Integration of Environment Sensing and Control Functions for Robust Rotorcraft UAV (RUAV) Guidance

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Navid Dadkhah Tehrani

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Professor Bérénice Mettler, Advisor

May, 2012

Acknowledgements

There are many people that have earned my gratitude for their contribution to my time during the PhD program at the *Department of Aerospace Engineering and Mechanics*. I would first like to express my gratitude to my advisor, Professor Bérénice Mettler, for her support and guidance during my thesis research. She provided me with the opportunity to conduct research on very exciting topics at *Interactive Guidance and Control Lab* (IGCL). I am also thankful for the time she spent reading my reports and papers and giving me constructive feedback. I particularly enjoyed her rigorous approach and unique insights to the physics of problems.

I would like to thank my committee members, Professor William Garrard, Professor Demoz Gebre-Egziabher and Professor Maria Gini, for their time and excellent feedback. I also owe gratitude to Professor Gary Balas and Professor Tryphoon Georgiou who offered me valuable suggestions in my oral exam.

I embarked on the research with the help of many individuals. I thank my current and former colleagues at IGCL. I send special thanks out to my co-workers Jon Andersh and Zhaodan Kong for the numerous discussions on research-related ideas including the brilliant ones that actually worked!. I enjoyed their company, cooperative attitude in research, and the fun times we shared in the lab discussing everything from history, politics, sports, tea, jobs to martial arts. Thanks to Venkat Korukanti with whom our lab developed the first version of the software which was the basis for further development. With Jon and Venkat we spent a lot of time debugging the system's code, conducting experiments, and saving the rotorcraft from crashing in the lab during technical or programming bugs. I also thank our former test pilots, James Rosenthal and Arion Mangio, for their help in the data collection experiments, Vivek Kumar Gupta for developing the CAD model of the Blade-CX2 helicopter, and Owen Anderson for the graphic modeling of Akerman Hall.

Thanks to all my friends, colleagues, and graduate program administrative assistants in the department who have made my experience in the PhD program most gratifying.

Without them, my graduate school years would have been tough. I will definitely miss the study sessions, assignment collaborations, and the good times we had in the TA office during my first year with Abhijit Chakraborty and Shervin Shajiee who kept us entertained with their humor. Thanks to my friend, Faraz Mirzaei, for many valuable discussions pertaining to the estimation and localization and for encouraging me to switch to Linux. I would like to acknowledge Dr. Pete Seiler for the fruitful discussions and his time answering my control theory questions whenever I reached him. A special thank you also goes out to my old friends, Shervin Shajiee and Hamid Bolandhemat, for encouraging me to peruse my PhD in the United States. My stay in Minnesota has been a smooth transition thanks to many fantastic roommates. I would run out of space if I wanted to name all of these beautiful individuals, but I will always remember them.

I am also grateful to Greg and Michael Piasecki for providing me with the internship opportunity with Dragonfly Pictures, Inc. (DPI) in Philadelphia in the summer of 2010. I met great engineers and had a fruitful experience at DPI and learned a lot about the distributed flight simulation in complex environments. Particularly, I would like to thank Dean Anderson, Basudeb Das, Orie Alpern, and Jim Hacunda for their help and support.

I dedicate this thesis to my parents, Nahid and Hassan for their constant support and unconditional love throughout this journey. I also owe gratitude to my brother, Nima, who has always endorsed my passions and interests.

Dedication

To my parents ***Nahid*** *and* ***Hassan***

Abstract

Unmanned Air Vehicles (UAVs) have started supplanting manned aircraft in a broad range of tasks. Vehicles such as miniature rotorcrafts with broad maneuvering range and small size can enter remote locations that are hard to reach using other air and ground vehicles. Developing a guidance system which enables a Rotorcraft UAV (RUAV) to perform such tasks involves combing key elements from robotics motion planning, control system design, trajectory optimization as well as dynamics modeling. The focus of this thesis is to integrate a guidance system for a small-scale rotorcraft to enable a high level of performance and situational awareness. We cover large aspects of the system integration including modeling, control system design, environment sensing as well as motion planning in the presence of uncertainty. The system integration in this thesis is performed around a Blade-CX2 miniature coaxial helicopter.

The first part of the thesis focuses on the development of the parameterized model for the Blade-CX2 helicopter with an emphasis on the coaxial rotor configuration. The model explicitly accounts for the dynamics of lower rotor and uses an implicit lumped parameter model for the upper rotor and stabilizer-bar. The parameterized model was identified using frequency domain system identification. In the second part of the thesis, we use the identified model to design a control law for the Blade-CX2 helicopter. The control augmentation for the Blade-CX2 helicopter was based on a nested attitude-velocity loop control architecture and was designed following classical loop-shaping and dynamic inversion techniques. A path following layer wrapped around the velocity control system enables the rotorcraft to follow reference trajectories specified by a sequence of waypoints and velocity vectors. Such reference paths are common in autonomous guidance systems. Finally, the third part of the thesis addresses the problem of autonomous navigation through a partially known or unknown 3D cluttered environment. The proposed multi-layer hierarchical guidance framework is based on optimal control principles and relies on the interaction of several subsystems such as environment sensing and mapping, Cost-to-Go (CTG) function update, reactive planning and Receding Horizon (RH) optimization. It is also tightly integrated with the path following controller.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Over the past few years small-scale, unmanned aerial vehicles have received considerable attention for confined and indoor autonomous flight. These vehicles can be utilized for missions including surveillance and reconnaissance where their small size allows them to enter remote locations that are hard to reach by ground vehicles. Small-scale helicopters, in particular, provide a unique platform because of their maneuverability, ability to fly slowly and to hover.

Unmanned aerial vehicles make it possible to perform critical tasks without endangering the life of human pilots. Currently, UAVs are typically operated remotely by human operators with low levels of autonomy from dedicated ground control stations. With the increasing complexity of the missions comes a need for higher levels of automation, reliability, and safety. Systems that need no or minor human input reduce operating costs and facilitates missions in unfriendly or remote environments. Small-scale autonomous vehicles are expected to operate in 3D environments that often will only be known to a limited level of detail.

Rotorcraft guidance in 3D partially known or unknown environments is a challenging problem. Unstable system dynamics, the dimensionality of search space for motion planning, environmental interaction complexity, the perceptual horizon limitation and limited on-board computation power all contribute to the difficultly of solving this problem.

This dissertation presents an effective system integration techniques for guiding rotorcraft UAVs in 3D partially known or unknown environments. Our approach combines state of the art modeling and control design techniques with optimal-control based motion planning technique to enable safe and reliable operation in

3D cluttered environments.

1.1 Motivation

In recent years there has been a steady increase in the fraction of military and civilian air operations that utilize unmanned aerial vehicles. These vehicles can be utilized for missions including surveillance and reconnaissance where their small size allows them to operate in remote locations that are hard to reach with crewed aircraft. Small-scale rotorcrafts, in particular, provide a unique platform because of their maneuverability, ability to fly slowly and to hover. Unmanned aerial vehicles make it possible to perform critical tasks without endangering the life of human pilots and access to confined spaces. Future generations of UAVs are expected to need a minimum level of human operator input and will be able to operate effectively under uncertain and complex conditions [4]. With the increasing complexity of the missions comes a need for higher levels of automation, reliability, and safety.

The critical properties that are sought in these operations are adaptability to unforeseeable events in the immediate environment, good overall performance and robustness to adverse operating conditions. This enables many missions that are impossible with current technology. Disaster areas such as those created during hurricane Katrina could be quickly searched from high altitude with small-scale rotorcraft. In addition, these vehicle could also fly down and into flooded or partially collapsed buildings. In 2005 Hurricane Katrina demolished the coastal regions of Louisiana and Mississippi. One of the first semi-autonomous operations using micro helicopter UAV system was performed by the Center for Robot Assisted Search And Rescue (CRASAR) from the University of South Florida [1]. Figure 1.1 shows the UAV that was used in the search and rescue operation. This figure also shows some of the images taken by the UAV's on-board camera system from the locations that are hard or impossible for human to enter.

Performing autonomous tasks in these scenarios imposes several concurrent technical requirements. From a physical perspective, the vehicle should be small and agile enough to be able to fly among obstacles. From the sensory perspective, it must carry some form of accurate, light-weight and low-power sensor package to provide accurate perception of the environment. Performing such missions

also requires information processing algorithms that enable accurate closed-loop command tracking, state estimation and localization as well as motion planning. Finally, the entire system must be integrated in a way that enables robustness and versatility.

(a) RUAV platform from iSENSYS.

(b) Damaged parking garage.

(c) Collapsed bridge.

(d) RUAV flying into a building.

Figure 1.1: Performing semi-autonomous rescue and search operation using an RUAV during hurricane Katrina. Pictures were taken from [1].

1.2 Background

The core underlying technologies of an autonomous Rotorcraft Unmanned Air Vehicle (RUAV) fall into six categories: state estimation, localization, environment mapping, modeling, closed-loop control and motion planning. The focus of this thesis is on autonomous guidance. The state estimation and localization are not addressed in this research. Figure 1.2 illustrate how the different components of a generic guidance system interact with one another given the states and pose of

the vehicle. In the following subsections we first review some of the small-scale rotorcraft test platforms that are being used in the autonomous RUAV research community. We then review the most recent research in the literature about each component of the guidance system.

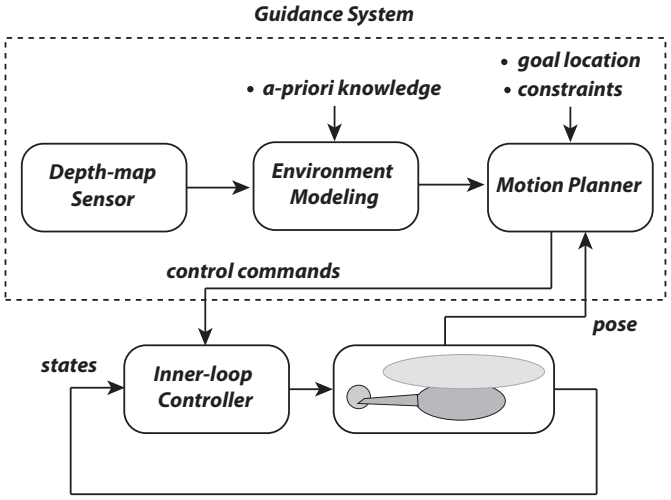


Figure 1.2: Components of a generic rotorcraft guidance system in partially known or unknown environments.

1.2.1 Miniature Rotorcraft Platforms

Research on autonomous systems has been expanded over the past few years and many research groups are using RUAVs as the test platforms (see, e.g., [5],[6]). These full-scale platforms, however, are too costly and present significant risks to humans, thereby imposing significant limitations on evaluation of high performance control laws or agile guidance systems. During recent years, highly miniaturized electric helicopters have gained popularity among research groups. These small-sized and light-weight platforms are ideal for indoor and confined operations. Miniature rotorcrafts also provide cheap platforms making them ideally suited for testing estimation, computer vision, guidance and control algorithms.

Current small scaled off-the-shelf rotorcrafts can be categorized in four different types based on the design of the rotor assembly. They are *single rotor*, *tandem rotor*, *coaxial rotor* and *quadrotor* as shown in Figure 1.3.

(a) Coaxial Blade-CX2 [7], total length = 41.7 cm.

(b) Single rotor HoneyBee [8], total length = 52.0 cm.

(c) Quadrotor AR.Drone [9], total length = 52.0 cm.

(d) Tandem T-Rex [10], total length = 101 cm.

Figure 1.3: Small-scale off-the-shelf rotorcrafts with different rotor configurations.

Single rotor setup is usually the most agile rotary wing platform due to the small time-constant of the rotor response. Figure 1.3-b shows one of the off-the shelf single rotor helicopters (HoneyBee from Esky [8]). Due to their unstable dynamics and fast response, these indoor vehicles are typically not used for guidance and control experiments. There has been significant research on state estimation and localization using indoor single rotor helicopters that can carry payload more than 100 grams [11]. Smaller single rotor helicopters have to rely on ground-based state estimation. The latency associated with communication can easily reach the bandwidth of the dynamics and therefore it is difficult to achieve good closed-loop response needed for guidance and control.

Tandem rotor vehicles have similar rotor mechanisms as single rotor, however, instead of a tail rotor another counter-rotating rotor is used. The main idea behind the original design of tandem helicopters was to be able to carry heavy payloads as

well as slung loads (see e.g. Boeing CH-47D Chinook Helicopter) due to the fact the that the dynamics are not as sensitive to Center-of- Gravity (CG) location. Figure 1.3-d shows a commercially available small-scale tandem helicopter with total length of 101 cm that has similar rotor and control mechanism as the full-scale CH-47D Chinook Helicopter. However, a tandem design of this size is not well suited for indoor experiments.

Quadrotors like the one shown in Figure 1.3-c, are very popular robotics platforms due to their inherent stability. The attitude and translational control of quadrotors are based on differential rotor rpm. Therefore, they do not have the dynamics complexity associated with tip-path-plane [12, 13] and are less challenging to control. Their dynamic response is more decoupled and do not have rotor-fuselage dynamics effect. Finally similar to tandem rotor platforms they are not sensitive to th CG location. Hence quadrotors are easier to control compared to other rotorcraft platforms. As a result, many research groups are currently using these vehicles to develop and test guidance and estimation algorithms [14, 15, 16, 17]. However, the fact that the control mechanism is based on differential rotor rpm, makes it hard or impossible to control a quadrotor when rotor inertia is high (for example a full-scale quadrotor). In fact, by using the swashplate mechanism that is used in helicopters, one can achieve a much higher bandwidth compare to differential rpm mechanism. Therefore, quadrotors have remained merely as robotics platforms rather than a scaled down of a full size vehicle.

Coaxial rotor setup is one of the most popular rotor configuration for indoor applications. Coaxial helicopters are particularly attractive thanks to the absence of fragile tail rotors and are a more compact design (see Figure 1.3-a) [2]. The concept of the coaxial helicopter was introduced many years ago aiming to eliminate some of the stability and performance issues in the single rotor designs. Kamov [18], Gyrodyne [19] and ABC Sikorsky [3] represent designs that have had some success in the helicopter industry. The basic idea of the coaxial design is to include counter-rotating rotors, which eliminates the need for an anti-torque tail rotor and makes for a more compact platform. Their small size allows operation in terrains and environments that are hard to reach by ground vehicles, making them ideally suited for missions such as surveillance and reconnaissance in confined environments or inspections of structures like bridges that otherwise

require heavy infrastructures. Regarding the small-scale platforms, the coaxial setup has the most compact design compare to the other types [2]. In addition, in terms of dynamic characteristics, coaxial miniature helicopters such as Blade-CX2 shown in Figure 1.3-a are more stable than their single rotor counterparts thanks to the upper-rotor/stabilizer-bar passive stabilizing effect. In this thesis we use coaxial miniature helicopters as our test platform. Some of the deciding factors were off-the-shelf availability, affordable price, small and compact design.

The implementations and experimental demonstrations in this thesis use the Blade-CX2 coaxial platform, however, due to the similarity of the dynamics characteristic of the miniature coaxial helicopters to the full-size helicopters, the general approach is relevant and should be valid for most rotorcraft platforms.

1.2.2 RUAV Modeling

There are two different approaches in analytical modeling of rotocrafts: *System IDentification* (SID) and *SImulation Modeling* (SIM) [20, 21]. The approach based on SIM mainly focuses on incorporating the detailed model of each component in the dynamic model of the UAV. The component models are typically based on first principle modeling of the rotor dynamics, inflow and fuselage aerodynamics, and their interactions. Wind tunnel tests and other experiments are necessary to measure or estimate the unknown parameters of each component such as, mass, moment of inertia and inflow distribution. However, careful component modeling does not guarantee that the model will be accurate in predicting the overall vehicle dynamic behavior. SIM is generally challenging especially for small-scale systems because of their distinctive design and low Reynolds number aerodynamics. Limited work on the rotor inflow and general aerodynamic characteristics is available. In addition, because of the uncertainty and simplifications in the first principle modeling, some amount of parameter tuning is necessary to get a sufficient good match between the predicted response and the actual dynamic response. Example of simulation modeling for unmanned acrobatic helicopters can be found in [22]. Rotorcraft SID modeling, on the other hand, is typically based on linearized, six degree of freedom equations of motion coupled with quasi-steady rotor aerodynamics [21]. The identification task is to determine the parameters of the state-space model of the system. The derivatives which parameterize the state-space model are complex combinations of aircraft geometry, aerodynamics

and inertia parameters. The primary applications of SID modeling is development of flight control systems, as well as, validation and refinement of simulation models. Example of system identification modeling for full-scale helicopter can be found in [21] and small-scale helicopters can be found in [23].

SID techniques are usually classified in two forms: *time-domain* methods and *frequency-domain* methods. Time-domain methods such as Maximum Likelihood (ML) [24, 20] are best suited for the identification of systems with nonlinearities. However, since the numerical integration of the system dynamics are an essential part of the time-domain system identification algorithms, biases in the inputs and measurements may introduce error in the estimates. Furthermore, careful filtering of the measured data is necessary to make sure that the collected input-output data is consistent with the model's frequency range. Failure to do this can cause local minima in the data fitting procedure. In contrast, frequency domain system identification gives direct insight into the relevant frequency range of the system via the extracted frequency responses. This knowledge can be directly applied to the determination of an appropriate model structure. Furthermore, this knowledge can help in future control system design, in particular when frequency domain control design techniques are used (see, e.g., H_∞ [25]). In addition, assuming that the measurement noise is uncorrelated with the input, the vehicle dynamics frequency content is insensitive to measurement noise as the vehicle dynamics frequency spectrum is often below the measurement noise spectrum. This eliminates the need for identifying the noise model structure which is necessary step when using time domain methods. Other advantages of frequency domain system identification that make it well suited for flight vehicles are discussed in [21]. The primary ones include (1) unbiased frequency response estimate in the presence of uncorrelated input-output noise and disturbances, (2) access to the coherence function as an unbiased measure of identification accuracy, (3) direct identification of time-delay in the measured input-output data.

