin Rates with Cross-correlation

Figure 8.19: TGNC HME Motor Spin Rate Comparison

Motor Spin Rate Analysis. Approximating the max motor angular velocity involves utilizing the motor's KV (rpm/V) rating and the quadcopter's battery voltage. Multiplying KV and the battery voltage gives rpm. Converting rpm to rad/s with a factor of $\frac{2\pi \, rad}{60s}$ and then squaring gives the approximate max motor spin rate. The DJI M100 has a 6S battery with a max voltage of 26.1 V (4.35 V/cell) or a nominal voltage of 25.8 (4.3 V/cell). Using the max voltage and DJI KV rating of 350 rpm/V gives:

$$\omega_{i,max}^2 \approx (350\frac{rpm}{V} \cdot 26.1V \cdot \frac{2\pi}{60})^2 = 915,112\frac{rad}{s}^2 \tag{8.9}$$

This approximate max motor spin rate is considerably smaller than the values from the Burgers and Staples thrust models, i.e., order of 10^{16} or 10^{17} $(rad/s)^2$. However, the motor spin rate estimates from acoustics is within a physically possible range, i.e., order of 10^4 $(rad/s)^2$. The motor spin rates with E Guidance peak at $1.4201 \cdot 10^6$ $(rad/s)^2$, which is outside the physically possible range but much closer than the Burgers and Staples thrust models. Recall that the motor spin rates with E Guidance utilize a static thrust model, which is not valid in dynamic conditions [100]. Contrarily, it is the static thrust model that demonstrates higher accuracy than the Burgers and Staples dynamic thrust models.

8.6 TGNC Integration

Figure 8.20 shows a block diagram for the integration of the targeting, guidance, navigation, and control functions described in this chapter, and it resembles the proposed TGNC block diagram in Fig. 1.1. Table 8.3 shows the inputs and outputs of the TGNC functions. Recall that the HME results were simulated, and HME could not be implemented onboard because the DJI M100's Raspberry Pi cannot read the NoiseLab-Lite SPL measurements in real-time.

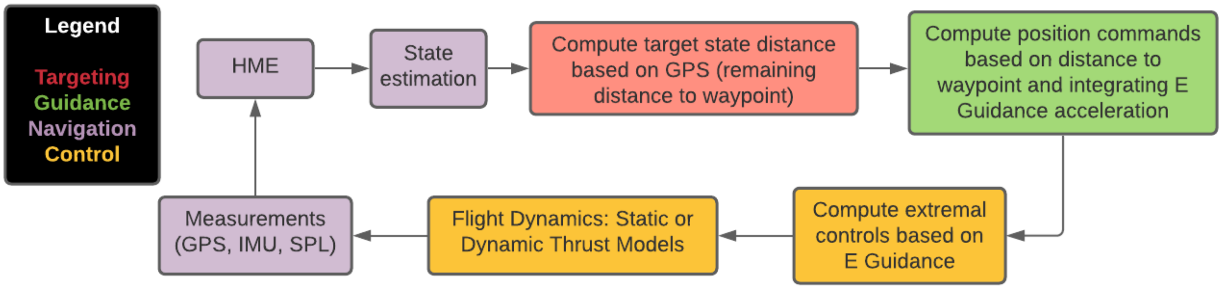


Figure 8.20: Proposed Integrated TGNC Block Diagram for Waypoint Guidance Maneuver

Table 8.3: Inputs and Outputs of TGNC Functions

Name	Inputs	Outputs
Targeting	State Estimation	Target States
Guidance	Distance to Waypoint (Target Point)	EG Position Commands
Control	EG Position Commands	Motor Spin Rates
Navigation	Measurements (GPS, IMU, SPL)	State Estimation

Applying TGNC to Fixed-wing Aircraft. Overall, the TGNC results do not easily translate to fixed-wing aircraft. First, the state vector and E Guidance would be the same. Second, HME would be different because of the different dynamical model equations. Third, the Pontryagin function would have a different form, which would lead to different extremal control laws and a different control vector with less motors but would possibly include the control surfaces such as ailerons, elevators, and rudder. Targeting and onboard implementation would be different because DJI does not have fixed-wing aircraft at the time of this writing. Staples has a pure pursuit controller function with quadcopters using a look ahead or lead parameter to determine target points further ahead on the nominal trajectory [143]. Even though he showed this for quadcopters, it could probably extend to fixed-wing aircraft too depending on the velocity and turning radius. If there are no constraints on the angles, then aerobatic fixed-wing aircraft could probably perform the roll maneuver.

Concluding Remarks. There are some potential reasons why the static thrust model produces more motor spin rate values closer to the acceptable range. First, not having sufficient flight test data to find adequate values for η_L in the Burgers thrust model or values for the constants, k_1 and k_2, in the Staples thrust model causes inaccuracies. Second, Staples shows that his model tends to underestimate dynamic thrust by as much as 15-30% [104]. Third, there are currently no citations for the Burgers thrust model exist, which is a general model such that it applies to propellers and motors. However, this model may not be valid for quadcopters and has the largest divergence and errors among all the thrust models. Fourth, the acoustic motor spin rate model is within physical limits. Future work may include further investigation into dynamic thrust models for quadcopters. Finally, simulations of the integrated targeting and re-targeting procedures requires a set of studies outside the scope of the proposed research but are considered potential future research topics.

CHAPTER 9
RECOMMENDATIONS AND NOVELTIES

9.1 Author's Recommendations

For researchers and engineers in GNC studies, the author would like to recommend more studies towards developing UAV models. Determining accurate UAV models as the static thrust and dynamic thrust models considered here have some assumptions, which leads to inaccuracies. Another recommendation is to consider more UAV guidance and targeting problems because they are not as prevalent as UAV control problems. Generally, control problems dominate research studies, followed by navigation problems, and then guidance and targeting problems are the least popular. The author recommends DJI platforms with OSDK capabilities for research with non-aerobatic applications. The DJI OSDK functions and documentation are user-friendly and avoid issues of possibly affecting the underlying default flight controller code, making debugging and troubleshooting complicated. Another recommendation is to consider more problems implementing TGNC in racing drones to compare who has better piloting skills: the pilot or the computer (similar to the human vs. the computer in chess). TGNC developments in racing drones may open new research fields and studies for autonomous applications in racing drones or larger aircraft and spacecraft.

9.2 Proposed Novelties

1. Modified original E Guidance and derived attitude E guidance
2. Extended E Guidance methods to three and four polynomials
3. Utilized the switching function for max-intermediate thrust arcs for takeoff maneuver
4. Integrated translational E Guidance with extremal control using the Hamiltonian formalism and the indirect (adjoint) method to generate extremal trajectories for the takeoff and waypoint guidance maneuvers
5. Integrated rotational E Guidance with extremal control using the Hamiltonian formalism and the indirect (adjoint) method to generate extremal trajectory for the roll maneuver
6. Considered acoustic parameters to estimate motor spin rates
7. Developed a HME framework with EKFs for UAVs to provide estimations for states and motor angular velocities, and this is the first time that HME has been used for UAVs
8. Simulated an exponential braking guidance law for UAVs through HITL

In Cherry's original derivations, he chose $p_1(t) = 1, p_2(t) = \tau = T - t$ [117]. The author chose different polynomials: $p_1(t) = \tau, p_2(t) = \tau^2$ for the takeoff maneuver and $p_1(t) = \tau^2, p_2(t) = \tau^3$

for the waypoint maneuver. Cherry originally applied E Guidance to only translational acceleration, so extending E Guidance to angular acceleration leads is novel and practical for rotational maneuvers like the 360° roll maneuver.

Combining E Guidance with extremal control satisfies the boundary conditions to solve the optimal control problem to yield extremal control laws for the takeoff, waypoint, and roll maneuvers. Using only the Hamiltonian formalism with the indirect method to solve the optimal control problem would not satisfy the boundary conditions, as mentioned in "Velocity and Altitude Profile Solutions of the Intermediate Thrust Arc" of subsection 6.4.1. Ultimately, the guided trajectory from E Guidance is extremal for the takeoff, waypoint, and roll maneuvers.

Incorporating acoustics and several EKFs for state estimation through HME has not been attempted for UAVs. The standard approach uses one EKF with multiple sensors but does not consider acoustics. The addition of acoustics provides a new approach and perspective for state estimation methods in UAVs. Even without acoustics, multiple EKFs in an HME framework would capture more parameters and factors than a single EKF, which leads to versatile state estimation by considering several different models and parameters.

CHAPTER 10
CONCLUSION

10.1 Main Conclusions

First, utilizing E Guidance allows one to solve the two-point boundary problem and satisfy the boundary conditions without having a specific dynamical model. Second, extending E guidance to three or four polynomials may solve the boundary conditions but is impractical due to moving away from the target point. A potentially useful application involves avoiding an incoming obstacle with a desire to move forward to reach a target point behind the incoming obstacle. Third, utilizing the exponential braking guidance law allows one to achieve the target point and satisfy the boundary conditions without having a dynamical model. Fourth, first-order necessary conditions demonstrate extremality for the takeoff, waypoint, and roll maneuvers. Fifth, incorporating acoustics for estimation provides a different approach for navigation solutions but has several underlying complexities, which make it challenging to model accurately. Sixth, the HME-EKF framework has near real-time capabilities, so implementing HME-EKF onboard flight vehicles is very likely today or in the near future. Seventh, the acoustic motor spin rate estimations tend to provide underestimations, while the extremal control laws with the static thrust model tend to provide overestimations. The Burgers and Staples dynamic thrust models in Ref. [102, 104] provided extremely large and inaccurate motor spin rate estimations due to assumptions in modeling, which demonstrates how complicated it is to generate accurate UAV dynamic thrust models. Eighth, comparing battery voltages between the PID and extremal controllers provides pilots practical insight in preferring one controller over another if they desire to maximize battery life to have longer flight times or higher position accuracy. Finally, the integrated GNC software demonstrates a robust and accurate integrated GNC system for the waypoint maneuver.

There are some core research contributions from these main conclusions. First, extending E Guidance to rotational maneuvers demonstrates attitude guidance in quadcopters through the 360° roll maneuver. Second, E Guidance extensions with higher order integration of three functions have some impractical results such as moving backward, but it can help avoid incoming obstacles if it does not need to satisfy the velocity terminal conditions. Third, translational E Guidance satisfies the boundary conditions for the optimal control problem, which leads to extremal control and extremal trajectories for the takeoff and waypoint maneuvers. Fourth, utilizing HME with EKFs and acoustic parameters shows the impact of considering several different models and parameters to have accurate state estimation. Simulation results have near real-time capabilities, so onboard implementation is feasible.