Identification-modeling techniques have already been applied successfully to small-scale helicopters. Precise dynamic models, as well as, detailed understanding of the flight dynamic performance and how these are affected by key design features of the aircraft have been developed [23]. Accurate dynamic models are necessary to design high-performance control laws [25] and in the development and

evaluation of algorithms for autonomous guidance [26], flight simulation and teleoperation [27]. More recently, with growing interest in indoor applications, there has been a focus on highly miniaturized small-scale helicopters. For example in [28], a simplified model, based on first principles for three different small scaled rotorcrafts (single rotor, coaxial and quadrotor) was identified using time domain Output Error (OE) method [24]. The model, does not account for the flapping dynamics and the rotor-body interactions. The key enabling technology for indoor flight testing is vision-based motion capture. These systems preclude using onboard sensors and thus have enabled system identification of highly miniaturized rotorcraft. Other work using the same motion tracking technology has been presented in the past few years. For example in [29] the authors describe results of system identification of a miniature off-the-shelf single rotor helicopter (HoneyBee from Esky [8]). The model structure for pitch ans roll dynamics was taken from [23]. The parametric model was identified using the time-domain ML method applied. In [30], a nonlinear model for simulation of a custom-made coaxial mirco helicopter about the hover flight condition was presented. The nonlinear model is based on the 6 Degree of Freedom (DOF) vehicle dynamics. The magnitude of rotor forces and moments are modeled as a function of rotor rpm only. The model also accounts for the DC motor dynamics. In addition, a simplified model for the thrust tilt angle was used instead of the standard tip-path-plane model. The thrust tilt angle dynamics was modeled as a first order transfer function with the swash plate angle as input. For parameter identification of the model, time domain Prediction Error Method (PEM) was applied. The identification was based on the attitude measurement from a small on-board Inertial Measurement Unit (IMU) as well as height measurement from an ultrasonic sensor. The results, however, do not include comprehensive validation such as off-axis accuracy.

1.2.3 RUAV Control Design

Several control techniques have been proposed and flight tested in the literature for controlling miniature rotorcraft. In [25] H_∞ loop-shaping techniques were used to design a nested flight control system based on attitude, velocity and position loops for a small-scale helicopter. In [28], a linear quadratic (LQ) tracker design was used as a rapid control technique to control and flight test different types of VSTL miniature rotorcraft. In [31], an L_1 adaptive control approach were used

to experimentally control miniature indoor platforms. Authors in [32] used classical loop-shaping method with model-following techniques to design a control law for small-scale helicopters that is composed of an inner attitude loop with an outer velocity and position loop. The helicopter in Reference [33] was controlled using neural network based adaptive flight controller that consists of an inner attitude loop and outer trajectory control loop. The outer-loop is used to correct the commanded attitude in order to follow position and velocity commands. In [34] hierarchical flight control system based on multi-loop PID method to track the desired position was designed. Then a nonlinear model predictive controller (MPC) was applied to minimize the tracking error due to non-linearities over the flight envelope as well input/state saturation. In [35] a classical attitude controller using the identified model of the vehicle dynamics was designed. The controller was further optimized for performance using the CONDUIT [36] (Control Designer's Unified Interface) control design framework with a frequency response envelope specification. In [37] a multi-variable controller for acrobatic maneuvers was design using a combination of model-based and Linear Quadratic (LQ) design techniques. Authors in [30] designed attitude and heave/yaw controllers for a small-scale coaxial helicopter. In their work, the identified nonlinear model of the helicopter was linearized around a hover operating point and then the controller was designed using mixed-sensitivity H_∞ optimization technique. In [38] a nested sequence of attitude, velocity and position dynamic inversion loops with feedforward acceleration and velocity commands was designed for a small-scale helicopter.

In typical autonomous guidance applications these control systems are ultimately used for tracking a reference path. This problem is classified in the literature into two groups [39], [40]: *Trajectory Tracking* where the vehicle is required to track a time parametrized reference and *Path Following* where the vehicle is required to converge to follow a desired path without a temporal law while tracking a desired speed profile. Theoretical studies showed that in path following, smoother convergence to the path and less demand on control effort is achieved [41, 42]. Classical trajectory tracking approaches include designing position control loops that are able to follow a commanded position (see e.g., [32] , [38], [25], [43]). An alternative modern approach is to utilize the linear or nonlinear error dynamics around the reference trajectory and design a controller to drive the error to zero

(see e.g., [44], [45], [46], [47]). In the latter approach, however, fewer experimental results have been reported. For path following, the classical approaches aim at designing control laws to eliminate the track and cross-track error along the path. This is done by closing the path following loop around the tangent velocity vector as well as cross-track position error (see e.g., [48], [49]). Similar to the modern approaches in trajectory tracking, the alternative modern approach for path following involves construction of path-dependent error space to express the dynamics of the vehicle. The error vector comprises velocity errors as well as cross track distance error. To drive the error dynamics to zero a variety of controller synthesis techniques such as Linear Parameter Varying (LPV) can be used [50].

1.2.4 RUAV Guidance in Uncertain Environment

The problem of motion planning under uncertainty has been a topic of interest in both the *Artificial Intelligence and Robotics* community as well as the *Dynamical Systems and Controls* community. In the *Artificial Intelligence and Robotics community* this problem is often referred to as *decision-theoretic planning under uncertainty*, where decision making–which provides a way to select among multiple plans with uncertain state outcomes–is the central problem (Boutilier et al. [51] and Blythe [52]). In *Dynamical System and Control*, however, it is referred to as *control-theoretic planning under uncertainty*, where the emphasis is on the dynamical response, including taking full advantage of vehicle dynamical capabilities and accounting for state and control constraints. Here, safety and stability are usually the main issues.

Practical RUAV guidance systems are designed to handle a variety of sources of uncertainty [53]. One of the most important sources of uncertainty is uncertainty in environment knowledge. Under this uncertainty, the RUAV has incomplete or imperfect information about its environment. This could be due to inaccuracy in the a-priori map or imperfect and noisy exteroceptive sensory information that is provided to the robot in order to map the environment.

Ref. [54] proposed a real-time path planning method for UAVs in an uncertain environment. In their approach, the kinematic system of the vehicle is reduced to a set of feasible trim trajectories and maneuvers (similar to the maneuver automaton in Frazzoli [55]). In addition, the environment is partitioned based on low-risk points given by the risk map. After computing the cost of all branches between

the low risk points, the Dijkstra algorithm is used to find a best sequence of points. Finally a local, finite horizon planner is used to plan the trajectory between the important waypoints of the initial path. The local planner iteratively determines the best trim trajectory by minimizing a cost function that considers the time integral of the risk and the path length. Randomized motion planning techniques such as Probabilistic Roadmap (PRM) [56] and Rapidly-exploring Random Trees (RRT) [57] have been successful in dealing with the high-dimensional configuration space arising in many real-world applications (see e.g., [58] for an application to unmanned helicopter). However they rely on accurate models of the environment. Authors in [59] proposed an extension of PRM that computes motion plans that are robust to environment uncertainty by incorporating uncertainty in PRM sampling as well as modeling the cost of collision in traveling through uncertain regions of the map.

In the aforementioned approaches, the uncertain area of the map is globally known and the vehicle has to determine a plan to reach the goal as safely as possible. However, many urban UAV applications require the vehicle to have sufficient on-board situational awareness to avoid collision with unanticipated obstacles in the immediate environment while fulfilling the global planning requirements. This can be achieved by environment sensing, mapping and fast re-planning in real-time.

To accommodate plan updates when operating in partially known or unknown environments, many incremental graph search algorithms have been proposed [60]. Authors in [61] proposed an incremental algorithm for a generalization of the shortest path problem in a graph with an arbitrary edge insertion, edge deletion and edge-length changes. Ref. [62] proposed the D* algorithm (Dynamic A*) for optimal and efficient re-planning in partially known environments. In this approach, rather than re-calculating the optimal path for the entire map when changes in the map are detected, a reduced set of cells are checked and the optimal path to the robot's pose is updated incrementally. Focused D* [63], focuses the repairs using heuristics to reduce the total time for re-planning. Ref. [64] proposed D* Lite, which implements the same navigation strategy as Focused D* but is algorithmically different. These incremental planners, which make use of the results of the previous plans to generate a new plan, can substantially speed up the planning cycles. However, finding an optimal plan within the available time might

not be possible. Anytime algorithms (Zilberstein and Russell [65]), in contrast, try to find the best plan within the given available time, Ref. [66] proposed the Anytime D* planning algorithm, which is both anytime and incremental. However, finding an optimal plan within the amount of available time for re-planning might not be possible. Anytime algorithms [65] try to find the best plan they can within the amount of time available to them. Authors in [66] propose the Anytime D* planning algorithm which is both anytime and incremental.

Robot mapping is an active research area in Robotics. It addresses the problem of acquiring spatial models of the environment through robot's on-board sensors. Ref. [67] has presented a survey of robot mapping algorithms with a focus on indoor environments. Since both the robot pose and the map are uncertain, a vast body of literature has focused on solving the mapping problem and the induced problem of localizing the robot with respect to the map. This process is referred to as *Simultaneous Localization and Mapping* (SLAM) (see e.g., [68]). The fundamental mathematical principles used in robot mapping and localization is Bayesian filtering. The filter is used to calculate the robot pose and map posterior probability distribution, given all the control commands and measurements. A fundamental issue with this formulation is that the posterior probability distribution has infinitely many dimensions and cannot be implemented. Hence most practical localization and mapping algorithms in the literature, such as the *Kalman filter* ([69]), *Expectation Maximization* (EM) ([70]) and *particle filter* ([71]), resort to simplifying assumptions such as normal probability distribution or representing the probability distributions by samples in order to provide a working mapping algorithm.

Many localization and mapping algorithms focus on indoor and cyclic environments (such as office space) that provide many landmarks. Therefore most of these algorithms cannot be directly applied to aerial vehicles flying in 3D outdoor environments. The problem of mapping with known pose alleviates some of the practical challenges and received much attention in the literature. One of these mapping algorithms, known as occupancy grid map (originating from Ref. [72]), is an approach to model the world and robot perception by using a probabilistic representation of spatial information. This approach employs a multidimensional tessellation of space into cells, where each cell stores a probabilistic estimate of its state. The probabilistic environment cell estimates are obtained by interpreting

the spatial sensor data using the probabilistic sensor model [73]. Improvements to the original occupancy grid algorithms have been proposed recently (see e.g., [74] and [75]).

For path planning in uncertain environments different combination of mapping and planning techniques have been used in the literature to form a working UAV guidance system. For practical applications, the guidance algorithm must be chosen based on the mission scenario and characteristics of the problem. For instance, availability of a position sensor suite (such as GPS), level of disturbances in the operational environment and type of exteroceptive on-board sensors are some of the deciding factors. In Ref. [76] the 3D environment is represented using occupancy maps obtained from stereo-based measurements. For path planning of the rotorcraft UAV, a combination of D* Lite and PRM was used. Authors in [77] used the *plan globally and react locally* approach to control a UAV helicopter that combined a slower path planning layer based on Laplacian (which is a form of potential function inspired from fluid motion model) and a high-frequency reactive obstacle avoidance algorithm. In their approach, a 3D laser scanner is used to sense the environment and provide the information to populate the occupancy grid. Ref. [78] presented an approach for 3D environment perception and global planning for unmanned helicopters. In their work, a 3D occupancy grid is built incrementally. In order to have a compact representation, an approximate polygonal prism shape world model is created from the occupancy grid data and sampling-based planning methods (such as PRM and QRM (Quasi Random Roadmap)) were also used for path planning. Authors in [79] posed motion planning of a UAV helicopter in an uncertain environment as the adaptive optimal control of an uncertain Markov decision process and use the *certainty equivalence principle* [80] to compute the control policy based on the current estimate of the environment. In [81] an exploration method in unknown environments using Model Predictive Control (MPC) based obstacle avoidance for a UAV helicopter with an on-board laser scanner to build obstacle maps. In [82] a risk minimization motion planning technique is proposed and flight tested on a UAV helicopter. In their approach, the environmental sensing error as well as inner-loop tracking error are combined into a risk map. The risk map is then used to determine a navigation function to guide the helicopter. Ref. [83] proposed a multi-layer trajectory planning architecture for rotorcraft UAVs. The obstacle detection method is based on filtering the scanning

information using Interactive Multiple Model Kalman Filter (IIM-KF). The path planning is done using a dual RRT and route enhancement based on the Dijkstra algorithm. Dual RRT allows the end point to grow simultaneously with the start point and when the two trees meet, the feasible path is found. The readers are referred to [84] and [53] for more details on the existing UAV planning methods.

1.3 Challenges of RUAV Guidance in Realistic Missions

Although research on RUAV guidance has become popular over the past decade, the following challenges have limited the rate of achievements:

- Inherently unstable, fast, and multi-variable dynamics of RAVs. Therefore, in addition to being hard to control, when things do go wrong, they go wrong very quickly. Small or infrequent errors can have serious consequences, causing the vehicle to physically crash, damaging the on-board electronics, and potentially endangering the people nearby. Even if replacement parts for these inevitable crashes are budgeted, the time necessary for frequent repairs detracts from the core research.

- Small vehicles also have limited payload capabilities for sensors and computation, severely limiting the data available to the system, the amount of processing that can be done on-board, communication performance and the flight endurance. Onboard exteroceptive sensors such as laser scanners are near the limit of payload for miniature rotorcrafts, and even though cameras are becoming increasingly miniaturized, the computational requirements for vision processing are significant and difficult to satisfy given the payload limitations. Sensor data can be transmitted wirelessly to powerful off-board computers, but this introduces several more challenges with bandwidth, latency, and reliability of these wireless connections.

- Real environments are uncertain. Small-scale autonomous vehicles are expected to operate in 3D environments that often will only be characterized to a limited level of details. In addition, the on-board exteroceptive sensor carried by the vehicle has a limited perceptual horizon.

- The added third-dimension of motion and constraints for RUAVs makes environment mapping and path planning fundamentally more complex than

typical 2D systems such as ground robots. On the other hand, however, it provides additional degree of freedom for obstacle avoidance.

- Given the RUAV's large number of state/input variables, the dimensionality of the search space for motion planning leads to the curse of dimensionality making the solution of the corresponding dynamic programming problem intractable.

1.4 Problem Statement

In this section we will define the problems that will be addressed in the rest of the dissertation. The system integration approach to make an autonomous guidance system for miniature coaxial RUAV involves of the following steps:

- **Modeling and System Identification**: Many RUAVs in industry that are capable of autonomous flight use empirically tuned PID controller to achive stability and command tracking. These approaches to flight control achieve limited performance. In order to apply more advance control and guidance techniques, an accurate model of the vehicle is required. Therefore, the first problem to be solved in this thesis is to develop a first principle based parametric model of the coaxial helicopter and perform comprehensive multi-input multi-output parameter identification at full frequency spectrum of operation.

- **Controller Design and Closed-loop Performance Evaluation**: To design and evaluate experimental performance of a path following control system for coaxial rotorcraft with special emphasis on the requirements called for by autonomous guidance. The main requirements are the ability to track trajectories generated by the guidance system as well as characterizing tracking error across the maneuvering envelope. Being able to *experimentally* characterize the performance of the closed-loop system can also arise in situations where no mathematical models of the vehicle dynamics and control system are available, e.g., these models may be restricted, inaccurate, or too cumbersome to be identified. The procedure used to experimentally characterize the closed-loop system performance can be used without having to separately identify each component.

- **Motion Planning**: To autonomously guide the rotorcraft to reach the destination location safely while satisfying the performance requirements and motion constraints in 3D partially known or unknown environments. The vehicle is assumed to carry exteroceptive sensors such as a laser scanner to perceive the immediate environment.

Assumptions

The state estimation and localization of the RUAV are not addressed in this thesis. Therefore, is is assumed that the position and states of the vehicle can be measured or estimated.

1.5 Approach Overview

1.5.1 System Identification of Blade-CX2 Coaxial Helicopter

The first part of the thesis describes the identification modeling and analysis of a miniature coaxial helicopter. We focus on the development of the low-complexity parameterized model with an emphasis on the co-axial rotor configuration. The model explicitly accounts for the dynamics of lower rotor and uses an implicit lumped parameter model for the upper rotor and stabilizer-bar. The parameterized model was identified using frequency domain system identification. Frequency domain system identification guaranties that a high fidelity dynamics model over a wide frequency range of operation is achieved. The block diagram in Figure 1.4 shows the overall system identification process. Once a set of good frequency responses with high coherence are obtained from flight-test data, an iterative process is used to find the a low complexity parametric model.

The flight-data collection experiments were performed in an indoor flight test facility built around a commercial vision-based tracking system. Before using this system for flight data collection the kinematic consistency of the data was verified. In parameter identification, we seek for a set of parameters in the system state space model that produces a frequency response matrix which best fits the estimated frequency response obtained from the measured data.

It is possible in frequency domain system identification that by increasing the order of the model or adding dynamic coupling effects to the model, the frequency domain matching between the model and experiment improves. However in some

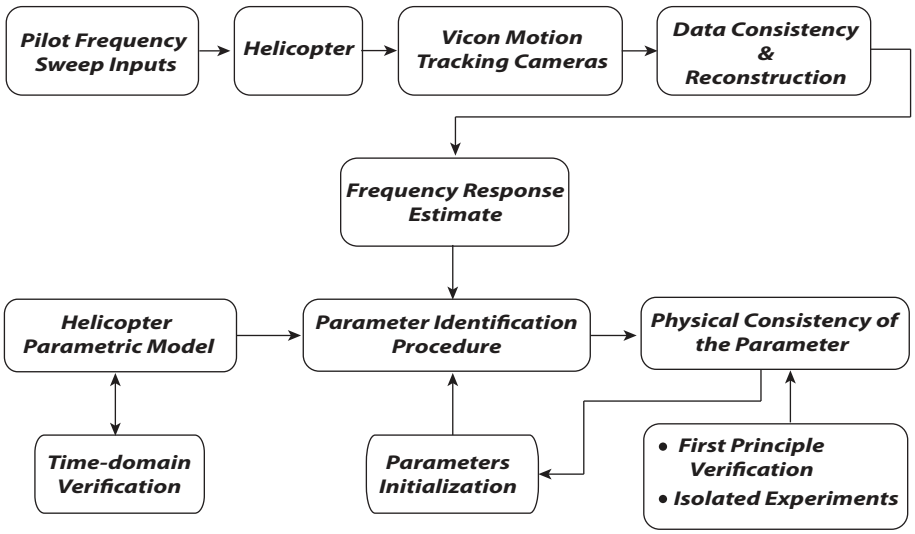


Figure 1.4: Frequency domain system identification procedure in this thesis.

cases only a marginal improvement is achieved in the time-domain. Therefore, a comprehensive time-domain validation to adjust/validate the model and to study the off-axis effects are necessary.

System identification is a mathematical process and even an accurate resulting model does not imply that the model is physically meaningful. Therefore, we provided an analysis and verification of the identified derivatives based on first principles to demonstrate that the model structure is physically meaningful. The identified parameters and model's physical consistency were examined using experiments in which specific aspects of the dynamics were isolated. For example, we used image sequences from a high frame-rate camera to verify the rotor and stabilizer bar time-constants. The helicopter's low frequency Phugoid modes were verified via piloted experiments. These modes are dominant characteristics in the translational dynamics and make the velocity control system design more challenging. We expect that the model structure will be appropriate for the identification of other small-scale coaxial helicopters.

1.5.2 Control System Design and Performance Evaluation for Rotorcraft Guidance

The concept of control performance within autonomous guidance is not clearly defined and the available literature has only focused on handling quality requirements for full-scale manned helicopters (see e.g., [85]). Additionally, in most of

the published work, the experimental or simulation results are limited to standard testing for control loops, such as steps or doublets, as well as defining feasible trajectories for evaluating path following/trajectory tracking controller performance. Therefore, most attention has been directed on precise tracking and closed-loop response and less upon characterizing the overall system performance from guidance perspective.

The second part of the thesis focuses on the design of practical control law for rotorcraft guidance and addresses specific challenges arising from 1) particular dynamics and modeling uncertainties found in miniature rotorcraft; 2) particular control requirements called for by path following; 3) determining adequate methods to experimentally characterize the performance; and 4) modeling the tracking error and path following performance, based on basic control system and dynamical systems setup. We provide experimental results and validation based on a miniature, coaxial Blade-CX2 helicopter.

The control design approach is based on a nested architecture combining an inner attitude loop and an outer velocity control loop as shown in Figure 1.5. A classical loop-shaping approach was applied to design a control law for the inner attitude loop. Blade-CX2 helicopters have a lightly damped Fuselage-Rotor-Stabilizer/bar (FRS) and Phugoid mode which makes loop-shaping a sensible design method for this system. The system identification of the Blade-CX2 helicopter shows that a reduced order model can relate the attitude and velocity dynamics. This is due to the fact that the velocity states are much slower in responding to commands and disturbances than the flapping or angular rate states. Thanks to this time-scale separation, a dynamic inversion technique was applied for the outer velocity control loop.

Within an autonomous guidance system, the control system has the role of executing trajectories that are typically generated by a planning algorithm. The reference path or trajectory is specified by a sequence of desired waypoints, and desired speeds of travel along each path segments. The control laws and guidance system are integrated through a path following architecture motivated by path following stability analysis of rigid-body helicopter dynamics.

The closed-loop behavior describes the vehicles basic dynamic response characteristics. Accurate specification of the *absolute maneuvering envelope* is necessary to avoid generating infeasible trajectories for the closed-loop system by the path

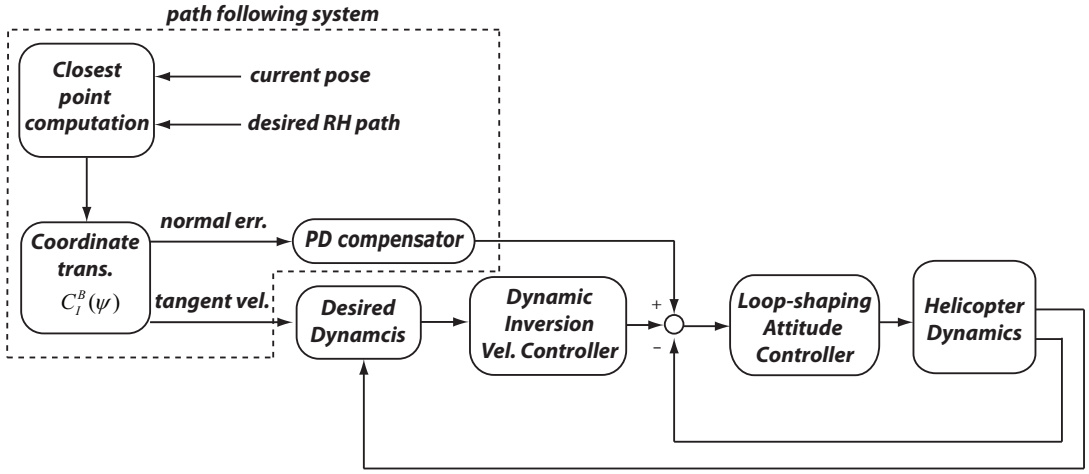


Figure 1.5: Inner-loop control system that is based on nested attitude and velocity control system as well as a path following layer to ensure the vehicle stays on the path during each planning cycle.

planner. The maneuvering envelope is typically described by the maximum normal acceleration as a function of velocity. In addition, for operation in confined environments it is important to be able to predict the tracking error-bound in order to determine the safety of a candidate trajectory to the Active Waypoint (AWP). The tracking error is also often a function of the velocity and normal acceleration. To summarize, the trajectory optimization must account for the following performance factors: 1) Absolute maneuvering envelope; 2) Nominal closed-loop model; 3) Trajectory error-bound for path following. To capture theses requirements of autonomous guidance, a novel set of experimental evaluation technique are developed for the closed-loop system.

1.5.3 Sensory Predictive Guidance for Partially Known or Unknown Environments

From a non-linear dynamical system's standpoint, there are two different approaches to trajectory generation [86, 87]: *One-degree of freedom* (1-DOF) and *two-degree of freedom* (2-DOF) as illustrated in Figure 1.6. The 2-DOF design consists of a trajectory generator and a feedback controller. The RH trajectory generator repeatedly finds a feasible open-loop trajectory by solving a finite-horizon constrained optimal control problem starting from the current state that satisfies system and actuation constraints. For execution, the appropriate commands are

sent to the inner-loop feedback controller. The inner-loop control system objective is to keep the vehicle as close as possible to the currently planned trajectory in the face measurement noise, unmodeled dynamics, uncertainties and disturbances which are not taken into account by the trajectory planner. The advantage of this approach is that the system is tracking a feasible trajectory along which the system can be stabilized. Furthermore, the reference trajectory can be as aggressive as allowed by the model.

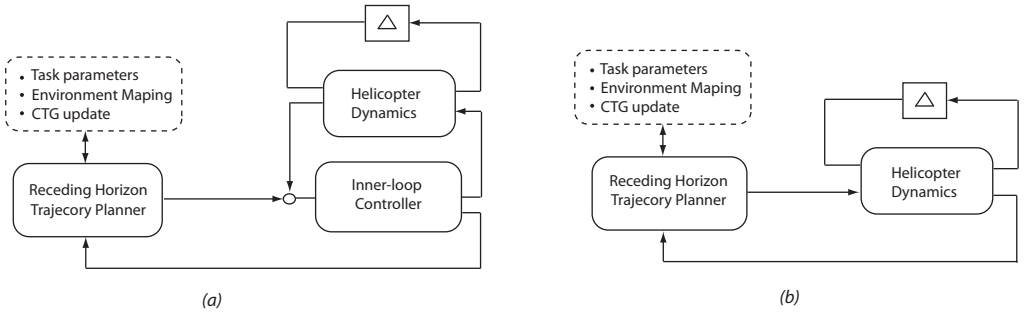


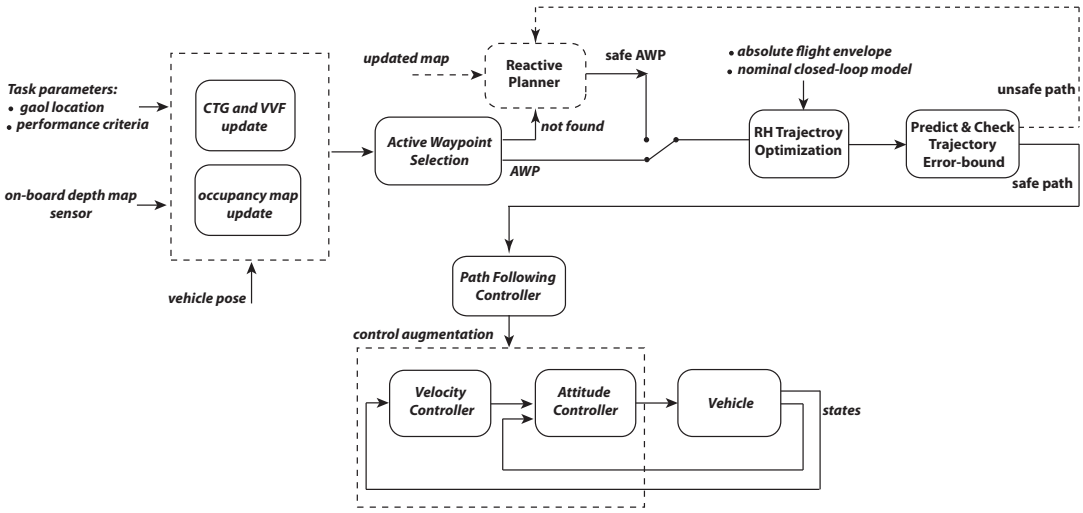
Figure 1.6: Receding horizon trajectory planning in 2-DOF (a) and 1-DOF (b) architectures.

Ideally, one would like to combine the trajectory generator with the stabilizing controller into a 1-DOF design. In this approach, trajectory generation and tracking are solved simultaneously as a single problem and the the trajectory generator directly sends the control commands to vehicle's actuator. Although some work has been done on global stability of this approach using a receding horizon implementation (see [88, 89] and references therein), the corresponding optimization problem in general is not computationally tractable when accounting for all the state/input constraints, unmodeled dynamics as well as dynamically changing environment states.

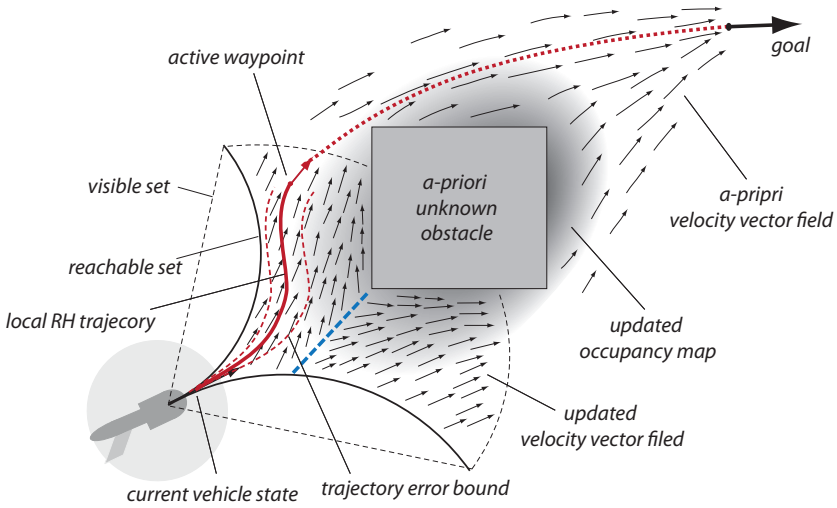
In addition, the time-scale separation between trajectory planner and controller is large enough that the two processes can be decoupled. The 2-DOF design decouples the planning from control design, which consequently limits the bandwidth of the autonomous system to that of the controller. However, the advantage of this hierarchical design is that it is computationally more tractable and provides flexibility to add modules needed to increase the functionality of the guidance system. In hierarchical design, the trajectory planner and other modules do not have to share the same information on environment and vehicle state which allows to reduce the dimensionality of the planning problem.

The 2-DOF guidance approach in this thesis is based on optimal control principles for trajectory optimization. We present an extended guidance framework for RUAVs operating in partially known or unknown environments. The approach builds on the previously flight-tested approach [90] based on RH guidance with a cost-to-go function computed using motion primitives. The proposed framework assumes that the vehicle carries an on-board depth sensor. An on-line occupancy map (see e.g., [73]) is used to recursively register and update the environment states. To accommodate for changes in the cost-to-go (CTG) function we use a dynamic version of Dijkstra's algorithm. Rather than re-calculating the entire CTG function when changes in the map are detected, a reduced set of cells limited to the vehicle's visible set are checked and their CTG values are updated in each planning cycle. By combining the CTG and occupancy probability map, we present a novel approach for guiding the vehicle to the goal in cluttered unknown environment while maintaining a trade-off between safety and performance. We also address the integration between the trajectory planning and the inner-loop control system. To ensure safe path tracking, a path tracking error model is provided that can be used in real-time to to predict the tracking error bound based on a pre-specified path generated by a RH path planner in each planning cycle. Finally, a reactive planning technique is integrated with the guidance framework to provide higher level of safety and reliability as well as a last resort solution for dealing with uncertainty and obstacle avoidance.

Figure 1.7 gives an overview of the sensory predictive guidance proposed in this paper. The multi-layer hierarchical guidance system relies on the interaction of several subsystems. Figure 1.7-b gives an illustration of the function and interaction of the key processes. Each subsystem is explained in detail throughout the thesis. The following serves as a quick overview. First, the CTG map, velocity vector field and occupancy map are updated based on depth-sensor measurements to account for changes in the immediate environment.



(a) Block diagram of sensory-predictive guidance system.



(b) Functional illustration of the guidance scheme.

Figure 1.7: Overview of the sensory-predictive guidance framework.

Then the *Active Waypoint* (AWP) (which acts as the next sub-goal) is selected based on a combination of risk (obtained from the occupancy map) and performance (obtained from CTG map) measures in the vehicle's visible set. Once an AWP is found, a time-optimal trajectory is computed in receding horizon fashion from the current vehicle state to the active waypoint, based on a model of the vehicle's absolute maneuvering envelope as well as nominal closed-loop model. Accurate specification of the absolute maneuvering envelope and nominal closed-loop is necessary to avoid generating infeasible trajectories for the closed-loop system.

For operation in confined environments it is important to be able to predict the tracking error-bound in order to determine the safety of the RH trajectory to the AWP. The tracking error is a function of speed and curvature of the path. If the RH path is labeled unsafe in this stage, the planner switches to the *Reactive Planner* module to find an AWP solely based on minimizing the risk of collision. Once a trajectory that satisfies all these requirements is generated, it is sent to the path following controller layer. The path following controller is integrated with the helicopter inner-loop controller to compensate for the track and cross-track error in order to follow the desired RH path. The inner-loop controller is based on a nested architecture combining an inner attitude loop and an outer velocity control loop.

The modules in the guidance architecture of Figure 1.7-a operate at different time-scales and they communicate in a distributed way through message passing mechanism. In the proposed guidance architecture it is possible to run the RH planner at a slower rate compared to the inned-loop controller (0.5-1 Hz versus 50 Hz) in order to reduce the computation load. The CTG and map update modules run as the depth measurement data becomes available.

1.6 Contributions

The primary contribution of this thesis is the design of a sensory predictive guidance architecture that is able to autonomously navigate a small-scale rotorcraft in unknown 3D indoor environment toward the destination location. Our approach combines key elements from robotics motion planning and control systems such as environment sensing and mapping, Cost-to-Go (CTG) function update, reactive planning, Receding Horizon (RH) optimization as well as path following and achieves tight integration between these different modules which results in a high level of performance and situational awareness. The significant contributions of this guidance system are:

- An optimal control based guidance system which extends our early work reported in [90]. The *Optimal Control Problem* (OCP) is solved approximately through hierarchical approach in multi parallel-layers (such as inner-loop control system, mapping process, CTG update, trajectory planning as well as reactive planner) to reduce dimensionality of the OCP and achieve

real-time performance.

- Through recursively updating the spatial value function and occupancy probability map from depth sensory measurements, a high level situational awareness for the immediate environment is achieved. In addition, by combing the CTG map and occupancy probability map we present a novel method of selecting subgoals in the vehicle's visible set (called the *Active Waypoint*) to achieve a trade-off between safety and performance.

- An existing reactive planner is tightly integrated with the guidance system to provide higher level of safety and collision avoidance.

- A path following layer is integrated with the guidance system which enables the rotorcraft to follow the RH trajectories that are generated in each RH planning cycle. Since the RH computations requires relatively larger computation time that other processes in the guidance system, the path following layer alleviates the need to increase the frequency of RH planing.