10.2 Future Studies

First, determining an accurate dynamic thrust model would be beneficial for estimation and control purposes. Inaccurate dynamical models led to enormous motor spin rate values due to assumptions in the models that do not capture all the real-world variables and parameters. Second, conducting flight tests of the exponential braking law will provide experimental results to compare against the default PID controller, and obtaining experimental results leads to publishing another paper. Third, deriving a UAV acoustic model would be novel and valuable. The author attempted to determine a UAV acoustic model following the Hawkings and Lowson's method but, unfortunately, did not have much success. Having accurate UAV acoustic models would lead to direct relationships between states and motor spin rates instead of using the curve fit approach described in Ref. [122]. Fourth, second-order conditions of optimality are not considered because the theory for second-order conditions of optimality is incomplete. Finally, conduct simulations of the integrated targeting and re-targeting procedures to demonstrate real-time re-targeting for dynamic obstacle avoidance.

APPENDIX A
EXPLICIT GUIDANCE EXTENSION TABLES

360° Roll Maneuver (rad & rad/s)							
p_1	p_2	p_3	$\phi(T)$	$\omega_x(T)$	τ [J]	$\sim \phi_{max}$	$\sim \omega_{x,max}$
$(T-t)^0$	$(T-t)^1+(T-t)^0$	$(T-t)^2+(T-t)^1+(T-t)^0$	-6.198	0.0524	0.3879	3.178	7.441
$(T-t)^1+(T-t)^0$	$(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^3+(T-t)^2+(T-t)+(T-t)^0$	-6.198	0.0524	0.4979	3.959	10.55
$(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	-6.198	0.0524	0.7874	4.661	13.94
$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^5+(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	-35.84	-29.07	1.159	5.307	5.308
$(T-t)^0$	$(T-t)^1$	$(T-t)^2$	-6.198	0.0524	0.3879	3.178	7.4407
$(T-t)$	$(T-t)^2$	$(T-t)^3$	-6.198	0.0524	0.7052	4.831	13.78
$(T-t)^2$	$(T-t)^3$	$(T-t)^4$	-6.283	0	1.411	6.473	21.83
$(T-t)^3$	$(T-t)^4$	$(T-t)^5$	-6.283	0	2.471	8.148	31.70
$(T-t)^4$	$(T-t)^5$	$(T-t)^6$	-6.283	0	3.955	9.827	43.34
$(T-t)^2$	$(T-t)^3$	$(T-t)^1$	-6.198	0.0524	0.7052	4.831	13.78
$(T-t)^2$	$(T-t)^3$	$(T-t)^0$	0	0	0.4698	10.10	10.08
$(T-t)^2$	$(T-t)^3$	$(T-t)^5$	-6.283	0	1.647	6.805	23.92
$exp((T-t)^{-1})$	$exp((T-t)^0)$	$exp((T-t)^1)$	NA	NA	NA	NA	NA
$exp((T-t)^0)$	$exp((T-t)^1)$	$exp((T-t)^2)$	34.19	261.65	0.765	34.1933	273.05
$exp((T-t)^1)$	$exp((T-t)^2)$	$exp((T-t)^3)$	NA	NA	NA	NA	NA
$cos((T-t)^0)$	$cos((T-t)^1)$	$cos((T-t)^2)$	-689.97	-262.84	0.3232	-256.31	-253.54
$cos((T-t)^1)$	$cos((T-t)^2)$	$cos((T-t)^3)$	NA	NA	NA	NA	NA
$sin((T-t)^0)$	$sin((T-t)^1)$	$sin((T-t)^2)$	-634.17	753.13	2.60	-634.17	785.14
$sin((T-t)^1)$	$sin((T-t)^2)$	$sin((T-t)^3)$	NA	NA	NA	NA	NA
$cos((T-t)^0)$	$sin((T-t)^1)$	$cos((T-t)^1)$	-20.77	-25.55	0.3811	27.052	-18.11
$cos((T-t)^1)$	$sin((T-t)^2)$	$cos((T-t)^2)$	-237.83	0	0.3974	-227.64	10.24
$cos((T-t)^2)$	$sin((T-t)^3)$	$cos((T-t)^4)$	NA	NA	NA	NA	NA
$sin((T-t)^0)$	$cos((T-t)^1)$	$sin((T-t)^2)$	349.44	341.4	0.7607	349.44	353.58
$sin((T-t)^1)$	$cos((T-t)^2)$	$sin((T-t)^3)$	NA	NA	NA	NA	NA

Table A.1: Roll Maneuver Results with Three Polynomials for E Guidance

Takeoff Maneuver to 50 m ($m, m/s, m/s^2$)							
p_1	p_2	p_3	$z(T)$	$v_z(T)$	$a_{z,max}$	z_{min}	$v_{z,min}$
$(T-t)^{-1}+(T-t)^0$	$(T-t)^0$	$(T-t)^1+(T-t)^0$	50	9.68	10.75	-12.46	-1.208
$(T-t)^0$	$(T-t)^1+(T-t)^0$	$(T-t)^2+(T-t)^1+(T-t)^0$	50	0	10.60	-25.54	-3.125
$(T-t)^1+(T-t)^0$	$(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^3+(T-t)^2+(T-t)+(T-t)^0$	50	0	11.08	-36.46	-5.363
$(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	50	0	11.97	-50.05	-8.594
$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^5+(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	50	0	13.28	-63.99	-12.91
$(T-t)^{-1}$	$(T-t)^{-2}$	$(T-t)^{-3}$	NA	NA	NA	NA	NA
$(T-t)^{-2}$	$(T-t)^{-3}$	$(T-t)^{-4}$	NA	NA	NA	NA	NA
$(T-t)^{-2}$	$(T-t)^{-1}$	$(T-t)^0$	NA	NA	NA	NA	NA
$(T-t)^{-1}$	$(T-t)^0$	$(T-t)^1$	50	9.68	10.75	-12.46	-1.208
$(T-t)^0$	$(T-t)^1$	$(T-t)^2$	50	0	10.60	-25.54	-3.125
$(T-t)^1$	$(T-t)^2$	$(T-t)^3$	50	0	11.20	-38.81	-5.806
$(T-t)^2$	$(T-t)^3$	$(T-t)^4$	50	0	12.12	-52.24	-9.127
$(T-t)^3$	$(T-t)^4$	$(T-t)^5$	50	0	13.43	-65.58	-13.37
$(T-t)^4$	$(T-t)^5$	$(T-t)^6$	50	0	15.19	-79.11	-17.80
$exp((T-t)^{-1})$	$exp((T-t)^0)$	$exp((T-t)^1)$	NA	NA	NA	NA	NA
$exp((T-t)^0)$	$exp((T-t)^1)$	$exp((T-t)^2)$	NA	NA	NA	NA	NA
$exp((T-t)^1)$	$exp((T-t)^2)$	$exp((T-t)^3)$	NA	NA	NA	NA	NA
$cos((T-t)^0)$	$cos((T-t)^1)$	$cos((T-t)^2)$	223.50	15.40	40.02	-303.80	-3.108
$cos((T-t)^1)$	$cos((T-t)^2)$	$cos((T-t)^3)$	NA	NA	NA	NA	NA
$sin((T-t)^0)$	$sin((T-t)^1)$	$sin((T-t)^2)$	163.01	15.36	30.87	-363.19	8.710
$sin((T-t)^1)$	$sin((T-t)^2)$	$sin((T-t)^3)$	NA	NA	NA	NA	NA
$cos((T-t)^0)$	$sin((T-t)^1)$	$cos((T-t)^1)$	-49.33	-3.182	346.88	-523.49	-347.87
$cos((T-t)^1)$	$sin((T-t)^2)$	$cos((T-t)^2)$	49.45	0	136.55	-6.339	-29.78
$cos((T-t)^2)$	$sin((T-t)^3)$	$cos((T-t)^4)$	NA	NA	NA	NA	NA

Table A.2: Takeoff Maneuver Results with Three Polynomials for E Guidance - part 1

Takeoff Maneuver to 50 m ($m, m/s, m/s^2$)							
p_1	p_2	p_3	$z(T)$	$v_z(T)$	$a_{z,max}$	z_{min}	$v_{z,min}$
$sin((T-t)^0)$	$cos((T-t)^1)$	$sin((T-t)^2)$	256.40	18.31	45.74	-361.11	-4.552
$sin((T-t)^1)$	$cos((T-t)^2)$	$sin((T-t)^3)$	NA	NA	NA	NA	NA
$(T-t)^0$	$(T-t)^1$	$(T-t)^2+5t^2$	50	0	10.60	-25.54	-3.125
$(T-t)^0$	$(T-t)^1$	$(T-t)^2+105t^2$	50	0	10.60	-25.54	-3.125
$1+1000sin(T-t)cos(T-t)$	$(T-t)+1000sin(T-t)cos(T-t)$	$(T-t)^2+1000sin(T-t)cos(T-t)$	125.53	2.53	25.85	-2.863	-8.728
$1+20sin(T-t)cos(T-t)$	$(T-t)+20sin(T-t)cos(T-t)$	$(T-t)^2+20(T-t)cos(T-t)$	88.29	1.28	18.32	-7.555	-5.502
$1+10sin(T-t)cos(T-t)$	$(T-t)+10sin(T-t)cos(T-t)$	$(T-t)^2+10(T-t)cos(T-t)$	75.47	0.8528	15.78	-13.55	-4.621
$1+sin(T-t)cos(T-t)$	$(T-t)+sin(T-t)cos(T-t)$	$(T-t)^2+(T-t)cos(T-t)$	53.63	0.1214	11.31	-25.53	-3.123
$1+exp(T-t)^1$	$(T-t)+exp(T-t)^2$	$(T-t)^2+exp(T-t)^2$	0	0	1180	-680.48	-83.39
$1+exp(T-t)^1$	$(T-t)+exp(T-t)^1$	$(T-t)^2+exp(T-t)^1$	902.54	28.42	13.06	-28.42	14.65
$1+exp(T-t)^0$	$T-t+exp(T-t)^0$	$(T-t)^2+exp(T-t)^0$	50	0	10.60	-25.54	-3.125
$1+exp(T-t)^0-10t^3$	$T-t+exp(T-t)^0-10t^2$	$(T-t)^2+exp(T-t)^0-10*t$	50	0	10.63	-22.88	-2.532
$1+exp(T-t)^0-100t^3$	$T-t+exp(T-t)^0-100t^2$	$(T-t)^2+exp(T-t)^0-100*t$	50	0	10.59	-23.96	-2.77
$1+exp(T-t)^0-1000t^3$	$T-t+exp(T-t)^0-1000t^2$	$(T-t)^2+exp(T-t)^0-1000*t$	50	0	10.88	-22.05	-2.596
$(T-t)^0-t^3-t^2$	$(T-t)^1-t^2-t$	$(T-t)^2$	50	0	10.63	-22.84	-2.521
$(T-t)^0-(T-t)^3-(T-t)^2-(T-t)-(T-t)^0$	$(T-t)^1-(T-t)^2-(T-t)-(T-t)^0$	$(T-t)^2-(T-t)-(T-t)^0$	50	0	11.02	-35.42	-5.146
$-(T-T)^3-(T-T)^2-(T-T)-(T-T)^0$	$-(T-t)^2-(T-t)-(T-t)^0$	$-(T-t)-(T-t)^0$	50	0	11.08	-36.46	-5.363