- Given the limited on-board computation power in RUAVs, the distributed multi-layer guidance architecture allows different processes to run at different time-scales. This allows, the computation power to be shared between processes.

The sensory predictive guidance is tightly integrated with the RUAV inner-loop control system. Therefore, developing a guidance system for an RUAV would not be practical without focusing on a specific test platform. As mentioned previously, in this research we chose Blade-CX2 coaxial miniature helicopter which has dynamic characteristics similar to a full-size RUAV as well as the compactness needed for indoor flight tests. The secondary contributions which result from effort to attain the main goal were development of a model and control system design for Blade-CX2 helicopter. The contributions obtained from this research were:

- Development of a low-complexity linear parameterized model with an emphasis on the co-axial rotor configuration. The model explicitly accounts for the dynamics of lower rotor and uses an implicit lumped parameter model for the upper rotor and stabilizer-bar.

- Provided an analysis and verification of the identified derivatives based on first principles to demonstrate that the model structure is physically meaningful. We used image sequences from a high frame-rate camera to verify the rotor and stabilizer bar time-constants. The helicopter's low frequency Phugoid mode and high frequency coupled Rotor-Fueselage-Stabilizerbar (RFS) modes were verified via piloted experiments.

- We found that in Blade-CX2 coaxial helicopter the Phugoid and RFS modes are lightly damped in comparison to other miniature helicopters (e.g. the R-50 [35]) which makes the controller design more challenging. In addition, extra phase margin (PM) created due to the rate feedback effect of the stabilizer-bar, makes the overall closed-loop sluggish.

- Existing control design techniques of *loop-shaping* and *dynamics inversion* were combined to design a novel nested velocity controller architecture for Blade-Cx2 helicopter that achieves robust stability against the parametric and structured uncertainty modeling.

- To be able to track trajectories that are generated by the guidance system, a path following layer was augmented with the velocity controller. The path following system is motivated by the nonlinear stability analysis of rigid body helicopter dynamics.

- Provided novel experimental performance evaluation techniques for the closed-loop system. The performance factors include 1) Absolute maneuvering envelope; 2) Nominal closed-loop model; 3) Trajectory error-bound for path following. All of these factors are obtained experimentally for a generic RUAV and are required for tight integration between inner-loop control system and the outer-loop guidance system.

- Developed novel methods to evaluate the overall performance of the control system in terms of the tracking error statistics. The approach is motivated by the need to characterize and measure performance requirements for autonomous guidance applications. A tracking error model, which is simple enough to be evaluated online within a guidance system, is derived and validated for our system.

Finally, experimental and simulation results in this thesis could not be achieved without a real-time software infrastructure. In that regard, we:

- Developed a high fidelity 3D simulation environment that is integrated with RIPTIDE [91] in order to demonstrate the performance of the sensory predictive guidance in cluttered a-priori unknown environments. The graphical simulation is able to mimic the functionality of most common on-board depth map sensors such as camera as well as laser scanner and can be used to test, visualize and validate RUAV planning and control algorithms.

- Developed a distributed software infrastructure for robotic research in our *Interactive Guidance and Control Lab* (IGCL). More specifically, we developed software modules for real-time data collection, control and path planning of RUAVs in the lab facility (see Chapter 2) using C++ in the ROS environment [92].

We believe that the software developed in this research are very useful resources to the future robotic research at IGCL.

1.7 Thesis Outline

The thesis is organized as follows. Chapter 2 introduces the software and hardware infrastructure of the *Interactive Guidance and Control Lab* (IGCL). Chapter 3 discusses system identification modeling of the Blade-CX2 test platform. In Chapter 4 we explain controller design and closed-loop performance evaluation for autonomous rotorcraft guidance. Chapter 5 describes autonomous guidance in a known environment using *Spatial Value Functions*. In Chapter 6 we address *Sensory Predictive Guidance* in unknown or partially known environments. Finally, in Chapter 7 we draw some conclusions, and present some ideas for future research in the area.

1.8 Publication Note

Portions of the literature survey presented in this Chapter appeared in [53]. The software and hardware infrastructure for our Interactive Guidance and Control

Lab (IGCL) in Chapter 2 was presented in [93]. The system identification modeling and flight characteristics analysis of Blade-CX2 helicopter in Chapter 3 was discussed in [94]. The Blade-CX2 control system design and experimental evaluation in Chapter 4 can be found in [95]. The receding horizon guidance scheme using spatial value function in Chapter 5 was presented in [90, 26]. Extension of this work for partially known environment in Chapter 6 can be found in [96].

Chapter 2

Hardware and Software System Infrastructure

This chapter addresses some of the hardware and software infrastructures at *Interactive Guidance and Control Lab* that were used for the simulation and experiments throughout the dissertation. We start by describing the Lab's hardware components and then present the software architecture for real-time experiments as well as the software architecture for the simulation environment. Finally, some related topics pertaining the use of the motion tracking cameras for identification and control are presented.

2.1 Hardware Components

2.1.1 Vicon Motion Tracking Cameras

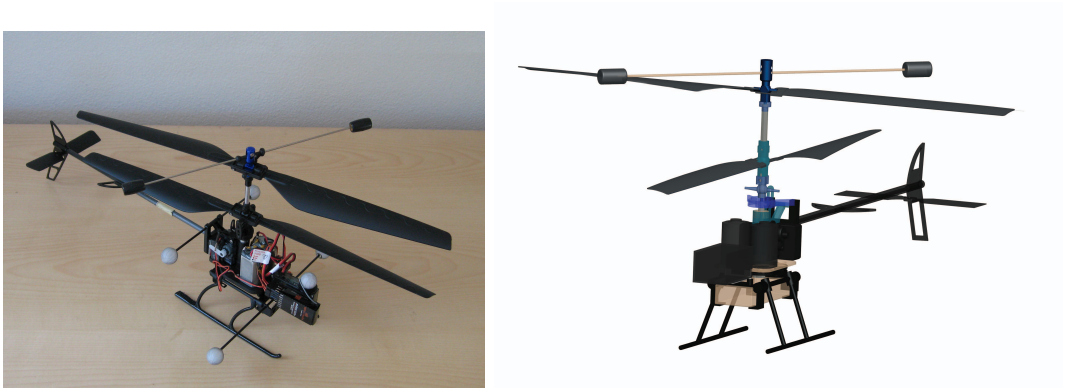
The motion tracking system consists of 6 high-speed MX-40 cameras. These cameras are affixed to the walls of the lab. For tracking an object retro-reflective markers have to be attached to the object. These markers are distributed on the object in a way that ensures that at least three markers are always visible to each camera. This marker configuration is entered in the tracking system software (Vicon Tacker) (see the reflective markers attached to the helicopter in Figure 2.2). The cameras are equipped with an array of infrared LEDs to increase the signal-to-noise ratio from the markers reflections. The cameras are equipped with their own image segmentation process needed to determine the coordinates of the markers' centroid in the image frame. An object's orientation and position is

determined by positions of the markers belonging to that object. The centroid of the bounding box corresponds to the position and orientation of the object. The cameras are connected to a router and from there to the computer via an Ethernet link.

Figure 2.1: Vicon Mx-40 Camera.

2.1.2 Miniature Helicopter

The helicopter studied in this paper is a Blade-CX2 from E-Flite [7]. This helicopter has been the primary aircraft used in our Interactive Guidance and Control Lab (IGCL). The helicopter is used to develop and validate high performance guidance, control and identification techniques. Table 2.1 shows Blade-CX2 physical specifications. Originally designed for indoor hobby flight, the helicopter coaxial rotor system features a free floating Bell style stabilizer bar, which is coupled to the upper rotor, and a swash-plate actuated lower rotor. Both rotors are two bladed and driven by individual electric motors. This helicopter has been one of the most popular commercially available miniature helicopter in this size class. In spite of its small size and low cost it possesses all the significant functions of a full-sized helicopter. In the manual mode, the helicopter is controlled by a joystick transmitter with built-in Spektrum 2.4GHz DSM technology. The lateral and longitudinal control is achieved through a conventional swash-plate mechanism which produces cyclic blade pitch variation on the lower rotor. Onboard electronics consist of a standard 4 channel receiver with integrated rate gyro for yaw damping feedback and dual speed controllers with mixing settings for heave and yaw motions.



(a) Physical System. (b) Detailed CAD model.

Figure 2.2: The Blade-CX2 coaxial helicopter.

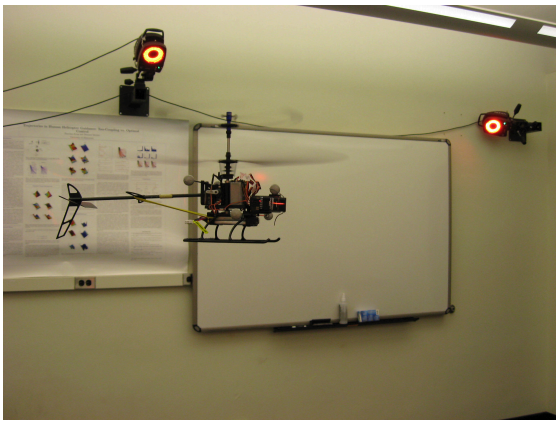
Table 2.1: The Blade-CX2 helicopter physical parameters.

Gross weight	227 grams
Payload capacity	50 grams
Nominal rotor speed	1800 rpm
Main rotor diameter	345 mm
Length	417mm
Battery	7.4V 800mah Li-Po

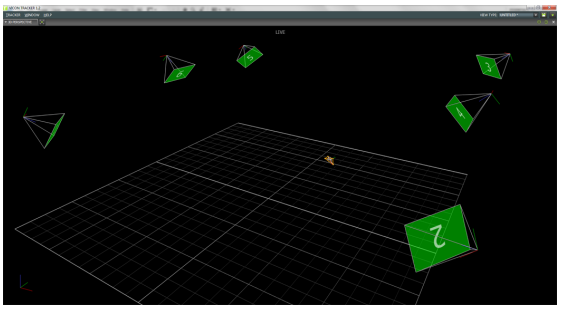
2.1.3 Vicon Processing Computer

The Vicon processing computer runs the *Vicon Tracker* software [97] that was described in Section 2.1.1. This computer is connected to a local area network (LAN) using a Gigabit network card. The Vicon Tracker software streams the tracking data with a very small latency (order of 10 msec) to the base-station computer through TCP/IP. The tracking data include position of the reflective markers as well as position and orientation of the centroid of the bounding box of the reflective markers. Our current Vicon processing desktop computer has an Intel core i7 processor, 8 GB RAM and running Windows 7 professional.

Using estimation filters we compute the angular and translational velocities and accelerations of the helicopter from the measure position and orientation. Details of the estimation filters are described in Subsection 2.3.2.



(a) Blade-CX2 Helicopter during the flight test.



(b) Screen-shot of the Vicon Tracker software interface shows the cameras tracking the helicopter.

Figure 2.3: Flight test experiment in the Interactive Guidance and Control Lab.

2.1.4 Base-station Computer

The base station computer is the main processing unit for real-time experiments in the lab. It interacts with the user, gathers the navigation data from Vicon processing computer and on-board sensors, and controls the helicopters through the trainer port of RC transmitter or Ethernet channel. Our current base-station desktop computer has a 2.4GHz Intel Quad Core 2 processor, 3.2GB of RAM, runs Ubuntu Linux 10.04 LTS and is connected to LAN. The user interacts with the base-station through the USB joystick and keyboard interfaces.

2.1.5 On-board Processing Computer

The off the shelf helicopter is modified to include on-board computer and sensors, as shown in Figure 2.4. The on-board computer is a Gumstix Verdex [2] board which runs a scaled down version of Linux. It has on-board memory and

Wi-Fi connectivity. Due to the weight limitations of these micro helicopters, the platform components are chosen to keep the weight within the limit and at the same time provide the required functionality. Figure 2.4 shows the helicopter with on-board computer and sensors. The on-board computer is connected to a LAN and communicates with the base station through Wi-Fi channel. Detailed discussion of on-board hardware platform is beyond the scope of this thesis. Interested readers are refer to [2] for more information.

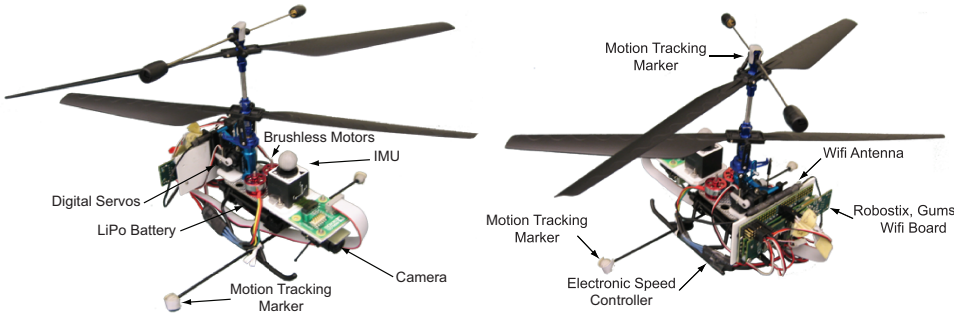


Figure 2.4: On-board platform electronics and sensors on modified off the shelf helicopter (adapted from [2]).

2.1.6 Simulation Computer

The simulation computer is the main processing unit for hight fidelity simulation environment. It is used to simulate guidance and control algorithms in realistic 3D graphical environments. The simulation computer can also communicate with the base-sation and Vicon computer to visualize the vehicle while flying in the lab or emulate virtual environments. The user interacts with the simulation computer through the USB joystick and keyboard. Our current desktop simulation computer has an Intel core i7 processor, 4 GB RAM and runs Ubuntu Linux 10.04 LTS.

2.2 Software Architecture

2.2.1 Base-station Software Architecture

The base-station software is written in C++ and integrated with the motion capture system and other modules shown in Figure 2.5 using ROS [92]. ROS is an open-source meta-operating system. It provides useful tools, code libraries as well as services such as message-passing between processes, package management

and low-level device control. ROS developed and supported by Willow Garage and widely used in the robotics community. Figure 2.6 shows a snapshot of a guidance experiment that was done using the software and hardware infrastructure.

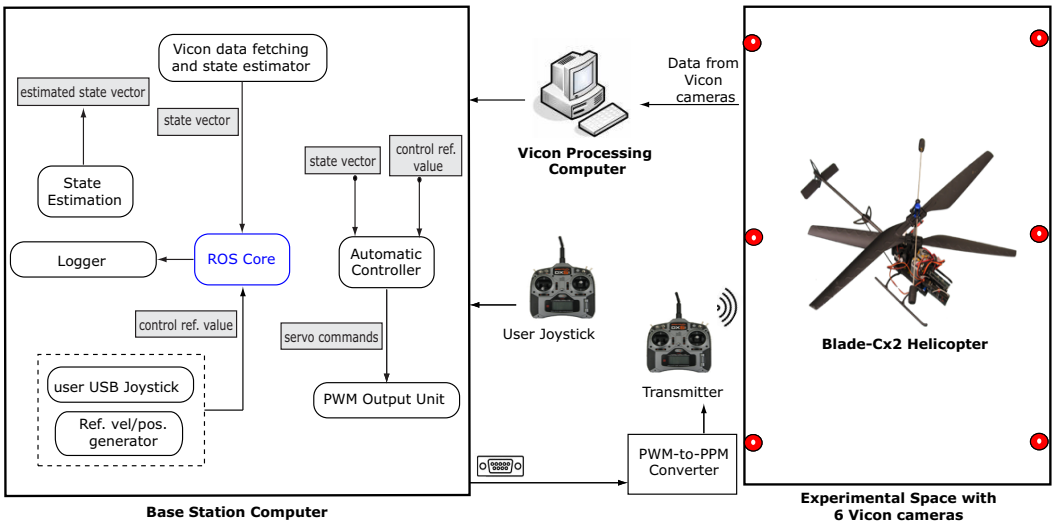


Figure 2.5: Software architecture of the lab.

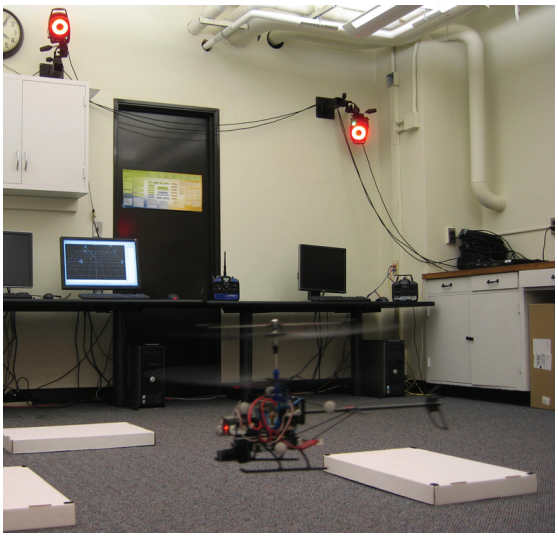


Figure 2.6: Snapshot of a guidance experiment in the lab facility.

2.2.2 Simulation Software Architecture

The 3D simulation is built around RIPTIDE (Real-time Interactive Prototype Technology Integration/Development Environment) [91] and our own custom C++

processes. RIPTIDE is a real-time simulator developed by the US Army Aeroflight-dynamics Directorate (AFDD) to provide an integrated rapid prototyping environment for simulating Unmanned Aerial Vehicles (UAVs) in complex environment. It has been used in a number of projects to simulate and test UAV guidance system [82, 83]. In RIPTIDE, the vehicle and control system is modeled in SIMULINK Real-Time Workshop (RTW) which allows rapid modification and evaluation of the different closed-loop control systems.