Table A.3: Takeoff Maneuver Results with Three Polynomials for E Guidance - part 2

Takeoff Maneuver to 50 m ($m, m/s, m/s^2$)							
p_1	p_2	p_3	$z(T)$	$v_z(T)$	$a_{z,max}$	z_{min}	$v_{z,min}$
$-(T-T)^2-(T-T)-(T-T)^0$	$-(T-t)-(T-t)^0$	$-(T-t)^0$	50	0	10.60	-25.54	-3.125
$(T-t)^0+sin(T-t)cos(T-t)+exp(T-t)$	$(T-t)^1+sin(T-t)cos(T-t)+exp(T-t)$	$(T-t)^2+sin(T-t)cos(T-t)+exp(T-t)$	902.55	28.42	13.06	-28.42	14.65
$(T-t)^0exp(T-t)$	$(T-t)^1exp(T-t)$	$(T-t)^2exp(T-t)$	0	0	276.52	-412.77	-248.32
$10(T-t)+1$	$10(T-t)^2+10(T-t)+1$	$10(T-t)^3+10(T-t)^2+10(T-t)+1$	50	0	11.18	-38.50	-5.747
$(T-t)^0$	$(T-t)^{0.5}$	$(T-t)^1$	-4500	-145.16	10.50	-4500	-147.52
$(T-t)^1$	$(T-t)^{1.5}$	$(T-t)^2$	0	0	10.94	-83.87	-4.646
$(T-t)^0exp(T-t)$	$(T-t)^1exp(T-t)$	$(T-t)^2exp(T-t)$	0	0	276.52	-412.77	248.32
$10exp((T-t)/10)$	$10exp((T-t)^2/10)$	$10exp((T-t)^3/10)$	NA	NA	NA	NA	NA
$10exp((T-t)/10)(T-t)$	$10exp((T-t)/10)(T-t)^2$	$10exp((T-t)/10)(T-t)^3$	NA	NA	NA	NA	NA
$tan((T-t)^0)$	$tan((T-t)^1)$	$tan((T-t)^2)$	NA	NA	NA	NA	NA
$tan(T-t)\cdot(T-t)^0$	$tan(T-t)\cdot(T-t)^1$	$tan(T-t)\cdot(T-t)^2$	NA	NA	NA	NA	NA

Table A.4: Takeoff Maneuver Results with Three Polynomials for E Guidance - part 3

Guided Waypoint Maneuver x Results (430, 160, 100) [m]						
p_1	p_2	p_3	$x(T)$	$v_x(T)$	x_{min}	$v_{x,min}$
$(T-t)^{-1}+(T-t)^0$	$(T-t)^0$	$(T-t)^1+(T-t)^0$	430	32.25	-107.5	-4.031
$(T-t)^0$	$(T-t)^1+(T-t)^0$	$(T-t)^2+(T-t)^1+(T-t)^0$	430	0	-220.16	-10.48
$(T-t)^1+(T-t)^0$	$(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^3+(T-t)^2+(T-t)+(T-t)^0$	430	0	-325.23	-18.70
$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	430	0	-442.75	-30.14
$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^5+(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	430	0	-560.08	-43.96
$(T-t)^{-2}$	$(T-t)^{-1}$	$(T-t)^0$	NA	NA	NA	NA
$(T-t)^{-1}$	$(T-t)^0$	$(T-t)^1$	430	32.25	-107.5	-4.031
$(T-t)^0$	$(T-t)^1$	$(T-t)^2$	430	0	-220.16	-10.48
$(T-t)^1$	$(T-t)^2$	$(T-t)^3$	430	0	-334.34	-19.38
$(T-t)^2$	$(T-t)^3$	$(T-t)^4$	430	0	-449.44	-30.75
$(T-t)^3$	$(T-t)^4$	$(T-t)^5$	430	0	-564.90	-44.51
$(T-t)^4$	$(T-t)^5$	$(T-t)^6$	430	0	-60.77	-60.77
$exp((T-t)^{-1})$	$exp((T-t)^0)$	$exp((T-t)^1)$	NA	NA	NA	NA
$exp((T-t)^0)$	$exp((T-t)^1)$	$exp((T-t)^2)$	NA	NA	NA	NA
$exp((T-t)^1)$	$exp((T-t)^2)$	$exp((T-t)^3)$	NA	NA	NA	NA
$cos((T-t)^0)$	$cos((T-t)^1)$	$cos((T-t)^2)$	1309.9	33.03	-1762.2	11.95
$cos((T-t)^1)$	$cos((T-t)^2)$	$cos((T-t)^3)$	NA	NA	NA	NA
$sin((T-t)^0)$	$sin((T-t)^1)$	$sins((T-t)^2)$	1551.9	-118.52	1551.9	-283.24
$sin((T-t)^1)$	$sin((T-t)^2)$	$sins((T-t)^3)$	NA	NA	NA	NA
$cos((T-t)^0)$	$sin((T-t)^1)$	$cos((T-t)^1)$	-418.32	-10.46	-600.31	-44.87
$cos((T-t)^1)$	$sin((T-t)^2)$	$cos((T-t)^2)$	436.	0	-25.30	-312.07
$cos((T-t)^2)$	$sin((T-t)^3)$	$cos((T-t)^4)$	NA	NA	NA	NA
$sin((T-t)^0)$	$cos((T-1)^1)$	$sin((T-t)^2)$	1345.9	34.39	-1835.2	13.10
$sin((T-t)^1)$	$cos((T-1)^2)$	$sin((T-t)^3)$	NA	NA	NA	NA

Table A.5: Guided Waypoint Maneuver x Results with Three Polynomials for E Guidance

Guided Waypoint Maneuver y Results (430, 160, 100) [m]						
p_1	p_2	p_3	$y(T)$	$v_y(T)$	y_{min}	$v_{y,min}$
$(T-t)^{-1}+(T-t)^0$	$(T-t)^0$	$(T-t)^1+(T-t)^0$	160	12	-40	-1.5
$(T-t)^0$	$(T-t)^1+(T-t)^0$	$(T-t)^2+(T-t)^1+(T-t)^0$	160	0	-81.92	-3.898
$(T-t)^1+(T-t)^0$	$(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^3+(T-t)^2+(T-t)+(T-t)^0$	160	0	-121.02	-6.959
$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	160	0	-164.74	-11.21
$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^5+(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	160	0	-208.40	-16.36
$(T-t)^{-2}$	$(T-t)^{-1}$	$(T-t)^0$	NA	NA	NA	NA
$(T-t)^{-1}$	$(T-t)^0$	$(T-t)^1$	160	12	-40	-1.5
$(T-t)^0$	$(T-t)^1$	$(T-t)^2$	160	0	-81.92	-3.898
$(T-t)^1$	$(T-t)^2$	$(T-t)^3$	160	0	-124.41	-7.211
$(T-t)^2$	$(T-t)^3$	$(T-t)^4$	160	0	-167.23	-11.44
$(T-t)^3$	$(T-t)^4$	$(T-t)^5$	160	0	-210.20	-16.56
$(T-t)^4$	$(T-t)^5$	$(T-t)^6$	160	0	-253.18	-22.61
$exp((T-t)^{-1})$	$exp((T-t)^0)$	$exp((T-t)^1)$	NA	NA	NA	NA
$exp((T-t)^0)$	$exp((T-t)^1)$	$exp((T-t)^2)$	NA	NA	NA	NA
$exp((T-t)^1)$	$exp((T-t)^2)$	$exp((T-t)^3)$	NA	NA	NA	NA
$cos((T-t)^0)$	$cos((T-t)^1)$	$cos((T-t)^2)$	487.40	12.29	-655.72	4.446
$cos((T-t)^1)$	$cos((T-t)^2)$	$cos((T-t)^3)$	NA	NA	NA	NA
$sin((T-t)^0)$	$sin((T-t)^1)$	$sins((T-t)^2)$	577.44	-44.10	577.44	-105.39
$sin((T-t)^1)$	$sin((T-t)^2)$	$sins((T-t)^3)$	NA	NA	NA	NA
$cos((T-t)^0)$	$sin((T-t)^1)$	$cos((T-t)^1)$	-155.65	-3.891	-223.37	-16.70
$cos((T-t)^1)$	$sin((T-t)^2)$	$cos((T-t)^2)$	162.34	0	-9.415	-116.12
$cos((T-t)^2)$	$sin((T-t)^3)$	$cos((T-t)^4)$	NA	NA	NA	NA
$sin((T-t)^0)$	$cos((T-1)^1)$	$sin((T-t)^2)$	500.81	12.80	-682.86	4.876
$sin((T-t)^1)$	$cos((T-1)^2)$	$sin((T-t)^3)$	NA	NA	NA	NA

Table A.6: Guided Waypoint Maneuver y Results with Three Polynomials for E Guidance

Guided Waypoint Maneuver z Results (430, 160, 100) [m]						
p_1	p_2	p_3	$z(T)$	$v_z(T)$	z_{min}	$v_{z,min}$
$(T-t)^{-1}+(T-t)^0$	$(T-t)^0$	$(T-t)^1+(T-t)^0$	50	3.75	-12.5	-0.4688
$(T-t)^0$	$(T-t)^1+(T-t)^0$	$(T-t)^2+(T-t)^1+(T-t)^0$	50	0	-25.6	-1.218
$(T-t)^1+(T-t)^0$	$(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^3+(T-t)^2+(T-t)+(T-t)^0$	50	0	-37.82	-2.174
$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	50	0	-51.48	-3.505
$(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	$(T-t)^5+(T-t)^4+(T-t)^3+(T-t)^2+(T-t)^1+(T-t)^0$	50	0	-65.125	-5.112
$(T-t)^{-2}$	$(T-t)^{-1}$	$(T-t)^0$	NA	NA	NA	NA
$(T-t)^{-1}$	$(T-t)^0$	$(T-t)^1$	50	3.75	-12.5	-0.4688
$(T-t)^0$	$(T-t)^1$	$(T-t)^2$	50	0	-25.6	-1.218
$(T-t)^1$	$(T-t)^2$	$(T-t)^3$	50	0	-38.88	-2.254
$(T-t)^2$	$(T-t)^3$	$(T-t)^4$	50	0	-52.26	-3.576
$(T-t)^3$	$(T-t)^4$	$(T-t)^5$	50	0	-65.69	-5.176
$(T-t)^4$	$(T-t)^5$	$(T-t)^6$	50	0	-79.12	-7.066
$exp((T-t)^{-1})$	$exp((T-t)^0)$	$exp((T-t)^1)$	NA	NA	NA	NA
$exp((T-t)^0)$	$exp((T-t)^1)$	$exp((T-t)^2)$	NA	NA	NA	NA
$exp((T-t)^1)$	$exp((T-t)^2)$	$exp((T-t)^3)$	NA	NA	NA	NA
$cos((T-t)^0)$	$cos((T-t)^1)$	$cos((T-t)^2)$	152.31	3.840	-204.91	1.390
$cos((T-t)^1)$	$cos((T-t)^2)$	$cos((T-t)^3)$	NA	NA	NA	NA
$sin((T-t)^0)$	$sin((T-t)^1)$	$sins((T-t)^2)$	180.45	-13.78	180.45	-32.93
$sin((T-t)^1)$	$sin((T-t)^2)$	$sins((T-t)^3)$	NA	NA	NA	NA
$cos((T-t)^0)$	$sin((T-t)^1)$	$cos((T-t)^1)$	-1.216	-1.216	-69.80	-5.218
$cos((T-t)^1)$	$sin((T-t)^2)$	$cos((T-t)^2)$	50.73	0	-2.942	-36.287
$cos((T-t)^2)$	$sin((T-t)^3)$	$cos((T-t)^4)$	NA	NA	NA	NA
$sin((T-t)^0)$	$cos((T-1)^1)$	$sin((T-t)^2)$	156.50	-213.39	-213.39	1.524
$sin((T-t)^1)$	$cos((T-1)^2)$	$sin((T-t)^3)$	NA	NA	NA	NA