RIPTIDE schedules four primary real-time processes: *vehicle model*, *control system*, *pilot inceptor* and *graphic rendering process*. Inter-process communication among these processes in RIPTIDE is facilitated using a set of shared memory structures created by the real-time executive. All the state, control as well as environmental variables are updated in real-time in the shared memory structures. Therefore, one main advantage of using RIPTIDE is that any costume process can access the shared memory variables and can write to them.

Our custom C++ processes use this feature to be able to communicate with RIPTIDE. Figure 2.7 shows the functional block diagram of the whole simulation environment. Our costume processes include *laser scanner simulator*, *mapper* as well as *path planner*. The inter-processes communication between theses processes is done using IPC [98]. IPC provides high-level support for connecting processes using TCP/IP sockets and sending data between processes. It takes care of opening sockets, registering messages, and sending and receiving messages.

In Figure 2.7 the process called *RIPTIDE IPC Interface* is the program that takes care of communication between RIPTIDE and the costume processes. It updates the RIPTIDE shared variables to the IPC server and also writes the control commands (or any other variable) to the RIPTIDE shared variables. The last part of the simulator setup is the 3D graphic model of the environment as well as the helicopter. We created the detailed full-scale graphic model of our *Interactive Guidance and Control Lab* as well as the Akerman Hall of the University of Minnesota in the OpenFlight format. The OpenFlight format is specifically for creating 3D models for visual simulation and gaming environments and is optimized for real-time rendering. Figure 2.8 shows snap shots of the simulation environment.

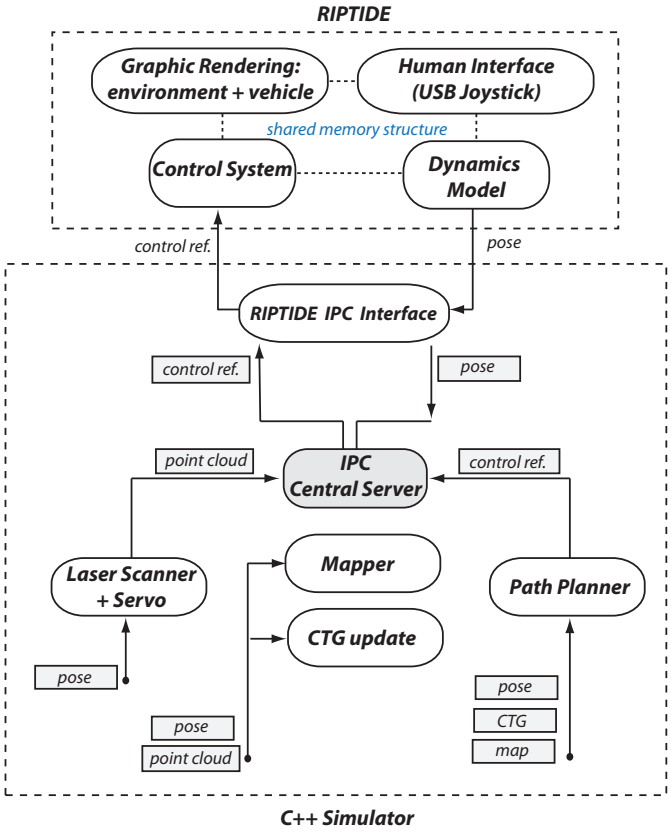
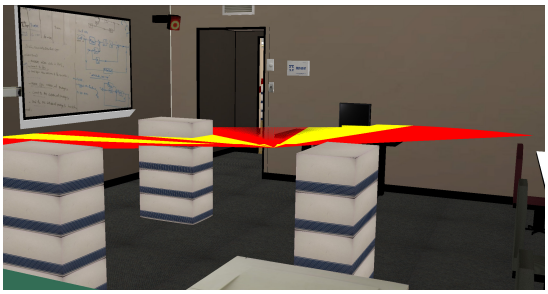


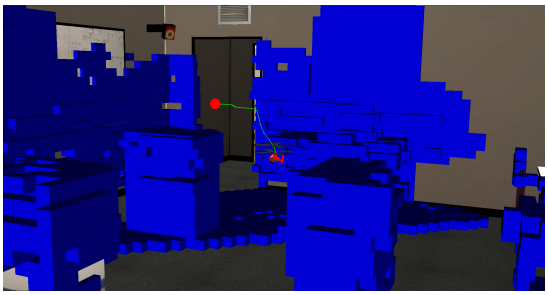
Figure 2.7: Functional block diagram of the multi-process simulation environment

(a) helicopter tracking an active waypoint



(b) laser scanning the environment

(c) on-board view



(d) mapping process

Figure 2.8: Snapshots of the simulation environment.

2.3 Related Topics

2.3.1 Time Delay Analysis of the Setup

The Block diagram of Figure 2.9 illustrates how the time-delay enters the setup: First, the helicopter states are measure by Vicon motion tracking cameras and sent to the control computer through TCP/IP. The computer generated control signal is then converted to Pulse Position Modulation (PPM) signal using a costume made electronic board. The signal is then sent to the RC transmitter. Once transmitted, the helicopter on-board receiver receives the control commands and send them to the servo actuators. This process involves greater time delay that is seen by the controller compare to the manual RC mode.

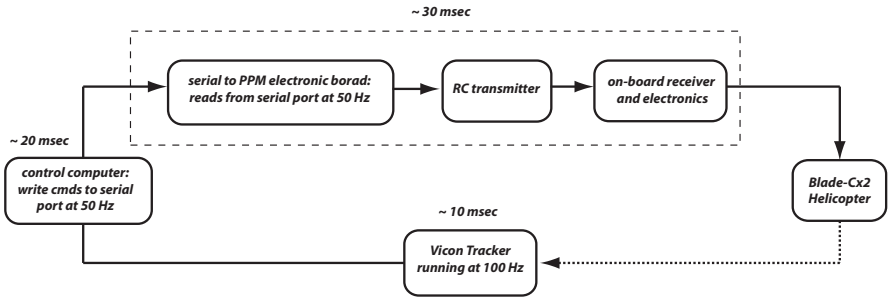


Figure 2.9: The diagram shows how the delay enters the system from different modules.

Figure 2.10 shows the servo actuator identification process that can be also used to determine the overal time-delay in the setup: Reflective markers are attached to the servo actuator and a command signal is sent to the servo via manual joystick. By measuring the response of the servo from Vicon cameras and comparing it to the commanded signal one can estimate the amount of time-delay. Figure 2.9 shows the approximate time-delay contribution for each component.

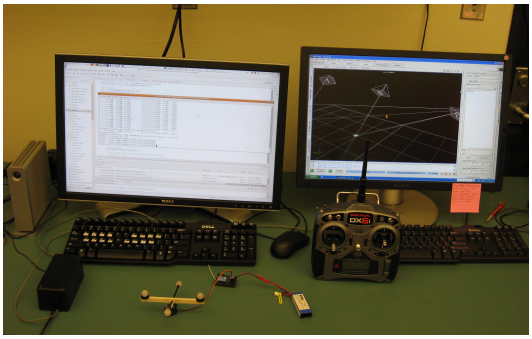


Figure 2.10: The experiment used to estimate the total time-delay of the experimental setup.

2.3.2 Estimating Velocity and Acceleration from Vicon Data

Vicon motion capture cameras measure the position and orientation of the rotorcraft. The velocity and acceleration information, however, are extremely important for control and guidance systems. This section deals with designing Kalman filters to provide an optimal estimate of the linear and angular velocities as well as accelerations given the Vicon camera measurements. The kinematics model that is used for angular velocity estimate is based on Wiener process velocity model. The kinematic model that is used for acceleration estimate is based on Wiener process acceleration model [99, 100].

Angular Velocity Estimate

The velocity and acceleration can be estimated by incorporating the full dynamics model of the helicopter from the position and orientation measurements via a Kalman filter. However, this approach may involve observability issues due to large number of state variables in the helicopter dynamics model as well as real-time computation issues. As a result, simple kinematic model is used to obtain these estimates. The following simple model in continuous-time captures the dynamics of the helicopter's angular rates

$$\frac{d}{dt}\begin{bmatrix}\theta\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}\theta\\ \dot{\theta}\end{bmatrix} + \begin{bmatrix}0\\ 1\end{bmatrix}\nu(t) \tag{2.1}$$

where $\nu(t)$ is the process noise with with variance q. Note that the states are $[\theta \;\; \dot{\theta}]$. The first state does not contain any process noise in this model due to the fact that it represent a kinematics relationship which is valid in theory. This model is a Wiener process velocity model. The discrete-time process model with sampling time T is given by

$$\begin{bmatrix}\theta_{k+1}\\ \dot{\theta}_{k+1}\end{bmatrix} = \overbrace{\begin{bmatrix}1 & T\\ 0 & 1\end{bmatrix}}^{\Phi}\begin{bmatrix}\theta_k\\ \dot{\theta}_k\end{bmatrix} + \overbrace{\begin{bmatrix}0\\ 1\end{bmatrix}}^{G}\nu_k \tag{2.2}$$

In addition, the disctere-time measurement model is

$$z_k = \overbrace{\begin{bmatrix}1 & 0\end{bmatrix}}^{H}\begin{bmatrix}\theta_k\\ \dot{\theta}_k\end{bmatrix} + \eta_k \tag{2.3}$$

The next step is to obtain the covariance of discrete-time process noise Q_k which can be obtained by performing the following integral over a sampling period

$$\begin{aligned} Q_k &= \int_0^T \Phi(\tau) G\, q\, G^T \Phi(\tau) d\tau \\ &= q \begin{bmatrix} \frac{1}{3}T^3 & \frac{1}{2}T^2 \\ \frac{1}{2}T^2 & T \end{bmatrix} \end{aligned} \tag{2.4}$$

The Kalman filter time propagation for state vector x and covariance matrix P have the following form

$$\begin{aligned} x_{k+1}^- &= \Phi_k x_k^+ \\ P_{k+1}^- &= \Phi_k P_k^+ \Phi_k^T + Q_k \end{aligned} \tag{2.5}$$

The measurement update proceeds by computing the Kalman gain

$$K = P^- H^T (H P^- H^T + R)^{-1} \tag{2.6}$$

where R is the variance of the measurement noise. Finally, the state and the estimation error covariance matrix is updated according to

$$\begin{aligned} \hat{x}^+ &= \hat{x}^- + K\, r \\ P^+ &= (I - KH) P^- (I - KH)^T + K R K^T \end{aligned} \tag{2.7}$$

It is to be noted that the steady state solution of the equations (2.5) to (2.7) exists and known as $\alpha - \beta$ filter [99].

Translational Velocity and Acceleration Estimates

In this section, the previous filter is extended to include the acceleration state. The kinematic model that is used for this purpose is a Winier process acceleration model in continuous-time

$$\frac{d}{dt} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \nu(t) \tag{2.8}$$

The state variables are position, velocity and acceleration. The discrete-time process model with sampling time T is given by

$$\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \\ \ddot{x}_{k+1} \end{bmatrix} = \overbrace{\begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}}^{\Phi} \begin{bmatrix} x_k \\ \dot{x}_k \\ \ddot{x}_k \end{bmatrix} + \overbrace{\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}}^{G} \nu_k \tag{2.9}$$

Similar to the previous section, the covariance of discrete-time process noise Q_k can be obtained by performing the following integral over a sampling period

$$\begin{aligned} Q_k &= \int_0^T \Phi(\tau) G\, q\, G^T \Phi(\tau) d\tau \qquad (2.10) \\ &= q \begin{bmatrix} \frac{1}{20}T^5 & \frac{1}{8}T^4 & \frac{1}{6}T^3 \\ \frac{1}{8}T^4 & \frac{1}{3}T^3 & \frac{1}{2}T^2 \\ \frac{1}{6}T^3 & \frac{1}{2}T^2 & T \end{bmatrix} \end{aligned}$$

The equations for the time propagate and measurement update of the Kalman filter are the same as equations (2.5) to (2.7) in the previous section. Note that the steady state solution of the filter exists and known as $\alpha - \beta - \gamma$ filter [99].

Tunning the Filter Parameters

There are two free parameters in the filters discussed above. The measurement noise variance R and the process noise variance q. The measurement noise variance R can be obtained from the sensor specifications. The filter is therefore designed by tunning the parameter q. According to equation (2.4) and (2.10), changes in velocity and acceleration over the sampling period are the order of $\sqrt{q\,T}$, which can be used as a guideline in choosing q. In addition, we check the $3-\sigma$ covariance bound of the residuals to ensure that the filter is consistent. If not, q is changed to obtain satisfactory results. Finally, the autocorrelation of the filter's residuals are monitored to ensure that the residuals are approximately white noise. Figure 2.11 shows the trajectory of the free flying helicopter. The measured data from this flight were used to tune the estimation filter parameters. Figure 2.12 to 2.15 show the quality of the estimates. Figure 2.16 and 2.17 show the $3-\sigma$ covariance bound of the residuals and finally the autocorrelation of the filter's residuals are shown in Figures 2.18 and 2.19.

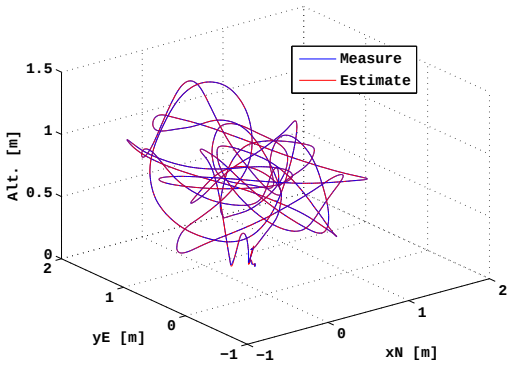
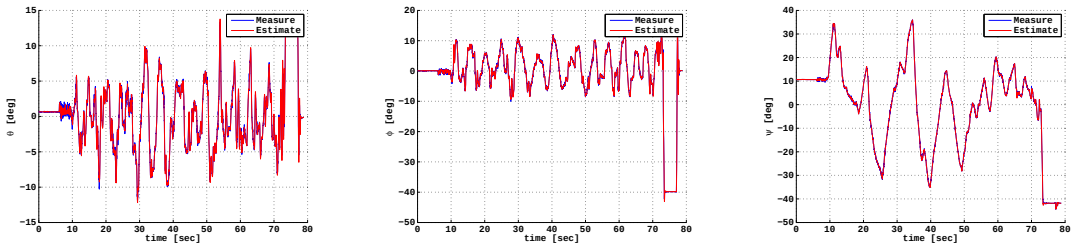
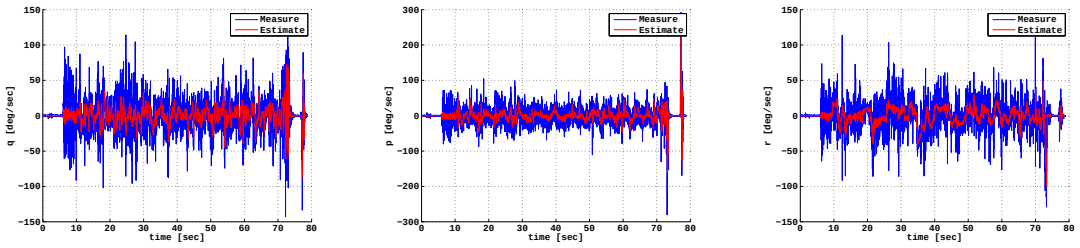


Figure 2.11: Trajectory of the free flying helicopter and the estimated trajectory. The measurements from this flight was used to tune the estimation filters.



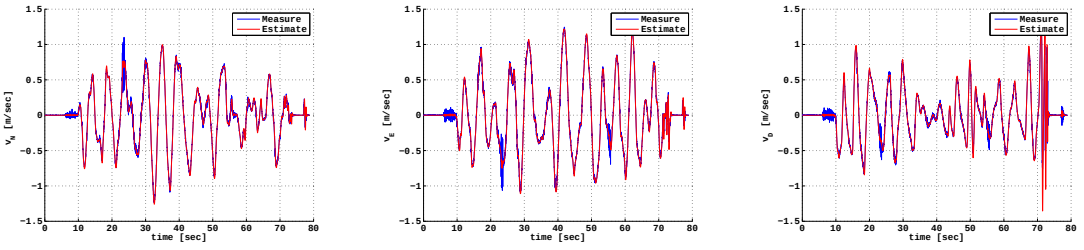
(a) pitch angle. (b) roll angle. (c) yaw angle.

Figure 2.12: The measured attitude angles and the estimated ones.



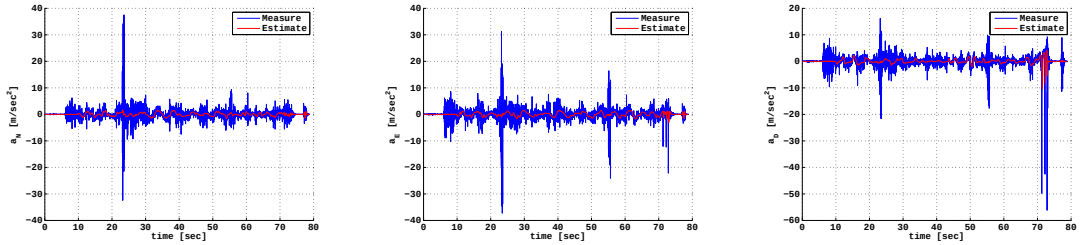
(a) pitch rate. (b) roll rate. (c) yaw rate.

Figure 2.13: The estimated angular rates. The 'measured' refers to direct differentiation of attitude angles.