Table A.7: Guided Waypoint Maneuver z Results with Three Polynomials for E Guidance

APPENDIX B
VELOCITY PARTIALS

B.1 Burgers Thrust Model: Velocity Partials for Error Covariance Propagation

$$
\begin{aligned}
&\frac{\partial \dot{v}_{B,N}}{\partial v_N} = -(AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)+v_{\infty,y}(2q_0q_3+2q_1q_2)\\
&-v_{\infty,z}(2q_0q_2-2q_1q_3))(q_0^2+q_1^2-q_2^2-q_3^2)^2-(S_b\eta_L\rho_0 exp(-9.611\cdot 10^{-5}U)\\
&(4(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3))\\
&(q_0^2+q_1^2-q_2^2-q_3^2)-(d^2(16(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)+v_{\infty,y}(2q_0q_3+2q_1q_2)\\
&-v_{\infty,z}(2q_0q_2-2q_1q_3))(q_0^2+q_1^2-q_2^2-q_3^2)+16(2q_0q_2+2q_1q_3)(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)\\
&-v_{\infty,y}(2q_0q_1-2q_2q_3)+v_{\infty,x}(2q_0q_2+2q_1q_3))-16(2q_0q_3-2q_1q_2)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)\\
&-v_{\infty,x}(2q_0q_3-2q_1q_2)+v_{\infty,z}(2q_0q_1+2q_2q_3))-(18(2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_2-q_1q_3)\\
&(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3))(q_0^2+q_1^2-q_2^2-q_3^2)\\
&-2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_1+q_2q_3)(2q_0q_3-2q_1q_2)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)\\
&-v_{\infty,x}(2q_0q_3-2q_1q_2)+v_{\infty,z}(2q_0q_1+2q_2q_3))))/(S_bd^2\eta_L\rho_0 exp(-9.611\cdot 10^{-5}U)\\
&(q_0^2-q_1^2-q_2^2+q_3^2))+(6(2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_2-q_1q_3)(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)\\
&+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3))(q_0^2+q_1^2-q_2^2-q_3^2)\\
&+2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_1+q_2q_3)(2q_0q_3-2q_1q_2)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)\\
&-v_{\infty,x}(2q_0q_3-2q_1q_2)+v_{\infty,z}(2q_0q_1+2q_2q_3))))/(S_bd^2\eta_L\rho_0 exp(-9.611\cdot 10^{-5}U)\\
&(q_0^2-q_1^2-q_2^2+q_3^2))-(4AC_d(2q_0q_2+2q_1q_3)(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)-v_{\infty,y}(2q_0q_1-2q_2q_3)\\
&+v_{\infty,x}(2q_0q_2+2q_1q_3)))/(S_b\eta_L)))/24+4(2q_0q_2+2q_1q_3)(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)\\
&-v_{\infty,y}(2q_0q_1-2q_2q_3)+v_{\infty,x}(2q_0q_2+2q_1q_3))-4(2q_0q_3-2q_1q_2)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)\\
&-v_{\infty,x}(2q_0q_3-2q_1q_2)+v_{\infty,z}(2q_0q_1+2q_2q_3)))-AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(2q_0q_2+2q_1q_3)\\
&(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)-v_{\infty,y}(2q_0q_1-2q_2q_3)+v_{\infty,x}(2q_0q_2+2q_1q_3)))(2q_0q_2+2q_1q_3)\\
&+2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_3-q_1q_2)(2q_0q_3-2q_1q_2)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)\\
&-v_{\infty,x}(2q_0q_3-2q_1q_2)+v_{\infty,z}(2q_0q_1+2q_2q_3)))
\end{aligned}
\tag{B.1}
$$

$$
\begin{aligned}
&\frac{\partial \dot{v}_{B,E}}{\partial v_E} = -((S_b\eta_L\rho_0 exp(-9.611\cdot 10^{-5}U)(4(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)-v_{\infty,x}(2q_0q_3-2q_1q_2)\\
&+v_{\infty,z}(2q_0q_1+2q_2q_3))(q_0^2-q_1^2+q_2^2-q_3^2)-(d^2(16(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)\\
&-v_{\infty,x}(2q_0q_3-2q_1q_2)+v_{\infty,z}(2q_0q_1+2q_2q_3))(q_0^2-q_1^2+q_2^2-q_3^2)-16(2q_0q_1-2q_2q_3)\\
&(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)-v_{\infty,y}(2q_0q_1-2q_2q_3)+v_{\infty,x}(2q_0q_2+2q_1q_3))+16(2q_0q_3+2q_1q_2)\\
&(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3))\\
&-(6(2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_1+q_2q_3)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)\\
&-v_{\infty,x}(2q_0q_3-2q_1q_2)+v_{\infty,z}(2q_0q_1+2q_2q_3))(q_0^2-q_1^2+q_2^2-q_3^2)\\
&-2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_2-q_1q_3)(2q_0q_3+2q_1q_2)(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)\\
&+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3))))/(S_bd^2\eta_L\rho_0 exp(-9.611\cdot 10^{-5}U)\\
&(q_0^2-q_1^2-q_2^2+q_3^2))-(18(2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_1+q_2q_3)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)\\
&-v_{\infty,x}(2q_0q_3-2q_1q_2)+v_{\infty,z}(2q_0q_1+2q_2q_3))(q_0^2-q_1^2+q_2^2-q_3^2)\\
&+2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_2-q_1q_3)(2q_0q_3+2q_1q_2)(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)\\
&+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3))))/(S_bd^2\eta_L\rho_0 exp(-9.611\cdot 10^{-5}U)\\
&(q_0^2-q_1^2-q_2^2+q_3^2))+(4AC_d(2q_0q_1-2q_2q_3)(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)-v_{\infty,y}(2q_0q_1-2q_2q_3)\\
&+v_{\infty,x}(2q_0q_2+2q_1q_3)))/(S_b\eta_L)))/24-4(2q_0q_1-2q_2q_3)(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)\\
&-v_{\infty,y}(2q_0q_1-2q_2q_3)+v_{\infty,x}(2q_0q_2+2q_1q_3))+4(2q_0q_3+2q_1q_2)(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)\\
&+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3)))+AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(2q_0q_1-2q_2q_3)\\
&(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)-v_{\infty,y}(2q_0q_1-2q_2q_3)+v_{\infty,x}(2q_0q_2+2q_1q_3)))(2q_0q_1-2q_2q_3)\\
&+AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)-v_{\infty,x}(2q_0q_3-2q_1q_2)\\
&+v_{\infty,z}(2q_0q_1+2q_2q_3))(q_0^2-q_1^2+q_2^2-q_3^2)^2-2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_3+q_1q_2)\\
&(2q_0q_3+2q_1q_2)(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3)))/m
\end{aligned}
\tag{B.2}
$$

$$
\begin{aligned}
\frac{\partial \dot{v}_{B,U}}{\partial v_U} = &-(2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_1+q_2q_3)(2q_0q_1+2q_2q_3)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)\\
&-v_{\infty,x}(2q_0q_3-2q_1q_2)+v_{\infty,z}(2q_0q_1+2q_2q_3))-(S_b\eta_L\rho_0 exp(-9.611\cdot 10^{-5}U)\\
&((d^2(16(2q_0q_2-2q_1q_3)(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)+v_{\infty,y}(2q_0q_3+2q_1q_2)\\
&-v_{\infty,z}(2q_0q_2-2q_1q_3))-16(2q_0q_1+2q_2q_3)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)-v_{\infty,x}(2q_0q_3-2q_1q_2)\\
&+v_{\infty,z}(2q_0q_1+2q_2q_3))-16(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)-v_{\infty,y}(2q_0q_1-2q_2q_3)\\
&+v_{\infty,x}(2q_0q_2+2q_1q_3))(q_0^2-q_1^2-q_2^2+q_3^2)+(18(2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)\\
&(q_0q_1+q_2q_3)(2q_0q_1+2q_2q_3)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)-v_{\infty,x}(2q_0q_3-2q_1q_2)\\
&+v_{\infty,z}(2q_0q_1+2q_2q_3))-2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_2-q_1q_3)(2q_0q_2-2q_1q_3)\\
&(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3))))/\\
&(S_bd^2\eta_L\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0^2-q_1^2-q_2^2+q_3^2))+(6(2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)\\
&(q_0q_1+q_2q_3)(2q_0q_1+2q_2q_3)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)-v_{\infty,x}(2q_0q_3-2q_1q_2)\\
&+v_{\infty,z}(2q_0q_1+2q_2q_3))+2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_2-q_1q_3)(2q_0q_2-2q_1q_3)\\
&(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3))))/\\
&(S_bd^2\eta_L\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0^2-q_1^2-q_2^2+q_3^2))+(4AC_d(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)\\
&-v_{\infty,y}(2q_0q_1-2q_2q_3)+v_{\infty,x}(2q_0q_2+2q_1q_3))(q_0^2-q_1^2-q_2^2+q_3^2))/(S_b\eta_L)))/24\\
&+4(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)-v_{\infty,y}(2q_0q_1-2q_2q_3)+v_{\infty,x}(2q_0q_2+2q_1q_3))(q_0^2-q_1^2-q_2^2+q_3^2)\\
&+4(2q_0q_1+2q_2q_3)(v_{\infty,y}(q_0^2-q_1^2+q_2^2-q_3^2)-v_{\infty,x}(2q_0q_3-2q_1q_2)+v_{\infty,z}(2q_0q_1+2q_2q_3))\\
&-4(2q_0q_2-2q_1q_3)(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3)))\\
&-AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(v_{\infty,z}(q_0^2-q_1^2-q_2^2+q_3^2)-v_{\infty,y}(2q_0q_1-2q_2q_3)\\
&+v_{\infty,x}(2q_0q_2+2q_1q_3))(q_0^2-q_1^2-q_2^2+q_3^2))(q_0^2-q_1^2-q_2^2+q_3^2)\\
&+2AC_d\rho_0 exp(-9.611\cdot 10^{-5}U)(q_0q_2-q_1q_3)(2q_0q_2-2q_1q_3)(v_{\infty,x}(q_0^2+q_1^2-q_2^2-q_3^2)\\
&+v_{\infty,y}(2q_0q_3+2q_1q_2)-v_{\infty,z}(2q_0q_2-2q_1q_3)))/m)
\end{aligned}
\tag{B.3}
$$