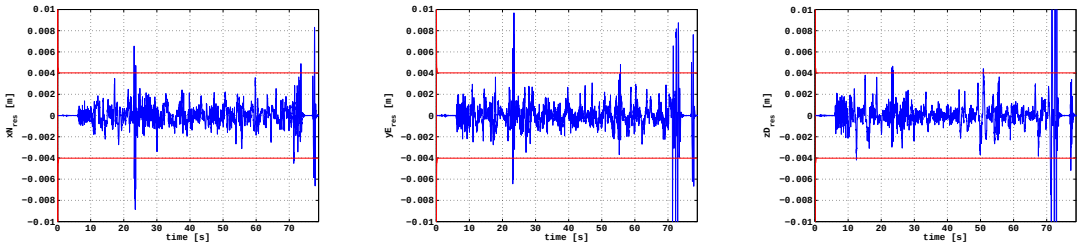
(a) velocity in x direction. (b) velocity in y direction. (c) velocity in z direction.

Figure 2.14: The estimated velocities in the inertial frame. The 'measured' refers to direct differentiation of position.



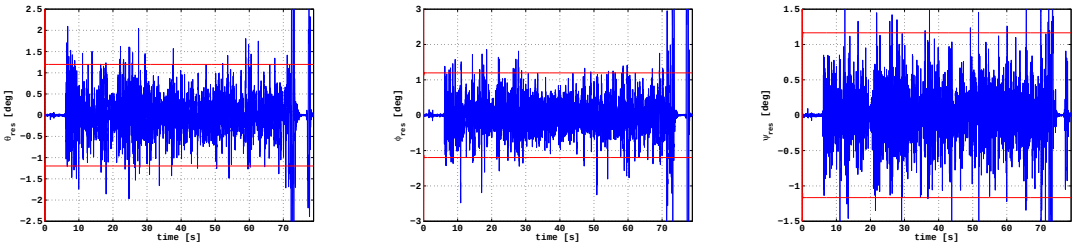
(a) acceleration in x direction. (b) acceleration in y direction. (c) acceleration in z direction.

Figure 2.15: The estimated acceleration in the inertial frame. The 'measured' refers to direct double differentiation of position.



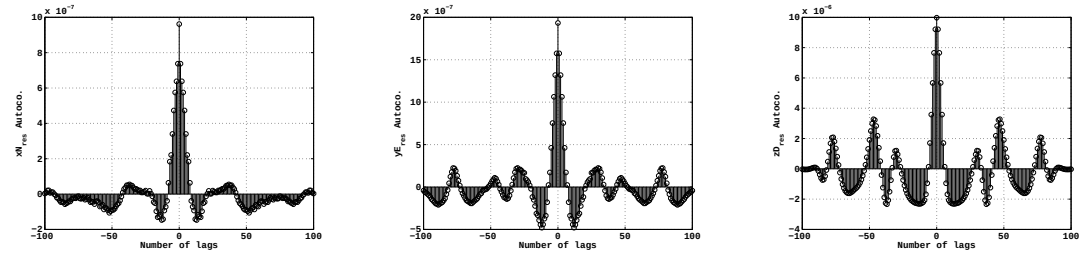
(a) x position residual. (b) y position residual. (c) z position residual.

Figure 2.16: Residuals of the position vector. The corresponding 3σ-bound of the covariance matrix is also plotted.



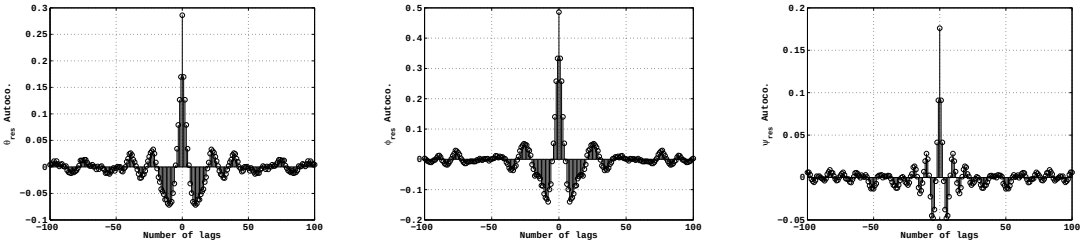
(a) pitch angle residual. (b) roll angle residual. (c) yaw angle residual.

Figure 2.17: Residuals of the attitude vector. The corresponding 3σ-bound of the covariance matrix is also plotted.



(a) x position residual autocorrelation. (b) y position residual autocorrelation. (c) y position residual autocorrelation.

Figure 2.18: The autocorrelation of the position residual vector.



(a) pitch residual autocorrelation. (b) roll residual autocorrelation. (c) yaw residual autocorrelation.

Figure 2.19: The autocorrelation of the attitude residual vector.

2.3.3 Motion Tracking Kinematic Data Consistency Check

Vision based motion tracking systems like the Vicon cameras have become a popular tool for indoor flight test experiments [28, 15]. It enables to measure the attitude and position of the vehicle with respect to a fixed reference fame within the experimental space. In many flight vehicle system identification methods, the

velocity and acceleration in the body frame is used for the identification purpose as equation of motion is often written in the body reference frame. Before using the flight data for system identification it is necessary to verify the accuracy of the data and validate the coordinate transformations. This verification can be achieved using a kinematic data consistency check.

Using kinematic relationship among measurement quantities is a common way to verify the accuracy of the flight measurements. The general idea is that a subset of measurements used as an input to the kinematics equations which are solved and used with output equations to generate reconstructed value for different subset of measurements (see, e.g., [20] and [24]). If all the measurements are compatible then the reconstructed output will match the measurements of the same output quantities up to some random measurement noise. For example, the measured angle of attack and sideslip angle must match with those reconstructed from the measured linear acceleration and angular rates. However, without on-board sensors such as accelerometers and gyros, the aircraft kinematic equations can not be used for data consistency check. Hence, we conducted an experiment in which the helicopter was hanged from the ceiling with a long wire as a free pendulum (see Figure 2.20). Due to the long length of the wire, the helicopter's body xy plane remains almost perpendicular to the wire. This provides a way to compare the body acceleration computed form pure kinematics relationship with the components of the gravity vector as will be described in details in this section.

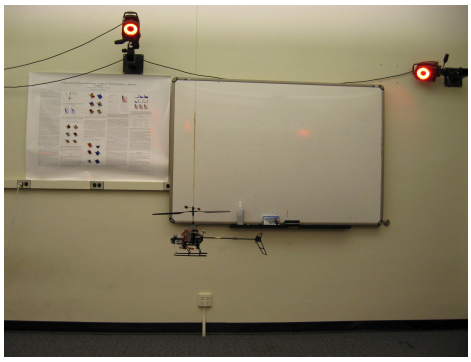


Figure 2.20: The experiment used for kinematic consistency. Helicopter was hanged from the ceiling with a wire to swing like a free pendulum. Vicon motion tracking system recorded its position and orientation.

The Body frame $\{B\}$ acceleration can be obtained from the Vicon measurements by double differentiation the position in Global frame $\{G\}$ and transform

that to the Body frame

$$a^B = C_G^B a^G \tag{2.11}$$

where $a^G = \frac{d^2}{dt^2} r^G$ and C_G^B is the rotation matrix from Global to Body frame. The Vicon system computes the orientation of the object via an attitude parametrization called exponential map [101]. Given α_x, α_y and α_z are the component of exponential map, the rotation matrix from the Body to the Global frame is obtained via the following rotation matrix

$$C_G^B = \begin{bmatrix} c + (1-c)\hat{\alpha}_x^2 & (1-c)\hat{\alpha}_x\hat{\alpha}_y + s\hat{\alpha}_z & (1-c)\hat{\alpha}_z\hat{\alpha}_x - s\hat{\alpha}_y \\ (1-c)\hat{\alpha}_x\hat{\alpha}_y - s\hat{\alpha}_z & c + (1-c)\hat{\alpha}_y^2 & (1-c)\hat{\alpha}_z\hat{\alpha}_y + s\hat{\alpha}_x \\ (1-c)\hat{\alpha}_x\hat{\alpha}_z + s\hat{\alpha}_y & (1-c)\hat{\alpha}_y\hat{\alpha}_z - s\hat{\alpha}_x & c + (1-c)\hat{\alpha}_z^2 \end{bmatrix} \tag{2.12}$$

where

$$\begin{aligned} \alpha &= \sqrt{\alpha_x^2 + \alpha_y^2 + \alpha_z^2} \\ \hat{\alpha}_x = \alpha_x/\alpha,\ \hat{\alpha}_y &= \alpha_y/\alpha,\ \hat{\alpha}_z = \alpha_z/\alpha \\ c &= \cos(\alpha),\ s = \sin(\alpha) \end{aligned} \tag{2.13}$$

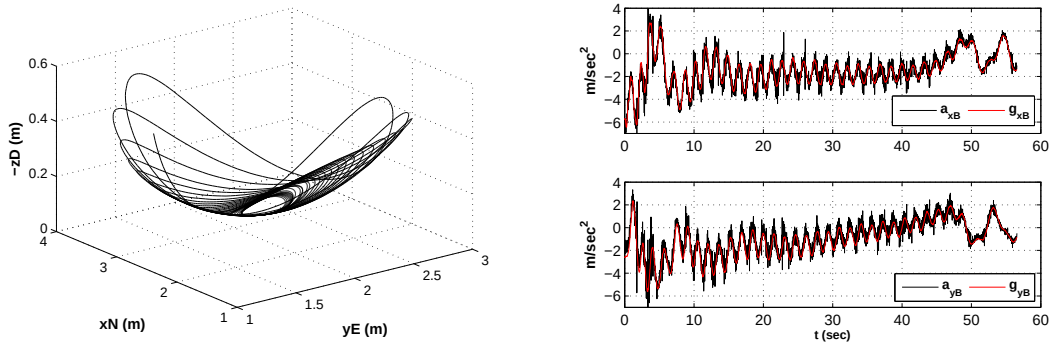
The corresponding Euler's angles from the Global to the Body frame are then computed as

$$\begin{aligned} \theta &= \sin^{-1}\left(-C_G^B(1,3)\right) \\ \phi &= \tan^{-1}\left(C_G^B(2,3)/C_G^B(3,3)\right) \\ \psi &= \tan^{-1}\left(C_G^B(1,2)/C_G^B(1,1)\right) \end{aligned} \tag{2.14}$$

The external forces that are applied to the helicopter in the pendulum motion in Figure 2.20 are cable tension T and gravity g. By expressing all the forces in the Body frame we have

$$\frac{1}{m}\sum F^B = a^B \Longleftrightarrow \begin{bmatrix} -g\ \sin(\theta) \\ g\ \cos(\theta)\sin(\phi) \\ g\ \cos(\theta)\cos(\phi) - T/m \end{bmatrix} = \begin{bmatrix} a_{xB} \\ a_{yB} \\ a_{zB} \end{bmatrix} \tag{2.15}$$

In the above equation a_{xB} and a_{yB} can be computed according to equation (2.11)



(a) 3D trajectory of the pendulum tracked by the Vicon system.

(b) components of the gravity vector resolved in body frame during the pendulum motion (red) and components of body acceleration vector computed from Vicon measurements (black).

Figure 2.21: Validating the consistency of data captured by the motion tracking system.

and compared to components of gravity vector in body frame. Figure 2.21, shows the x and y component of gravity vector and the body acceleration computed from Vicon measurements. As it can be seen, the two quantities are very close which validates the measurements and the corresponding coordinate transformation.

Chapter 3

System Identification Modeling of Blade-CX2 Coaxial Helicopter

This chapter addresses the system identification modeling and flight characteristics analysis of Blade-CX2 coaxial helicopter. The focus is on the lateral-longitudinal dynamics of the Blade-CX2 helicopter in particular the distinctive characteristics of the attitude and translational dynamics found in this particular coaxial helicopter such as low damped coupled Rotor-Fuselage-Stabilizerbar (FRS) and the Phugoid mode. For indoor flight in the absence of wind, the altitude-heading dynamics are weakly coupled to the lateral-longitudinal dynamics. In addition, for most indoor flight operations, the altitude and heading of the helicopter can be regulated by a simple PI controller with empirically tuned gains. Such simple control augmentation, provide satisfactory decoupling between altitude-heading and lateral-longitudinal dynamics. Interested readers are referred to [102] for modeling and analysis of the coupled rotor-inflow and drive train dynamics of the Blade-CX2 helicopter in axial flight.

In this chapter we first develop a reduced-order parametric model for identification of the helicopter's lateral-longitudinal dynamics. We then present the frequency domain parameter identification results and time domain validation. Finally, analysis and interpretation of the identified derivatives, which focus on the coupled rotor-fuselage and Phugoid modes are presented.

3.1 Overview of Blade-CX2 Rotor Mechanism

The helicopter physical specifications are listed in Table 2.1 (also see Figure 2.2 for illustration of the system). The coaxial rotor system features a free floating Bell style stabilizer-bar, which is coupled to the upper rotor, and a swash-plate actuated lower rotor. Both rotors are two bladed and driven by individual electric motors. Figure 3.1 shows the mechanization of the co-axial rotor system. The stabilizer bar and the upper rotor are coupled to the inner drive shaft. The lower rotor is driven by the outer shaft. The rotor system does not posses a collective pitch control. Instead, rotor thrust is controlled via the rotor speed. This setup is common in very small rotorcrafts.

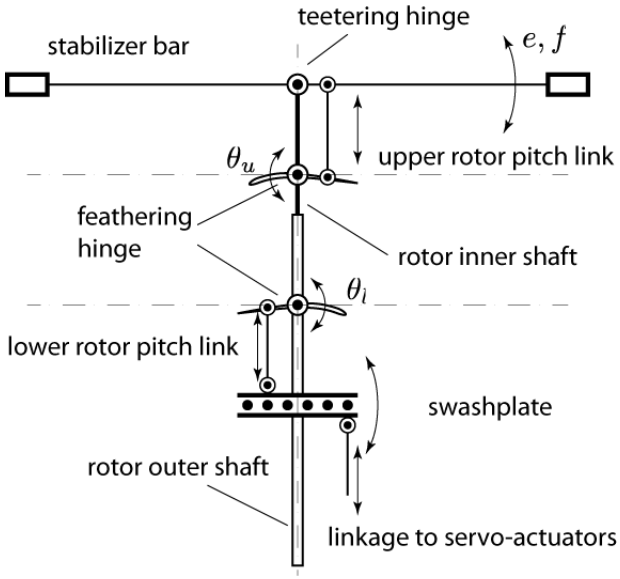


Figure 3.1: Blade-CX2 rotors and stabilizer-bar mechanization.

3.2 Development of the Parameterized Model

In this section we develop the parameterized model for the Blade-CX2 coaxial helicopter. The model accounts for the rotor and stabilizer-bar as well as the coupled rotor-fuselage dynamics. We start the section by component-wise modeling of the coaxial rotor system. We then propose an implicit model for the upper

rotor and stabilizer-bar which reduces the order of the state space model as well as number of unknown parameters. Finally, we obtain a complete dynamic model of the helicopter by combining rotor and fuselage dynamics.

3.2.1 Component-wise Modeling of the Co-Axial Rotor System

The dynamics of the rotor system play an important role in the helicopter behavior. The rotor system of Blade-CX2 helicopter consists of three components: Bottom rotor, top rotor and stabilizer-bar. As illustrated in Figure 3.1, the bottom rotor is connected to the swashplate and its cyclic pitch is controlled by two servo actuators. The top rotor receives cyclic pitch inputs from the stabilizer-bar; the stabilizer-bar acts as swashplate for the top rotor. As a result, the flapping of the top rotor is controlled by the stabilizer-bar. Overall, the bottom rotor is an actively controlled system, while the top rotor is passive. The following subsections describe detailed mathematical model of the components of the rotor system.

Stabilizer-Bar Model

The stabilizer-bar is a metal rod with two cylindrical weights attached to each ends. The bar is attached to the rotor shaft through a simple hinge. In order to model the stabilizer-bar we conducted an experiment where we subject the entire rotor system to a constant angular rate while the stabilizer-bar was rotating. Under these conditions, the stabilizer-bar's tip-path-plane lags behind the shaft's perpendicular plane without phase offset (with respect to the angular rate). This behavior is consistent with an unrestrained, centrally hinged rotor, and can be approximated by a first order system with a time-constant τ_s, i.e.

$$\begin{aligned} \dot{e} &= -\frac{1}{\tau_s} e - q \\ \dot{f} &= -\frac{1}{\tau_s} f - p \end{aligned} \tag{3.1}$$

where variables e and f are longitudinal and lateral flapping angles.

Top Rotor Model

A pitch link connects the stabilizer-bar and the top rotor (see Figure 3.1). As opposed to conventional single rotor helicopter where pilot has full authority to

control the swashplate and flapping of the blades, in the Blade-CX2 coaxial helicopter the top rotor is not directly actuated by the pilot and therefore present controllability challenges. The Blade-CX2 coaxial helicopter is more stable in hover compared to a similar scale single-rotor counterparts. However, since part of the forces and moments imparted on the body originate from the top rotor assembly, which is passively controlled by the stabilizer-bar, modeling this system in details is challenging.

As known from helicopter theory, for an articulated rotor system, the cyclic input must be applied 80 to 90 degrees in advance of the point where the maximum moment output is desired [13]. However, for a coaxial rotor system, this offset angle is different. Reference [3], evaluated the stability and control characteristic of the full scale Sikorsky ABC coaxial helicopter and reported that "*For the ABC rotor, the cyclic pitch input must be applied 30 to 40 degrees in advance of the desired output*". This behavior is illustrated in Figure 3.2 which is taken from [3].

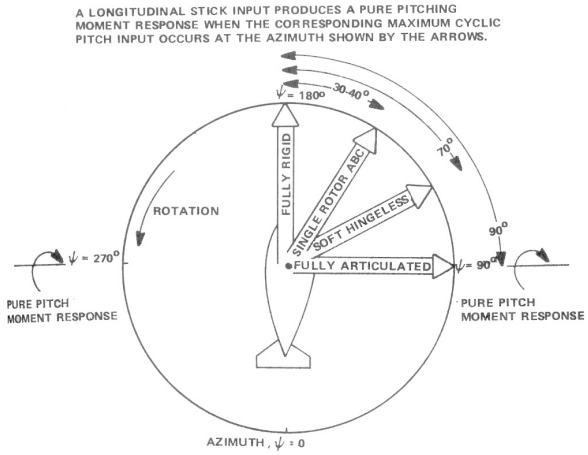


Figure 3.2: Comparison between the rotor system response to cyclic control input for different coaxial and single rotor systems, adapted from [3].

A similar discussion was also reported in reference [103] for a small-scale coaxial helicopter in which the phase lag between the cyclic input and rotor flapping is described to be $\psi_s = 45$ degrees. Based on inspection of the Blade-CX2, there exists a similar 45 degree mechanical phase offset between the stabilizer-bar and the top rotor, as well as for the bottom rotor swashplate.

When the body experiences a pitch rate, the stabilizer-bar inertia resists to that motion and remains approximately horizontal (acts like a gyroscope). Then

the linkage that connects the stabilizer-bar and top rotor, applies a cyclic pitch command to the top rotor with 45 degree offset with respect to the shaft motion. Now, since the top blade flapping motion response to this cyclic input with 45 degrees lag, the tip-path-plane (TPP) of the top rotor exactly tilts to the direction opposite to the shaft. This behavior provides damping to the angular rate of the body and stabilizes the helicopter at hover. In fact, the top rotor and stabilizer-bar form a mechanical rate feedback system whose purpose is to provide stability for the helicopter against sudden changes in angular rate induced by disturbances or inexperienced pilots. In addition, this way most of the cross-coupling typical in single-rotor helicopter is mechanically removed by the 45 degrees adjustment of the the stabilizer-bar.

In order to simplify the mathematical model of the top rotor, we consider the above mentioned phenomena as internal effects and we provide a model which captures the net response of the top rotor due to angular rate inputs. The upper rotor system can then be modeled as a standard first-order tip-path-plane (TPP) system with longitudinal and lateral flapping variables (c and d) as follows

$$\begin{aligned} \dot{c} &= -\frac{c}{\tau} - q + C_e e \\ \dot{d} &= -\frac{d}{\tau} - p + D_f f \end{aligned} \tag{3.2}$$

The first term in the right hand side of equation (3.2) is due to the rotor response which acts as a first order system with time-constant τ. The second term is the gyroscopic effect and the last term in the right hand side is due to the interaction of the stabilizer-bar and the top rotor in which the stabilizer-bar flapping angles e and f act as effective cyclic pitch for the upper rotor with gearing C_e and D_f (see [104]).

Bottom Rotor Model

The bottom rotor of Blade-CX2 helicopter takes its cyclic inputs directly from the swash plate. Therefore, flapping of the bottom rotor is controlled by the swash plate through periodic changes of the pitch angle of the blades as it rotates. Since the phase lag in rotor flapping response for this helicopter is 45 degrees, the manufacturer has rotated the swash plate by the same amount to compensate

the coupling effect that would otherwise result. The flapping dynamics for the bottom rotor can be modeled as a standard first-order tip-path-plane system with longitudinal and lateral flapping variables (a and b) as follows

$$\begin{aligned} \dot{a} &= -\frac{a}{\tau} - q + A_{lon}\delta_{lon} + A_{lat}\delta_{lat} + A_u u + A_v v \\ \dot{b} &= -\frac{b}{\tau} - p + B_{lon}\delta_{lon} + B_{lat}\delta_{lat} + B_u u + B_v v \end{aligned} \tag{3.3}$$

where δ_{lon} and δ_{lon} are longitudinal and lateral cyclic inputs, respectively. The cross coupling A_{lat} and B_{lon} mainly account for the swash plate 45 degrees phase lag with respect to the cyclic inputs δ_{lon} and δ_{lon}. This derivatives also account for the remaining unmodeled coupling in the system which is not removed mechanically. In addition, A_u, A_v, B_u and B_v account for the aerodynamic effects of the rotor: There are several rotor aerodynamics effects that are unique to flight at low speed and contribute to the coupling. One effect involves the *non-uniform distribution of the induced velocity* flow pattern across the rotor. In hover, the pattern in almost symmetrical, however, at forward flight the distribution is in such a way that the induced flow at the leading edge of the disc is essentially zero and increasing at the rear of the rotor [105, 13]. The change causes the blade over the nose to see an increase in the angle of attach which make the counter-clockwise rotating blade to flap up on the left side. In addition to non-uniform aerodynamics effects at low forward speeds, there are two other effects that can be categorized as uniform aerodynamics effect: The first one is called *coning effect* [105] which causes the component of forward speed to be more upward with respect to the blade over the nose than the blade over the tail. Thus, the counter-clockwise rotating blade flaps up on the left side due to the coning effect. The second effect is called *speed stability* [105] which is created due to increase in relative speed of the advancing blade and decrease in the relative speed of retreating blade in forward flight. As a result, the counter-clockwise rotating blade tends to flap up on the nose producing a nose up pitching moment with respect to helicopter center of gravity which decreases the helicopter's speed. We will provide an analysis on these derivatives based on inflow velocity and first principle modeling.

3.2.2 Reduced-Order Rotor-Stablizer-bar Model: Merged Top Rotor-Stablizer-bar

Modeling all three interacting rotor subsystems provides a complete mechanical representation which may provide a more accurate prediction of the rotorcraft behavior. However, detailed componentwise modeling leads to an increased number of parameters and without direct measurement of the blade and stabilizer-bar flapping motion, not all parameters may be determined from the available data. In the absence of such measurements, we propose an implicit model for the upper rotor and stabilizer-bar which reduces the order of the state space model as well as number of unknown parameters.

To derive the flapping equation for the merged upper rotor-stabilizer-bar, consider again the tip-path-plane (TPP) flapping equations of the upper rotor

$$\begin{aligned} \dot{c} &= -\frac{c}{\tau} - q + C_e e \\ \dot{d} &= -\frac{d}{\tau} - p + D_f f \end{aligned} \tag{3.4}$$

Differentiating equation (3.4) leads to

$$\begin{aligned} \ddot{c} &= -\frac{\dot{c}}{\tau} - \dot{q} + C_e \dot{e} \\ \ddot{d} &= -\frac{\dot{d}}{\tau} - \dot{p} + D_f \dot{f} \end{aligned} \tag{3.5}$$

Substituting for $\dot{e}$ and $\dot{f}$ from the stabilizer-bar TPP model (equation 3.1), yields

$$\begin{aligned} \ddot{c} &= -\frac{\dot{c}}{\tau} - \dot{q} + C_e(-\frac{1}{\tau_s} e - q) \\ \ddot{d} &= -\frac{\dot{d}}{\tau} - \dot{p} + D_f(-\frac{1}{\tau_s} f - p) \end{aligned} \tag{3.6}$$

Now, according to equation (3.4), we have

$$\begin{aligned} e &= \frac{1}{C_e}(\dot{c} + \frac{c}{\tau} + q) \\ f &= \frac{1}{D_f}(\dot{d} + \frac{d}{\tau} + p) \end{aligned} \tag{3.7}$$

Substituting equation (3.7) into equation (3.6) gives

$$\begin{aligned} \ddot{c} &= -\dot{c}(\frac{1}{\tau}+\frac{1}{\tau_s}) - \frac{1}{\tau\,\tau_s}c - q(\frac{1}{\tau_s}+C_e) - \dot{q} \\ \ddot{d} &= -\dot{d}(\frac{1}{\tau}+\frac{1}{\tau_s}) - \frac{1}{\tau\,\tau_s}d - p(\frac{1}{\tau_s}+D_f) - \dot{p} \end{aligned} \tag{3.8}$$

By neglecting the fast acceleration effects $\ddot{c}$, $\ddot{d}$, $\dot{q}$ and $\dot{p}$, the equation for the implicit upper rotor-stabilizer-bar is obtained according to the following first order differential equation

$$\begin{aligned} \dot{c} &= -\frac{1}{\tau^{'}}c - C_q q \\ \dot{d} &= -\frac{1}{\tau^{'}}d - D_p p \end{aligned} \tag{3.9}$$

where

$$\begin{aligned} \tau^{'} &= \tau + \tau_s \\ C_q &= \frac{\tau}{\tau_s+\tau}(1+\tau_s C_e) \\ D_p &= \frac{\tau}{\tau_s+\tau}(1+\tau_s D_f) \end{aligned} \tag{3.10}$$

Note that c and d in equation (3.9) represent the state variables of the merged upper rotor-stabilizer-bar. Finally, by including the lower rotor dynamics, the equivalent reduced order coaxial rotor system model has the following form

$$\begin{aligned} \dot{a} &= -\frac{a}{\tau} - q + A_{lon}\delta_{lon} + A_{lat}\delta_{lat} + A_u u + A_v v \\ \dot{b} &= -\frac{b}{\tau} - p + B_{lon}\delta_{lon} + B_{lat}\delta_{lat} + B_u u + B_v v \\ \dot{c} &= -\frac{1}{\tau^{'}}c - C_q q \\ \dot{d} &= -\frac{1}{\tau^{'}}d - D_p p \end{aligned} \tag{3.11}$$

3.2.3 Coupled Rotor-Fuselage Dynamics

A key aspect of helicopter dynamics is the dynamical coupling between the rotors and the helicopter fuselage. Omitting this coupling effect has been shown to limit the accuracy of the helicopter model in the medium to high frequency range

both for full scale [21] and small-scaled helicopters [104]. In order to obtain the complete model of the helicopter, the rigid body and the coaxial rotor dynamics should be coupled. The lateral-longitudinal rigid body dynamics around the cruise trim condition (u_0, v_0, w_0) are as follows

$$\begin{aligned} \dot{u} &= -w_0 q + v_0 r - g\sin(\theta) + X/m \\ \dot{v} &= u_0 r + w_0 p + g\cos(\theta)\sin(\phi) + Y/m \\ \dot{p} &= -qr(I_{yy} - I_{zz})/I_{xx} + L/I_{xx} \\ \dot{q} &= -pr(I_{zz} - I_{xx})/I_{yy} + M/I_{yy} \end{aligned} \tag{3.12}$$

where X and Y are the total aerodynamic forces acting on the body. L and M are also external rolling and pitching moments, respectively.

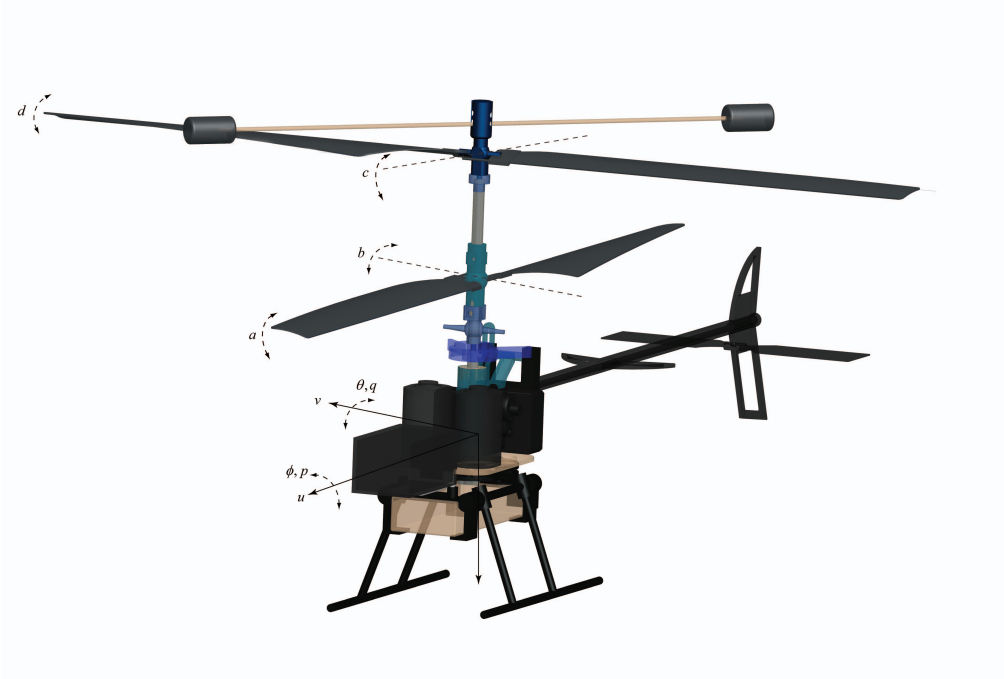


Figure 3.3: Blade-CX2 body reference frame with key state variables used in the dynamic model

Coupled rotor-fuselage dynamics can be obtained by expanding the forces and moments acting on the helicopter body in terms of flapping derivatives (i.e., M_a, M_c, L_b and L_d) and damping derivatives (i.e., X_u and Y_v). Hence, by linearizing the nonlinear dynamics (3.12) around hover and slow flight conditions and incorporating the rotor dynamics, the final linear coupled rotor-fuselage dynamics are

parametrized as follows

$$
\begin{aligned}
\dot{u} &= X_u u + X_\theta \theta \\
\dot{v} &= Y_v v + Y_\theta \theta \\
\dot{\theta} &= q \\
\dot{\phi} &= p \\
\dot{q} &= M_a a + M_c c \\
\dot{p} &= L_b b + L_d d \\
\dot{a} &= -\frac{a}{\tau} - q + A_{lon}\delta_{lon} + A_{lat}\delta_{lon} + A_u u + A_v v \\
\dot{b} &= -\frac{b}{\tau} - p + B_{lon}\delta_{lon} + B_{lat}\delta_{lat} + B_u u + B_v v \\
\dot{c} &= -\frac{1}{\tau'} c - C_q q \\
\dot{d} &= -\frac{1}{\tau'} d - D_p p
\end{aligned}
\tag{3.13}
$$

Equation set (3.13) is the parametric model for lateral-longitudinal dynamics of the Blade-CX2 helicopter. The unknown parameters can be identified using various time domain and frequency domain system identification techniques. In the remainder of this paper we provide a brief overview of the frequency domain identification method that was used in this paper. The reader is referred to [24], [21] and [20] for a complete discussion on various system identification techniques.

3.3 Frequency Domain System Identification

For the system identification, **Comprehensive Identification from FrEquency Responces** (CIFER [106]) analysis tool was used. CIFER originally developed for rotorcrafts and has been successfully applied to a several full scale and small-salced helicopters (see, e.g., [107] and [23]). A comprehensive description of CIFER is given in Reference [21]. The overall procedure for system identification in frequency domain illustrated by the flowchart in Figure 1.4.

3.3.1 Overview

In the following we provide a brief high-level description of the methods that are used in CIFER procedure: After performing the flight test, the spectral analysis is applied to the collected input and output flight data to compute input autospectra $\hat{G}_{uu}$, output autospectra $\hat{G}_{yy}$ and cross spectra $\hat{G}_{uy}$. To reduce the level of random errors in spectral estimates, the method of overlapped windowing is applied. Additional processing is also applied to condition or correct these frequency responses to alleviate the effects of multiple partially correlated inputs that can occur in MIMO systems. The result are the conditioned frequency responses and partial coherences [108]. The frequency response $\hat{H}(f)$ can be then estimated directly from the smoothed spectral estimates at each frequency f via

$$\hat{H}(f) = \frac{\hat{G}_{yy}(f)}{\hat{G}_{uy}(f)} \tag{3.14}$$

Another product of the smoothed spectral function is the coherence function estimate $\hat{\gamma}^2_{uy}$ defined by

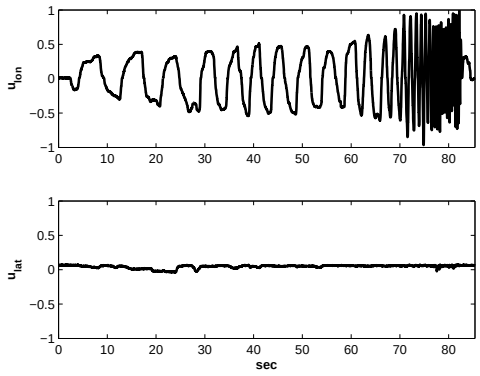
$$\hat{\gamma}^2_{uy} = \frac{\left|\hat{G}_{uy}(f)\right|^2}{\left|\hat{G}_{uu}(f)\right| \left|\hat{G}_{yy}(f)\right|} \leq 1 \tag{3.15}$$

The coherence function indicates how well the output y is linearly correlated with the input u over the examined frequency range. If the process under investigation were perfectly linear without any input-output noise, the coherence would have the ideal value of 1. However, noise in the measured output signal, nonlinearity that can not be described by the frequency response function and finally noise associated with the unknown or unmeasured input cause a reduction in the coherence.