B.2 Staples Thrust Model: Velocity Partials for Error Covariance Propagation

$$
\begin{aligned}
&\frac{\partial \dot{v}_{S,N}}{\partial v_N} = -((17734023044399104 S_b \theta_p^2((93215778045774245396482(2AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)\\
&(q_0q_2 - q_1q_3)(v_{\infty,x}(q_0^2 + q_1^2 - q_2^2 - q_3^2) + v_{\infty,y}(2q_0q_3 + 2q_1q_2) - v_{\infty,z}(2q_0q_2 - 2q_1q_3))\\
&(q_0^2 + q_1^2 - q_2^2 - q_3^2) + 2AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(q_0q_1 + q_2q_3)(2q_0q_3 - 2q_1q_2)\\
&(v_{\infty,y}(q_0^2 - q_1^2 + q_2^2 - q_3^2) - v_{\infty,x}(2q_0q_3 - 2q_1q_2) + v_{\infty,z}(2q_0q_1 + 2q_2q_3))))/\\
&(38083524501102255 S_b \theta_p^2((1517d)/(5000\theta_p))^{1.5}(4q_0^2 - 4q_1^2 - 4q_2^2 + 4q_3^2))\\
&-(233039445114435613491 2AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(2q_0q_2 + 2q_1q_3)\\
&(v_{\infty,z}(q_0^2 - q_1^2 - q_2^2 + q_3^2) - v_{\infty,y}(2q_0q_1 - 2q_2q_3) + v_{\infty,x}(2q_0q_2 + 2q_1q_3)))/\\
&(38083524501102255 S_b \theta_p^2((1517d)/(5000\theta_p))^{1.5}))((1517d)/(5000\theta_p))^{1.5})/\\
&108517448005469335937 5 + AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(2q_0q_2 + 2q_1q_3)(v_{\infty,z}(q_0^2 - q_1^2 - q_2^2 + q_3^2)\\
&-v_{\infty,y}(2q_0q_1 - 2q_2q_3) + v_{\infty,x}(2q_0q_2 + 2q_1q_3)))(2q_0q_2 + 2q_1q_3)\\
&-AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(v_{\infty,x}(q_0^2 + q_1^2 - q_2^2 - q_3^2) + v_{\infty,y}(2q_0q_3 + 2q_1q_2)\\
&-v_{\infty,z}(2q_0q_2 - 2q_1q_3))(q_0^2 + q_1^2 - q_2^2 - q_3^2)^2 - 2AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(q_0q_3 - q_1q_2)\\
&(2q_0q_3 - 2q_1q_2)(v_{\infty,y}(q_0^2 - q_1^2 + q_2^2 - q_3^2) - v_{\infty,x}(2q_0q_3 - 2q_1q_2) + v_{\infty,z}(2q_0q_1 + 2q_2q_3)))
\end{aligned}
\tag{B.4}
$$

$$
\begin{aligned}
&\frac{\partial \ddot{v}_{S,E}}{\partial v_E} = -((17734023044399104 S_b \theta_p^2((9321577804577424539648(2AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U) \\
&(q_0q_1 + q_2q_3)(v_{\infty,y}(q_0^2 - q_1^2 + q_2^2 - q_3^2) - v_{\infty,x}(2q_0q_3 - 2q_1q_2) + v_{\infty,z}(2q_0q_1 + 2q_2q_3)) \\
&(q_0^2 - q_1^2 + q_2^2 - q_3^2) - 2AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(q_0q_2 - q_1q_3)(2q_0q_3 + 2q_1q_2) \\
&(v_{\infty,x}(q_0^2 + q_1^2 - q_2^2 - q_3^2) + v_{\infty,y}(2q_0q_3 + 2q_1q_2) - v_{\infty,z}(2q_0q_2 - 2q_1q_3))))/ \\
&(38083524501102255 S_b \theta_p^2((1517d)/(5000\theta_p))^{1.5}(4q_0^2 - 4q_1^2 - 4q_2^2 + 4q_3^2)) \\
&- (2330394451144356134912AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(2q_0q_1 - 2q_2q_3) \\
&(v_{\infty,z}(q_0^2 - q_1^2 - q_2^2 + q_3^2) - v_{\infty,y}(2q_0q_1 - 2q_2q_3) + v_{\infty,x}(2q_0q_2 + 2q_1q_3)))/ \\
&(38083524501102255 S_b \theta_p^2((1517d)/(5000\theta_p))^{1.5}))((1517d)/(5000\theta_p))^{1.5})/ \\
&1085174480054693359375 + AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(2q_0q_1 - 2q_2q_3)(v_{\infty,z}(q_0^2 - q_1^2 - q_2^2 + q_3^2) \\
&- v_{\infty,y}(2q_0q_1 - 2q_2q_3) + v_{\infty,x}(2q_0q_2 + 2q_1q_3)))(2q_0q_1 - 2q_2q_3) - AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U) \\
&(v_{\infty,y}(q_0^2 - q_1^2 + q_2^2 - q_3^2) - v_{\infty,x}(2q_0q_3 - 2q_1q_2) + v_{\infty,z}(2q_0q_1 + 2q_2q_3))(q_0^2 - q_1^2 + q_2^2 - q_3^2)^2 \\
&- 2AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(q_0q_3 + q_1q_2)(2q_0q_3 + 2q_1q_2)(v_{\infty,x}(q_0^2 + q_1^2 - q_2^2 - q_3^2) \\
&+ v_{\infty,y}(2q_0q_3 + 2q_1q_2) - v_{\infty,z}(2q_0q_2 - 2q_1q_3)))
\end{aligned}
\tag{B.5}
$$

$$
\begin{aligned}
&\frac{\partial \ddot{v}_{S,U}}{\partial v_U} = ((17734023044399104 S_b \theta_p^2((9321577804577424539648(2AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U) \\
&(q_0q_1 + q_2q_3)(2q_0q_1 + 2q_2q_3)(v_{\infty,y}(q_0^2 - q_1^2 + q_2^2 - q_3^2) - v_{\infty,x}(2q_0q_3 - 2q_1q_2) \\
&+ v_{\infty,z}(2q_0q_1 + 2q_2q_3)) + 2AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(q_0q_2 - q_1q_3)(2q_0q_2 - 2q_1q_3) \\
&(v_{\infty,x}(q_0^2 + q_1^2 - q_2^2 - q_3^2) + v_{\infty,y}(2q_0q_3 + 2q_1q_2) - v_{\infty,z}(2q_0q_2 - 2q_1q_3))))/ \\
&(38083524501102255 S_b \theta_p^2((1517d)/(5000\theta_p))^{1/5}(4q_0^2 - 4q_1^2 - 4q_2^2 + 4q_3^2)) \\
&+ (2330394451144356134912AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(v_{\infty,z}(q_0^2 - q_1^2 - q_2^2 + q_3^2) \\
&- v_{\infty,y}(2q_0q_1 - 2q_2q_3) + v_{\infty,x}(2q_0q_2 + 2q_1q_3))(q_0^2 - q_1^2 - q_2^2 + q_3^2))/ \\
&(38083524501102255 S_b \theta_p^2((1517d)/(5000\theta_p))^{1/5}))((1517d)/(5000\theta_p))^{1/5})/ \\
&1085174480054693359375 - AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(v_{\infty,z}(q_0^2 - q_1^2 - q_2^2 + q_3^2) \\
&- v_{\infty,y}(2q_0q_1 - 2q_2q_3) + v_{\infty,x}(2q_0q_2 + 2q_1q_3))(q_0^2 - q_1^2 - q_2^2 + q_3^2))(q_0^2 - q_1^2 - q_2^2 + q_3^2) \\
&- 2AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(q_0q_1 + q_2q_3)(2q_0q_1 + 2q_2q_3)(v_{\infty,y}(q_0^2 - q_1^2 + q_2^2 - q_3^2) \\
&- v_{\infty,x}(2q_0q_3 - 2q_1q_2) + v_{\infty,z}(2q_0q_1 + 2q_2q_3)) + 2AC_d\rho_0 exp(-9.611 \cdot 10^{-5}U)(q_0q_2 - q_1q_3) \\
&(2q_0q_2 - 2q_1q_3)(v_{\infty,x}(q_0^2 + q_1^2 - q_2^2 - q_3^2) + v_{\infty,y}(2q_0q_3 + 2q_1q_2) - v_{\infty,z}(2q_0q_2 - 2q_1q_3)))
\end{aligned}
\tag{B.6}
$$

APPENDIX C
EXTRAMURAL RESEARCH AND COMMUNICATIONS WITH COMMUNITY

C.1 List of Publications

This section provides a list of conference and journal papers that have been accepted, published, submitted, and/or are under review.

Table C.1: List of Conference Papers and Conference Presentations

Ref.	Conference	Conference Dates	Links
[131]	ICUAS	June 12-15, 2018	`https://ieeexplore.ieee.org/document/8453467`
[120]	AAS/AIAA	Jan. 13-17, 2019	`http://www.univelt.com/book=7472`
[119]	ICUAS	June 11-14, 2019	https://ieeexplore.ieee.org/document/8797766
N/A	HESTEMP	May 2017	`https://tinyurl.com/hestempmay2017`
N/A	HESTEMP	Dec 2017	`https://tinyurl.com/hestempdec2017`
N/A	HESTEMP	Dec 2017	`https://tinyurl.com/hestempmay2018`
N/A	HESTEMP	May 2019	`https://tinyurl.com/hestempmay2019`

Table C.2: List of Journal Papers

Ref	Journal	Status
[124, 123]	*Journal of Intelligent & Robotic Systems*	Published on 7/4/20
[121]	*Journal of Dynamic Systems, Measurement, and Control*	Under Review as of on 7/9/20
[122]	*International Journal of Micro Air Vehicles*	Under Review as of 10/5/20

ME 691 Seminars

- **Autonomous Energy-Optimized Path Planning for Small UAS in Urban Environments**
  The author's first ME 691 seminar was in Fall 2018 (11/15/18) and his summer 2018 internship work on path planning using the A* algorithm in urban environments.
- **Integrated Guidance, Navigation, and Control for Unmanned Aerial Vehicles**
  The author's second ME 691 seminar was in Spring 2020 (3/4/20) and discussed the HESTEMP, ACTUAS projects, and the results in Ref. [119, 120, 121].

C.2 NASA Ames Research Center Internships

The author had three summer internships at NASA Ames Research Center in 2018, 2019, and 2020. The project work in 2018 involved implementing a Path Planning System for UAVs in the (Safe Autonomous Flight Environment) (SAFE50) Project under the Unmanned Aerial Systems Traffic Management (UTM) project. The summer internship in 2019 involved more hardware and flight tests:

1. LIDAR test 6/24/19 demonstrates LIDAR test on a mobile vehicle: `https://youtu.be/bu6QUslQjNo`
2. Flight test on 7/11/19 shows tethered flight with LIDAR: `https://youtu.be/lp7lKYjFPkA`
3. DJI SITL on 7/29/19: `https://youtu.be/tBMw5i7-DNo`
4. Flight test on 8/16/19 demonstrates DJI OSDK functions and capabilities to autonomously return to a specified point: `https://youtu.be/NGpKISQY7yU`

key takeaways from the 2019 summer internship are demonstrating DJI OSDK capabilities autonomously during the flight test on 8/16/19, which became the backbone for the C++ codes for the takeoff and waypoint guidance maneuvers in Ref. [121], [124], and [123] and accepting a NASA ARC Pathways internship (civil servant) position, which provides employment opportunities after graduation. The author presented his NASA ARC summer 2020 pathways internship work at the Advanced Control and Evolvable Systems (ACES) seminar on 8/13/20. The project involves using the HME framework for state estimation with acoustic parameters, which evolved to the proposed HME structure in Ref. [122].

C.3 Online Material

The DJI M100 and its OSDK functions provide a method for implementing extremal control and explicit guidance onboard in real-time. The link to the video is: `https://youtu.be/0zox9P-rV0I` Links to Ref. [123, 124] and the video are on the ACTUAS website: `http://manoa.hawaii.edu/actuas/actuas-work-f2019-s2020/`

APPENDIX D
EDUCATIONAL AND EXTRACURRICULAR ACTIVITIES

D.1 Classes

D.1.1 ACTUAS Classes

The author coordinated ACTUAS classes for the members to learn about assembly, image processing, and kinematics and dynamics of UAVs. Weekly assembly classes by Seth Kirchbaum involved assembling a FliteTest 270 Chase Quad and a 650 Tarot quadcopter frame. James Thesken taught image processing classes with topics such as edge detection, lane detection, detecting fruits, significance of converting RGB to HSV, and using Mask R-CNN to detect coffee beans. The module classes by Dr. Dilmurat Azimov involved airplane flight mechanics, PID control, kinematics, and dynamics. The author subbed in for Dr. Dilmurat Azimov for the module classes when he was unavailable.

D.1.2 Substitute Teaching

The author subbed in for Dr. Dilmurat Azimov for ME 451: Feedback-Control Systems in Spring 2020. ME 451 topics included dynamic models, feedback control elements and definitions, PID control, time domain specifications, transfer functions, PID tuning, direction cosine matrix, quaternions, and Euler angles.

The author subbed in for Dr. Dilmurat Azimov for ME 696: Guidance, Navigation and Control in Fall 2018. ME 696 topics included motion planning, Bug I algorithm, Bug II algorithm, quadcopter attitude stabilization simulation using PID control, and how to change the PID coefficients in Betaflight and Mission Planner.

D.2 University of Guam Workshop

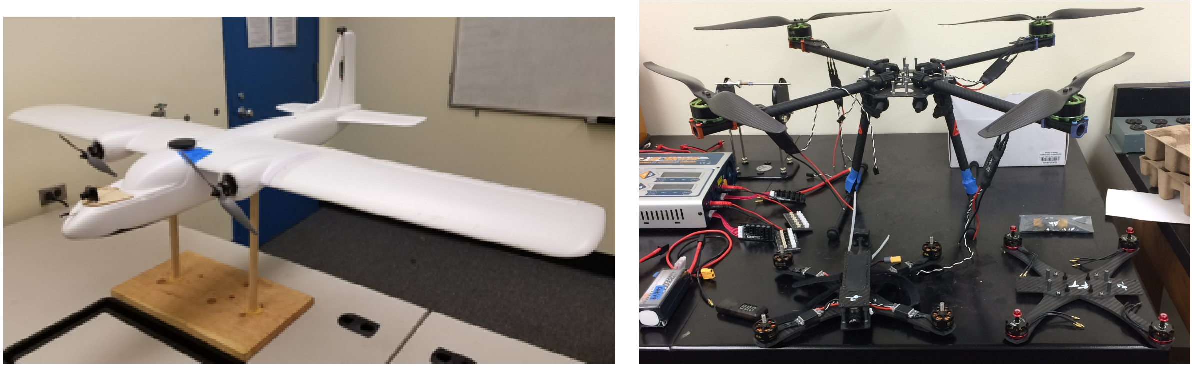
The author provided presentations to the participants from May 28 to May 31, 2019. Presentations included an introduction of the speaker's background, assembling Tiny Whoop and Flybrix quadcopters, flying tips, fundamentals of flight mechanics for airplanes with a paper plane activity, and fundamentals of PID control. The last day had mini quadcopter flying competitions. More details and pictures are available on the ACTUAS website: `http://manoa.hawaii.edu/actuas/actuas-outreach-university-of-guam-workshop-may-2019/`

D.3 University of Hawaiʻi Drone Technologies Mentorship

The author has been part of University of Hawaii Drone Technologies (UHDT) since Spring 2014 and has served as a mentor since Fall 2016.

D.4 UH Manoa UAV Club Leadership and ACTUAS Mentorship

The author led the ACTUAS club activities by scheduling flight tests and classes, docuemnting work, and weekly meetings with members. The author's mentorship activities in ACTUAS impacted his research within HESTEMP and EPSCoR projects and developed his leadership and mentorship skills. Projects included UAS 3D Mapping and Parameter Calculations, Effect of Physical Parameters (Mass) on Control of Quadcopters, Understanding the Fundamentals of Unmanned Aerial Vehicles (UAV) using PID Controls, Quadcopter PID Control and Simulations with Surveying Applications, Fixed-wing Drone Simulations and Applications for Surveying, Effect of Physical Parameters (Mass and Length) on Control of Quadcopters, PID Tuning Methods, Aircraft Noise Reduction Studies, ACTUAS - FWD & MC, Quadcopter Stabilization on Mars, and Simulating Drone Stabilization on the Moon.



(a) MyTwinDream Fixed-wing Drone

(b) 650 Tarot and FliteTest 270 Chase Quad

Figure D.1: ACTUAS Fixed-wing and Multicopter Platforms

BIBLIOGRAPHY

[1] D. Azimov, "Autonomous control technology for unmanned aerial systems with agricultural and environmental applications in central pacific islands," NASA EPSCoR. Research Performance Progress Report. July 11, 2019. 15 p.

[2] W. Zhang, *Computational Ecology: Artificial Neural Networks and Their Applications*. World Scientific, 2010.

[3] S. E. Yuksel, J. N. Wilson, and P. D. Gader, "Twenty years of mixture of experts," *IEEE transactions on neural networks and learning systems*, vol. 23, no. 8, pp. 1177–1193, 2012.

[4] NASA, "Nasa strategic plan 2018," 2018. [Online]. Available: https://www.nasa.gov/sites/default/files/atoms/files/nasa_2018_strategic_plan.pdf

[5] K. Kim, P. Pant, and E. Yamashita, "Integrating travel demand modeling and flood hazard risk analysis for evacuation and sheltering," *International Journal of Disaster Risk Reduction*, vol. 31, pp. 1177–1186, 2018.

[6] A. Nagaoka, "Hawaii study: Impacts of sea level rise already being felt — and it will only get worse," Hawaii News Now, 2018. [Online]. Available: https://www.hawaiinewsnow.com/2018/09/20/hawaii-study-impacts-sea-level-rise-already-being-felt-it-will-only-get-worse/

[7] R. W. Beard and T. W. McLain, *Small unmanned aircraft: Theory and practice*. Princeton university press, 2012.

[8] C. Hanson, J. Richardson, and A. Girard, "Path planning of a dubins vehicle for sequential target observation with ranged sensors," in *Proceedings of the 2011 American Control Conference*. IEEE, 2011, pp. 1698–1703.

[9] H. H. Hurt, *Aerodynamics for naval aviators*. Office of the Chief of Naval Operations, Aviation Training Division, 1965.

[10] 3DRobotics, "Pixhawk autopilot," 2014.

[11] "Fortini f4 osd 32khz flight controller," Furious FPV, 2017. [Online]. Available: https://furiousfpv.com/product_info.php?cPath=25&products_id=740

[12] S. Park, J. Deyst, and J. How, "A new nonlinear guidance logic for trajectory tracking," in *AIAA guidance, navigation, and control conference and exhibit*, 2004, p. 4900.

[13] B. Jones, "Plane: L1 control for straight and curved path following," 2013. [Online]. Available: https://github.com/ArduPilot/ardupilot/pull/101

[14] Ardupilot, "Navigation tuning," 2019. [Online]. Available: http://ardupilot.org/plane/docs/navigation-tuning.html

[15] D. Agar and R. Bapst, "Px4/ecl," 2018. [Online]. Available: https://github.com/PX4/ecl/blob/master/l1/ecl_l1_pos_controller.h

[16] S. Lee, J. Lee, S. Lee, H. Choi, Y. Kim, S. Kim, and J. Suk, "Sliding mode guidance and control for uav carrier landing," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 55, no. 2, pp. 951–966, 2018.

[17] D. Azimov and J. S. Allen, "Analytical model and control solutions for unmanned aerial vehicle maneuvers in a vertical plane," *Journal of Intelligent & Robotic Systems*, vol. 91, no. 3-4, pp. 725–733, 2018.

[18] S. P. Viswanathan, A. K. Sanyal, and M. Izadi, "Integrated guidance and nonlinear feedback control of underactuated unmanned aerial vehicles in se (3)," in *AIAA Guidance, Navigation, and Control Conference*, 2017, pp. 1044–1056.

[19] P. Lu, "Entry guidance: a unified method," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 3, pp. 713–728, 2014.

[20] Ardupilot, "Extended kalman filter (ekf)," 2019. [Online]. Available: http://ardupilot.org/copter/docs/common-apm-navigation-extended-kalman-filter-overview.html

[21] P. Riseborough and K. Mohammed, "Px4/ecl/ekf," 2018. [Online]. Available: https://github.com/PX4/ecl/tree/master/EKF

[22] PX4, "Using the ecl ekf," 2018. [Online]. Available: https://dev.px4.io/en/tutorials/tuning_the_ecl_ekf.html

[23] J. Novuh, "Betaflight software filters setup guide," 2018. [Online]. Available: https://www.propwashed.com/betaflight-software-filters-setup-guide/

[24] Montis, "Butterflight – a better flight firmware?" 2018. [Online]. Available: http://www.multirotorguide.com/news/butterflight-a-better-flight-controller-firmware/

[25] Quadmeup, "Butterflight kalman filter calculator," 2018. [Online]. Available: https://quadmeup.com/butterflight_kalman_calculator/

[26] S. Kolpuke, S. Kashmiri, and M. Ewing, "Bio-inspired navigation algorithm for gps denial modes," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 196–205.

[27] J. L. Mendoza-Soto and H. R. Cortés, "Generalized predictive control for trajectory tracking of a quadcopter vehicle," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 206–212.

[28] A. R. Vetrella, F. Causa, A. Renga, G. Fasano, D. Accardo, and M. Grassi, "Flight demonstration of multi-uav cdgps and vision-based sensing for high accuracy attitude estimation," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 237–246.

[29] A. Gatter and F. Andert, "Finding self-cast shadows in aerial camera images," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 213–220.

[30] E. Alpaydin, *Introduction to Machine Learning. [Sl].* The MIT Press, 2010.

[31] W. S. Chaer, R. H. Bishop, and J. Ghosh, "A mixture-of-experts framework for adaptive kalman filtering," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 27, no. 3, pp. 452–464, 1997.

[32] W. S. Chaer, R. H. Bishop, and J. Ghosh, "Hierarchical adaptive kalman filtering for interplanetary orbit determination," *IEEE transactions on aerospace and electronic systems*, vol. 34, no. 3, pp. 883–896, 1998.

[33] R. A. Jacobs and M. I. Jordan, "Learning piecewise control strategies in a modular neural network architecture," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 2, pp. 337–345, 1993.

[34] A. V. Rao, D. Miller, K. Rose, and A. Gersho, "Mixture of experts regression modeling by deterministic annealing," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2811–2820, 1997.

[35] H. S. Boroujeni and N. M. Charkari, "Robust moving shadow detection with hierarchical mixture of mlp experts," *Signal, Image and Video Processing*, vol. 8, no. 7, pp. 1291–1305, 2014.

[36] M. I. Jordan and R. A. Jacobs, "Hierarchical mixtures of experts and the em algorithm," *Neural computation*, vol. 6, no. 2, pp. 181–214, 1994.

[37] M. K. Titsias and A. Likas, "Mixture of experts classification using a hierarchical mixture model," *Neural Computation*, vol. 14, no. 9, pp. 2221–2244, 2002.

[38] J. W. Ng and M. P. Deisenroth, "Hierarchical mixture-of-experts model for large-scale gaussian process regression," *arXiv preprint arXiv:1412.3078*, 2014.

[39] C. M. Bishop and M. Svensén, "Bayesian hierarchical mixtures of experts," *arXiv preprint arXiv:1212.2447*, 2012.

[40] Ardupilot, “Advanced tuning,” 2019. [Online]. Available: http://ardupilot.org/copter/docs/tuning.html

[41] PX4, “Multicopter pid tuning guide,” 2018. [Online]. Available: https://docs.px4.io/en/advanced_config/pid_tuning_guide_multicopter.html

[42] S. Korokithakis, “How should i tune my copter?” 2018. [Online]. Available: https://github.com/betaflight/betaflight/wiki/PID-Tuning-Guide

[43] J. Walser, “Pid tuning,” 2017. [Online]. Available: https://github.com/cleanflight/cleanflight/blob/master/docs/PID%20tuning.md

[44] stronnag, “Pid tuning,” 2017. [Online]. Available: https://github.com/iNavFlight/inav/blob/master/docs/PID%20tuning.md

[45] N. Nguyen, *Model-Reference Adaptive Control*. Springer International Publishing, 2018.

[46] G. A. Garcia, S. Kashmiri, and D. Shukla, “Nonlinear control based on h-infinity theory for autonomous aerial vehicle,” in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 336–345.

[47] E. Ibarra and P. Castillo, “Nonlinear super twisting algorithm for uav attitude stabilization,” in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 640–645.

[48] C. de Cos, J. Acosta, and A. Ollero, “Relative-pose optimisation for robust and nonlinear control of unmanned aerial manipulators,” in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 999–1005.

[49] A. Matus-Vargas, G. Rodriguez-Gomez, J. Martinez-Carranza, and A. Muñoz-Silva, “Numerical optimization techniques for nonlinear quadrotor control,” in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 1309–1315.

[50] X. Yu, X. Zhou, and Y. Zhang, “Collision-free trajectory generation for uavs using markov decision process,” in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 56–61.

[51] R. G. Beall, “Engineering of fast and robust adaptive control for fixed-wing unmanned aircraft,” Naval Postgraduate School Monterey United States, Tech. Rep., 2017.

[52] A. Holtsov, R. Farhadi, V. Kortunov, and A. Mohammadi, “Comparison of the uav adaptive control with the robust control based on mu-synthesis,” in *2016 4th International Conference on Methods and Systems of Navigation and Motion Control (MSNMC)*. IEEE, 2016, pp. 18–21.

[53] E. Frazzoli, M. Dehleh, and E. Feron, "Trajectory tracking control design for autonomous helicopters using a backstepping algorithm," in *Proceedings of the American control conference*, vol. 6, 2000, pp. 4102–4107.

[54] S. L. Waslander, G. M. Hoffmann, J. S. Jang, and C. J. Tomlin, "Multi-agent quadrotor testbed control design: Integral sliding mode vs. reinforcement learning," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2005, pp. 3712–3717.

[55] D. Noble and S. Bhandari, "Neural network based nonlinear model reference adaptive controller for an unmanned aerial vehicle," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 94–103.

[56] D. Lunni, A. Santamaria-Navarro, R. Rossi, P. Rocco, L. Bascetta, and J. Andrade-Cetto, "Nonlinear model predictive control for aerial manipulation," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 87–93.

[57] J. M. Levin, A. Paranjape, and M. Nahon, "Agile fixed-wing uav motion planning with knife-edge maneuvers," in *2017 international conference on unmanned aircraft systems (ICUAS)*. IEEE, 2017, pp. 114–123.

[58] E. Bulka and M. Nahon, "Autonomous control of agile fixed-wing uavs performing aerobatic maneuvers," in *2017 international conference on unmanned aircraft systems (ICUAS)*. IEEE, 2017, pp. 104–113.

[59] B. Pugach, B. Beallo, D. Bement, S. McGough, N. Miller, J. Morgan, L. Rodriguez, K. Winterer, T. Sherman, S. Bhandari, *et al.*, "Nonlinear controller for a uav using echo state network," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 124–132.

[60] A. A. Munishkin, D. Milutinović, and D. W. Casbeer, "Stochastic optimal control navigation with the avoidance of unsafe configurations," in *The 2016 International Conference on Unmanned Aircraft Systems*. IEEE, 2016, pp. 211–218.

[61] M. Beul and S. Behnke, "Analytical time-optimal trajectory generation and control for multirotors," in *The 2016 International Conference on Unmanned Aircraft Systems*. IEEE, 2016, pp. 87–96.

[62] C. Blouin, E. Lanteigne, and W. Gueaieb, "Optimal control for the trajectory planning of micro airships," in *The 2017 International Conference on Unmanned Aircraft Systems*. IEEE, 2017, pp. 885–892.

[63] theArchLadder, "Developer info," 2016. [Online]. Available: https://github.com/iNavFlight/inav/wiki/Developer-info

[64] C. Massé, O. Gougeon, D.-T. Nguyen, and D. Saussié, “Modeling and control of a quadcopter flying in a wind field: A comparison between lqr and structured $\mathbb{H}_\infty$ control techniques,” in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2018, pp. 1408–1417.

[65] M. Yavari, K. Gupta, M. Mehrandezh, and A. Ramirez-Serrano, “Optimal real-time trajectory control of a pitch-hover uav with a two link manipulator,” in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2018, pp. 930–938.

[66] M. B. Vankadari, K. Das, C. Shinde, and S. Kumar, “A reinforcement learning approach for autonomous control and landing of a quadrotor,” in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2018, pp. 676–683.

[67] G. B. Haberfeld, D. Sun, and N. Hovakimyan, “Stabilization and optimal trajectory generation for a compact aerial manipulation system with a delta-type parallel robot,” in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2018, pp. 1091–1100.

[68] A. Manoharan, R. Sharma, and P. Sujit, “Nonlinear model predictive control to aid cooperative localization,” in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 26–32.

[69] R. Srivastava, R. Lima, K. Das, and A. Maity, “Least square policy iteration for ibvs based dynamic target tracking,” in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 1089–1098.

[70] W. Yuan and L. Rodrigues, “Onboard generation of optimal flight trajectory for delivery of fragile packages,” in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 1–8.

[71] Y. Kim, G. H. Kim, and J.-H. Choi, “Optimal guidance for range maximization of guided projectile: The effects of autopilot delay and fin deployment timing on the flight range,” in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 1143–1152.

[72] S. Ahmad and R. Fierro, “Real-time quadrotor navigation through planning in depth space in unstructured environments,” in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 1467–1476.

[73] R. Schacht-Rodriguez, J.-C. Ponsart, C. D. Garcia-Beltran, C. M. Astorga-Zaragoza, and D. Theilliol, “Mission planning strategy for multirotor uav based on flight endurance estimation,” in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 778–786.

[74] M. Farrell, J. Jackson, J. Nielsen, C. Bidstrup, and T. McLain, "Error-state lqr control of a multirotor uav," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 704–711.

[75] M. Singh, A. Manoharan, A. Ratnoo, and P. Sujit, "Three dimensional uav path following using sdre guidance," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 482–490.

[76] D. M. Azimov, *Analytical Solutions for Extremal Space Trajectories*. Butterworth-Heinemann, 2017.

[77] M. Chandarana, E. L. Meszaros, A. Tmjillo, and B. D. Allen, "Analysis of a gesture-based interface for uav flight path generation," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 36–45.

[78] B. M. Sathyaraj, L. C. Jain, A. Finn, and S. Drake, "Multiple uavs path planning algorithms: a comparative study," *Fuzzy Optimization and Decision Making*, vol. 7, no. 3, p. 257, 2008.

[79] A. J. Ten Harmsel, I. J. Olson, and E. M. Atkins, "Emergency flight planning for an energy-constrained multicopter," *Journal of Intelligent & Robotic Systems*, vol. 85, no. 1, pp. 145–165, 2017.

[80] I. J. Olson, A. J. Ten Harmsel, and E. M. Atkins, "Safe landing planning for an energy-constrained multicopter," in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2014, pp. 1225–1235.

[81] L. Zheng, P. Zhang, J. Tan, and M. Chen, "The uav path planning method based on lidar," in *International Conference on Intelligent Robotics and Applications*. Springer, 2019, pp. 303–314.

[82] Q. Zhou and S.-s. Gao, "3d uav path planning using global-best brain storm optimization algorithm and artificial potential field," in *International Conference on Intelligent Robotics and Applications*. Springer, 2019, pp. 765–775.

[83] M. Benyamin and G. H. Goldman, "Acoustic detection and tracking of a class i uas with a small tetrahedral microphone array," ARMY RESEARCH LAB ADELPHI MD, Tech. Rep., 2014.

[84] J. A. Feight, S. Whyte, J. D. Jacob, and R. J. Gaeta, "Acoustic characterization of a multi-rotor uas as a first step towards noise reduction," in *55th AIAA Aerospace Sciences Meeting*, 2017, p. 1174.

[85] R. Kapoor, S. Ramasamy, A. Gardi, and R. Sabatini, "A bio-inspired acoustic sensor system for uas navigation and tracking," in *2017 IEEE/AIAA 36th digital avionics systems conference (DASC)*. IEEE, 2017, pp. 1–7.

[86] N. Kloet, S. Watkins, X. Wang, S. Prudden, R. Clothier, and J. Palmer, “Drone on: a preliminary investigation of the acoustic impact of unmanned aircraft systems (uas),” in *24th international congress on sound and vibration*, 2017.

[87] U. Papa, G. Del Core, and G. Giordano, “Determination of sound power levels of a small uas during flight operations,” in *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*, vol. 253. Institute of Noise Control Engineering, 2016, pp. 692–702.

[88] M. Leonardo, A. M. Jensen, C. Coopmans, M. McKee, and Y. Chen, “A miniature wildlife tracking uav payload system using acoustic biotelemetry,” in *ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers Digital Collection, 2013.

[89] E. E. Case, A. M. Zelnio, and B. D. Rigling, “Low-cost acoustic array for small uav detection and tracking,” in *2008 IEEE National Aerospace and Electronics Conference*. IEEE, 2008, pp. 110–113.

[90] A. Finn and S. Franklin, “Acoustic sense & avoid for uav’s,” in *2011 Seventh International Conference on Intelligent Sensors, Sensor Networks and Information Processing*. IEEE, 2011, pp. 586–589.

[91] N. Kloet, S. Watkins, and R. Clothier, “Acoustic signature measurement of small multi-rotor unmanned aircraft systems,” *International Journal of Micro Air Vehicles*, vol. 9, no. 1, pp. 3–14, 2017.

[92] F. Farassat, “Linear acoustic formulas for calculation of rotating blade noise,” *AIAA journal*, vol. 19, no. 9, pp. 1122–1130, 1981.

[93] C. E. Tinney and J. Valdez, “Thrust and acoustic performance of small-scale, coaxial, corotating rotors in hover,” *AIAA Journal*, vol. 58, no. 4, pp. 1657–1667, 2020.

[94] C. E. Tinney and J. Sirohi, “Multirotor drone noise at static thrust,” *AIAA Journal*, vol. 56, no. 7, pp. 2816–2826, 2018.

[95] C. E. Tinney and J. Valdez, “Acoustic scaling for small rotors in hover,” in *Vertical Flight Society’s 75th Annual Forum & Technology Display*, 2019, pp. 1–11.

[96] D. L. Huff and B. S. Henderson, “Electric motor noise for small quadcopters: Part 1–acoustic measurements,” in *2018 AIAA/CEAS Aeroacoustics Conference*, 2018, p. 2952.

[97] Ardupilot, “Copter attitude control,” 2019. [Online]. Available: http://ardupilot.org/dev/docs/apmcopter-programming-attitude-control-2.html

[98] DJI, "Dji developer technologies," 2018. [Online]. Available: https://developer.dji.com/

[99] D. A. Vallado, *Fundamentals of Astrodynamics and Applications*. Springer Science & Business Media, 2001, vol. 12.

[100] A. Gibiansky, "Quadcopter dynamics, simulation, and control," 2012. [Online]. Available: https://github.com/gibiansky/experiments/blob/master/quadcopter/project.pdf

[101] J. L. Junkins and H. Schaub, *Analytical mechanics of space systems*. American Institute of Aeronautics and Astronautics, 2018.

[102] P. Burgers, "A thrust equation treats propellers and rotors as aerodynamic cycles and calculates their thrust without resorting to the blade element method," *International Journal of Aviation, Aeronautics, and Aerospace*, vol. 6, no. 5, pp. 1–21, 2019.

[103] J. D. Anderson, *Aircraft performance and design*. The McGraw-Hill Companies, Inc., 1999.

[104] G. Staples, "Propeller static & dynamic thrust calculation - part 2 of 2 - how did i come up with this equation?" *Electric RC Aircraft Guy, LLC*, 2015. [Online]. Available: https://www.electricrcaircraftguy.com/2014/04/propeller-static-dynamic-thrust-equation-background.html

[105] D. Muzar and E. Lanteigne, "Experimental characterization of brushless dc motors and propellers for flight application," in *The Canadian Society for Mechanical Engineering International Congress 2016*, 2016, pp. 1–11.

[106] M. Protacio, D. Myren, M. Zielinski, C. Jones, and R. Manning, "SAE Aero Aircraft Design & Build," in *Multidisciplinary Senior Design Conference*. Rochester Institute of Technology, 2016, pp. 1–35. [Online]. Available: http://edge.rit.edu/edge/P16121/public/MSD%20II/P16121%20Technical%20Paper.pdf

[107] S. Abujoub, "Development of a landing period indicator and the use of signal prediction to improve landing methodologies of autonomous unmanned aerial vehicles on maritime vessels," Ph.D. dissertation, Carleton University, 2019.

[108] P. Sharma and S. A. Arockia, "Conceptual design and non-linear analysis of triphibian drone," *Procedia computer science*, vol. 133, pp. 448–455, 2018.

[109] M. Konar, "Simultaneous determination of maximum acceleration and endurance of morphing UAV with abc algorithm-based model," *Aircraft Engineering and Aerospace Technology*, pp. 1–8, 2020.

[110] J. Cimbala and Y. Cengel, *Fluid Mechanics: Fundamentals and Applications*. Boston: McGraw-HillHigher Education, 2006.

[111] "*U.S. Standard Atmosphere, 1976*. U.S. Government Printing Office, Washington, D.C."

[112] F. J. Regan and S. M. Anandakrishnan, *Dynamics of atmospheric re-entry*. American Institute of Aeronautics and Astronautics, 1993.

[113] N. Hall, "Relative velocities," *NASA Glenn Research Center*, 2015. [Online]. Available: https://www.grc.nasa.gov/www/k-12/airplane/move.html

[114] C.-F. Lin, *Modern navigation, guidance, and control processing*. Prentice Hall Englewood Cliffs, NJ, 1991, vol. 2.

[115] W. Fehse, *Automated rendezvous and docking of spacecraft*. Cambridge university press, 2003, vol. 16.

[116] A. R. Klumpp, "Apollo lunar descent guidance," *Automatica*, vol. 10, no. 2, pp. 133–146, 1974.

[117] G. Cherry, "A general, explicit, optimizing guidance law for rocket-propelled spaceflight," in *Astrodynamics Guidance and Control Conference*, 1964, p. 638.

[118] A. Gelb, *Applied optimal estimation*. MIT press, 1974.

[119] E. Kawamura and D. Azimov, "Integrated optimal control and explicit guidance for quadcopters," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 513–522.

[120] E. Kawamura and D. Azimov, "Integrated targeting, guidance, navigation, and control for unmanned aerial vehicles," in *Volume 168 of the Advances in the Astronautical Sciences Series*. Univelt, 2019, pp. 4259–4277.

[121] E. Kawamura and D. Azimov, "Extremal control and explicit guidance for aerobatic unmanned aerial vehicles," *Journal of Dynamics, Measurement, and Control*, Under review as of 7/9/20.

[122] E. Kawamura, D. Azimov, J. S. Allen, and C. Ippolito, "Hierarchical mixture of experts utilizing acoustic parameters for unmanned aerial vehicles," *International Journal of Micro Air Vehicles*, pp. 1–37, Submitted on 10/5/20.

[123] E. Kawamura and D. Azimov, "Correction to: Integrated extremal control and explicit guidance for quadcopters," *Journal of Intelligent & Robotic Systems*, pp. 1–1, 2020.

[124] E. Kawamura and D. Azimov, "Integrated extremal control and explicit guidance for quadcopters," *Journal of Intelligent & Robotic Systems*, pp. 1–31, 2020.

[125] "Unmanned aircraft systems (UAS) frequently asked questions," Federal Aviation Administration, 2018. [Online]. Available: https://www.faa.gov/uas/faqs/

[126] D. M. Azimov, *Analytical Solutions for Extremal Space Trajectories*. Butterworth-Heinemann, 317 p, 2017.

[127] DJI, "DJI Matrice 100 user manual," 2016. [Online]. Available: https://dl.djicdn.com/downloads/m100/M100_User_Manual_EN.pdf

[128] D. F. Lawden, *Optimal trajectories for space navigation*. Butterworths, 1963, vol. 3.

[129] A. A. Milyutin and N. P. Osmolovskii, *Calculus of variations and optimal control*. American Mathematical Soc., 1998, vol. 180.

[130] V. Stepanyan, K. S. Krishnakumar, and A. Bencomo, "Identification and reconfigurable control of impaired multi-rotor drones," in *AIAA Guidance, Navigation, and Control Conference*, 2016, p. 1384.

[131] D. Azimov and E. Kawamura, "Real-time guidance, navigation and control framework for fixed-wing aircraft maneuvers in a vertical plane," in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2018, pp. 621–630.

[132] A. E. Bryson and Y.-C. Ho, "Applied optimal control, revised printing," *Hemisphere, New York*, 1975.

[133] FPVDroneReviews, "These are the fastest fpv racing drones of 2018," 2019. [Online]. Available: https://fpvdronereviews.com/guides/fastest-racing-drones/

[134] R. Haschke, E. Weitnauer, and H. Ritter, "On-line planning of time-optimal, jerk-limited trajectories," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 3248–3253.

[135] J. R. G. Martínez, J. R. Reséndiz, M. Á. M. Prado, and E. E. C. Miguel, "Assessment of jerk performance s-curve and trapezoidal velocity profiles," in *2017 XIII International Engineering Congress (CONIIN)*. IEEE, 2017, pp. 1–7.

[136] T. Yang, "Optimal control for a rocket in a three-dimensional central force field," 1969. [Online]. Available: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19700008097.pdf

[137] D. Burkhart, T. Crain, R. Bishop, D. Azimov, C. Chomel, L. Croix, and M. Heyne, "EDL GN&C Peer Review: EDL Navigation," 2004.

[138] T. Damarla, *Battlefield Acoustics*. Springer, 2015, p.7.

[139] Databot, 2020. [Online]. Available: https://databot.us.com/

[140] MicWAudio, "i437l." [Online]. Available: http://www.micwaudio.com/product.php?id=139

[141] D. Azimov and E. Kawamura, "Real-time guidance, navigation and control framework for fixed-wing aircraft maneuvers in a vertical plane," in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2018, pp. 621–630.

[142] C. Ippolito, S. Hening, S. Sankararaman, and V. Stepanyan, "A modeling, simulation and control framework for small unmanned multicopter platforms in urban environments," in *2018 AIAA Modeling and Simulation Technologies Conference*, 2018, pp. 1–24.

[143] G. Staples, "3/3 - autonomous quadcopter guidance & control (*physics-based* pure pursuit demonstration)," Youtube, 2017. [Online]. Available: https://www.youtube.com/watch?v=H5kXzpPFdII