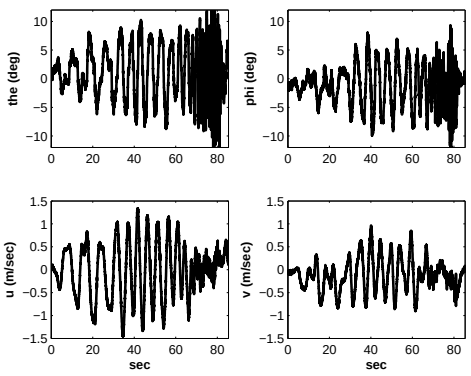
3.3.2 Collection of Flight Data

A series of flight experiments was conducted during which the pilot applied a frequency sweep control excitation to one of the lateral-longitudinal control inputs. The inputs were applied via a remote control (RC) unit. No stability augmentation were active (at the exception of the mechanical stabilizer-bar, which acts as a pseudo-attitude rate regulator [104]). Frequency content that could be applied is

limited from bellow due to excitation of the translational motion and the finite tracking volume provided by the motion tracking cameras. At higher frequencies, the range is limited by the constraints of the pilot's neuro-muscular system. The lowest sweep frequency that could be applied to the system within our $4m$ by $4m$ lab area was around 0.1 Hz (for reference, the phugoid mode for our helicopter was found to be at about 0.25 Hz). An example of the input channel recorded during a longitudinal cyclic sweep is shown in Figure 3.4. Notice that four non-consecutive frequency inputs for each channel were concatenated in order to achieve a rich spectral density.



(a) Longitudinal frequency sweep applied to the helicopter.



(b) The corresponding attitudes and velocities which shows a strong coupling in the system.

Figure 3.4: Time histories from a sample flight data.

During the flight test, all the pilot control inputs as well as helicopter position and attitude are recorded at 50 Hz. Velocities in body frame are obtained following the coordinate transformation in Section 2.3.3. The measurements that are used for system identification are pitch and roll angles as well as the body lateral and longitudinal velocities (i.e., $y = [\theta, \phi, u, v]$). The pilot's longitudinal and lateral cyclic pitch commands $u = [u_{lon}, u_{lat}]$ are recorded along with the other measurements. We apply a lowpass filter with a corner frequency of 10 Hz to both the input and output data to remove high-frequency noise due to vibration and sensor noise. In system identification it is important to apply the same smoothing filters to both input and output data to preserve the correlation between data. Filtering and smoothing is less important in frequency domain system identification since measurement noise that is uncorrelated with the input, such as vibration,

will automatically be discarded when extracting the frequency responses. Further, vibrations found in our helicopter occur at frequencies that are higher than the helicopter rigid-body dynamics and coupled rotor-fuselage mode.

3.3.3 Identification of the Helicopter Servo Actuators

The helicopter servo actuator (Efilte servo [7]) was identified separately from the airframe so that its contribution can be studied. The frequency response was obtained by applying frequency sweep to the servo actuators and measuring the response of the servo actuator by Vicon motion tracking cameras. This was done by attaching reflective markers to the servo (see Figure 2.10) and measuring the markers position while the frequency sweep is applied to the servo. The servo deflection can be easily obtained from the position of the individual markers.

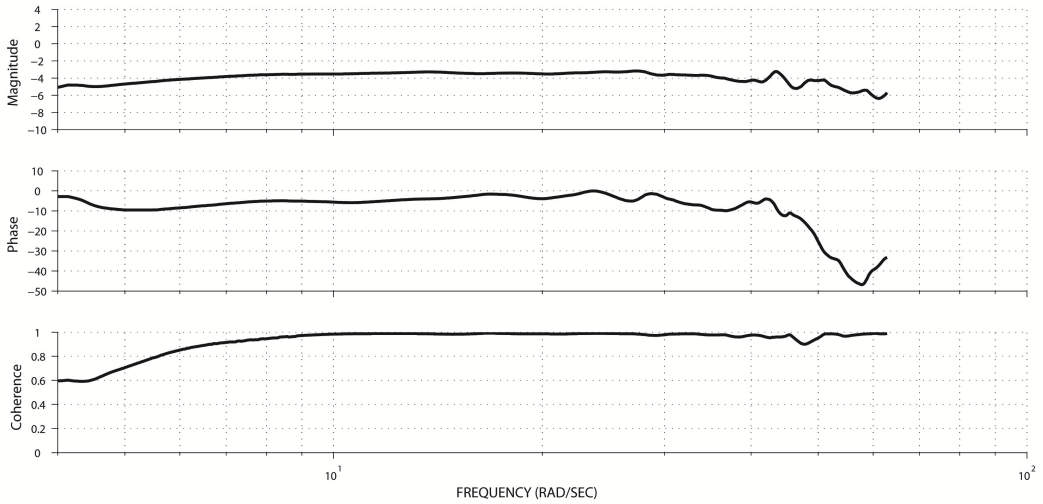


Figure 3.5: Frequency response of the Efilte servo actuator.

Figure 3.5 shows the frequency response of the servo actuator. It is to noted that the low coherence below the frequency of $5\,rad/sec$ is due to the fact that the servo was not excited enough at low frequencies. As it can be seen, the servo bode plot is almost constant over the helicopter frequency of operation (until $25\,rad/sec$). Therefore, we did not include the servo actuator dynamics model in the entire system dynamics.

3.3.4 Identification of Bare-Airframe Parametric Model

In the parameter identification we seek a set of parameters in the system's state space model that produces a frequency response matrix which best fits the estimated frequency response obtained from the measured data. To be more specific, the solution of the MIMO identification problem involve determining the unknown parameters vector Θ in matrices M, F and G of the general state space form

$$\begin{aligned} M(\Theta)\dot{x} &= F(\Theta)x + G(\Theta)u \\ y &= H_0 x + H_1 \dot{x} \end{aligned} \tag{3.16}$$

that produces a frequency response matrix $T(s)$

$$T(s) = [H_0 + sH_1](sI - M^{-1}F)^{-1}M^{-1}G \tag{3.17}$$

which most closely matches the estimated frequency response $\hat{T}(s)$. The associated cost function to be minimized is the summed cost for the n_{TF} transfer functions [21]

$$J(\Theta) = \sum_{i=1}^{n_{TF}} \left\{ \frac{20}{n_\omega} \sum_{\omega_1}^{\omega_n} W_\gamma \left[W_g(|T| - |\hat{T}|)^2 + W_p(\angle T - \angle \hat{T})^2 \right] \right\} \tag{3.18}$$

where n_ω frequency points exist in the each frequency range $(\omega_1, \ldots, \omega_n)$ and W_γ, W_g and W_p are weighting functions. Because of the fact that the frequency responce errors are a nonlinear function of unknown parameters Θ, the solution for the minimum cost function $J(\Theta)$ must be obtained using an iterative process. It turned out that a pattern search method such as secant method (see e.g. [109]) which are used in CIFER is much better suited for this application than gradient based minimization methods. The secat method is very robust to sharp changes and discontinuities in the optimization space and also has been found to be adept at coping both poor initial guesses and local minima.

While the optimization process returns the best matches for the frequency responses and the associated parameter vector Θ, a measure of accuracy of the identification parameters in needed. The Cramer-Rao (CR) inequality provides a fundamental basis for the theoretical accuracy analysis. It provides the minimum

expected standard deviation σ_i of the estimated parameter Θ_i

$$\sigma_i \geq CR_i \tag{3.19}$$

The Cramer-Rao bound of the i*th* identified parameter is determined from the associated diagonal elements of the inverse Hessian matrix

$$CR_i = \sqrt{\mathbf{H}_{ii}^{-1}} \tag{3.20}$$

where

$$\mathbf{H} = \frac{\partial^2 J(\Theta)}{\partial \Theta \partial \Theta} \tag{3.21}$$

Analytical expression for the Hessian matrix can be obtained by taking derivatives of the cost function $J(\Theta)$ with respect to the identification vector Θ. The reader is referred to [110] and [21] for the complete discussion.

3.4 Results

This section focuses on validating the identified parameters of the system both in frequency and time domain. This provides an insight to the dynamics range in which the identified parametric model is most accurate. In addition, the Cramer-Rao bounds for the identified parameters can be regarded as standard deviation of the error in each parameter [21] which can be used to build a parametric uncertainty model for the system. Finally, we present an interpretation of the identified model parameters based on key physical characteristics of the co-axial helicopter. The interpretation provides explanations on some important flight dynamics characteristics.

3.4.1 Frequency Domain Validation and the Identified Parameters

The identified parameters and corresponding CR bounds of Blade-CX2 helicopter are shown in Table 3.1. The predicted frequency responses from the identified model are compared with the frequency responses obtained from the flight test in Figures 3.6 and 3.7. These graphs demonstrate well agreement between the model and flight tests for the attitude dynamics of the helicopter. In addition, the on-axis velocities are very well matched with the experimental results. The results

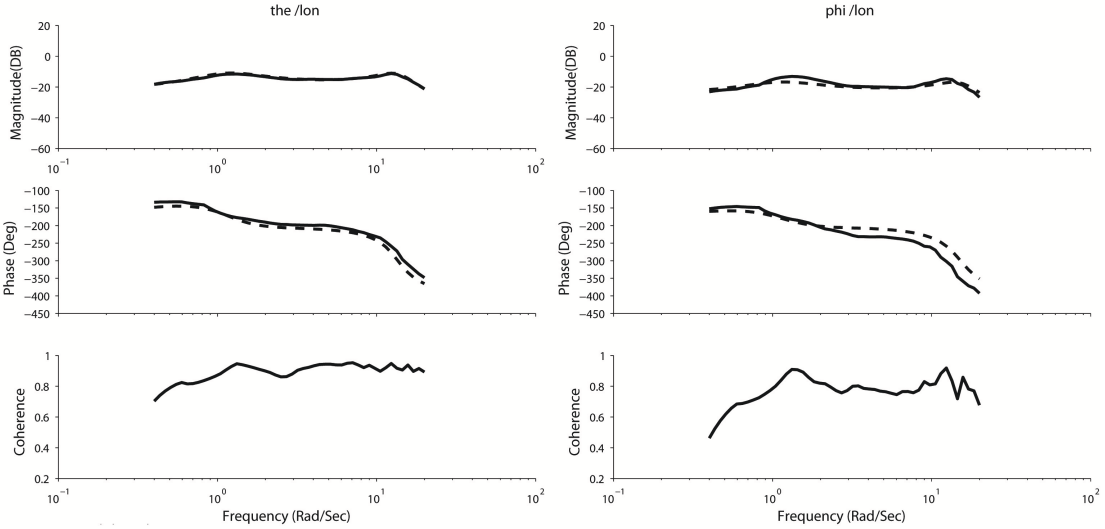
in Figure 3.7, however, show that there is a phase offset in the off-axis velocity frequency responses compare to the experiment, especially at mid-range to high-range frequencies. The main reason for the mismatch is the low coherence at the corresponding frequencies. Due to space limitations in our indoor experimental facility, we were unable to achieve better coherences for the off-axis velocities.

Table 3.1: Parameters of the Blade-CX2 coaxial helicopter obtained from system identification techniques along with their Cramer-Rao bounds.

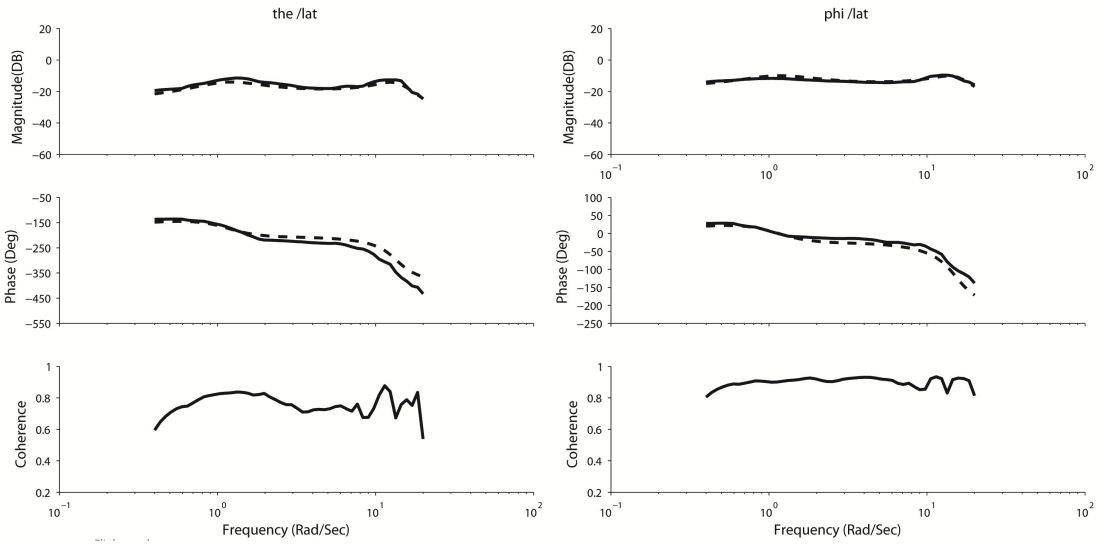
Param.	Nom. value	CR bound	Param.	Nom. value	CR bound
X_u	-0.446	10.3 %	Y_v	-0.604	7.1 %
A_u	2.289	10.9%	B_v	-2.141	8.1%
X_θ	-9.810	—	Y_ϕ	9.810	—
$1/\tau$	-36.170	21.0 %	$1/\tau^{'}$	-1.789	8.4%
M_a	189.600	18.7 %	M_c	154.500	3.4 %
L_b	265.400	8.7	L_d	186.200	3.5%
A_{lon}	-4.157	10.3%	A_{lat}	-2.908	10.5%
B_{lon}	-2.232	8.2%	B_{lat}	4.900	7.9%

3.4.2 Time Domain Validation

In order to validate the frequency based identified model, a time domain validation preferably with dissimilar forms of control excitation must be done. Doublets are often used for this purpose as they have a relatively wide frequency spectrum that is of interest for most flight control systems. A comparison between the time histories of the model response and flight test is presented in Figure 3.8. As we would expect from the frequency responses, there is a very good agreement between the time histories of the predicted attitude and the flight test. In addition, the on-axis velocities are matched well with the flight test. The time histories of off-axis velocities, however, exhibit a phase offset between the predicted response and the experiment. This behavior was expected from the frequency responses of the off-axis velocities.

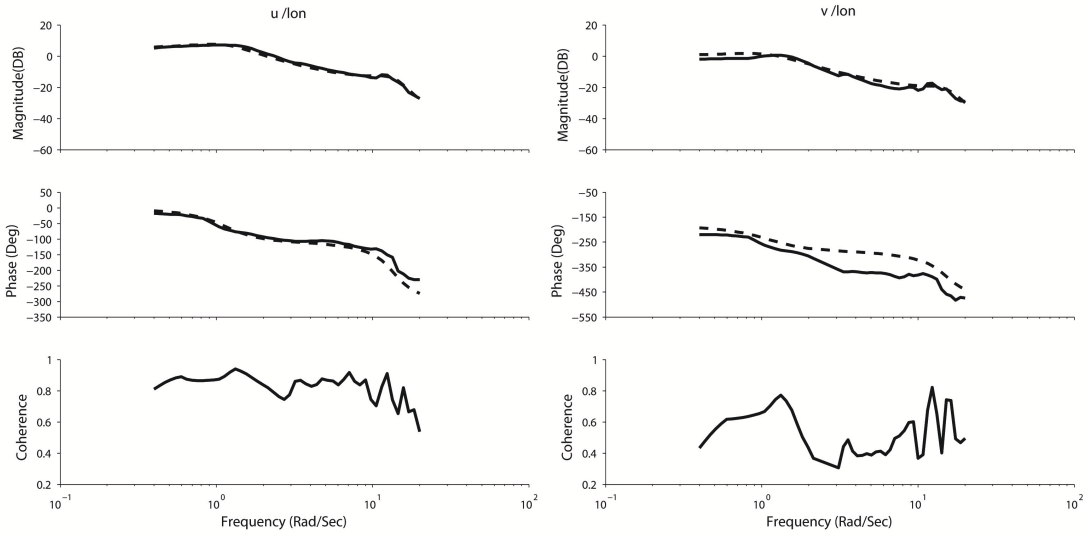


(a) attitude response to longitudinal input

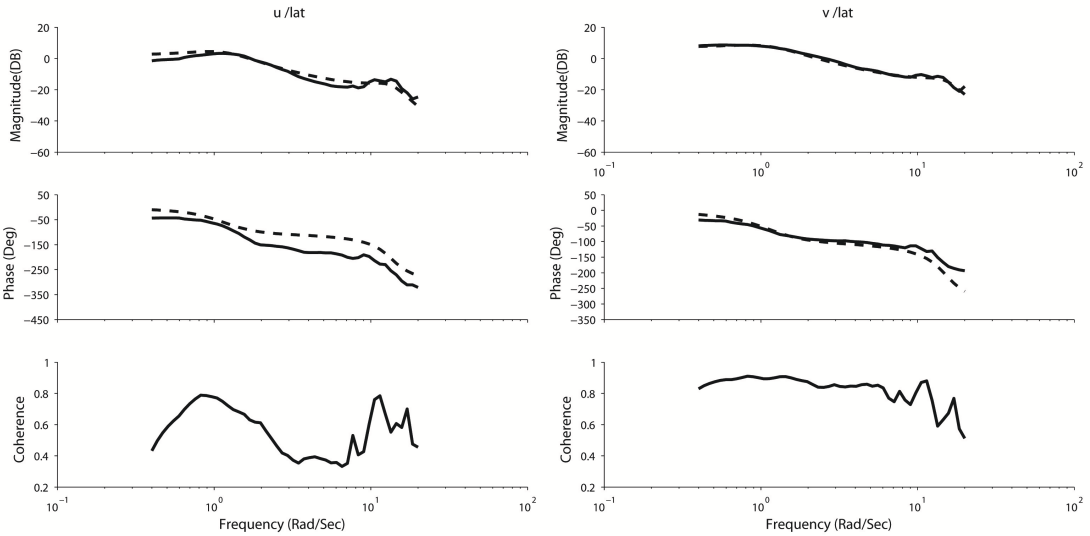


(b) attitude response to lateral input

Figure 3.6: Comparison between the frequency responses computed from the flight data (solid) and those derived from the identified model (dashed).

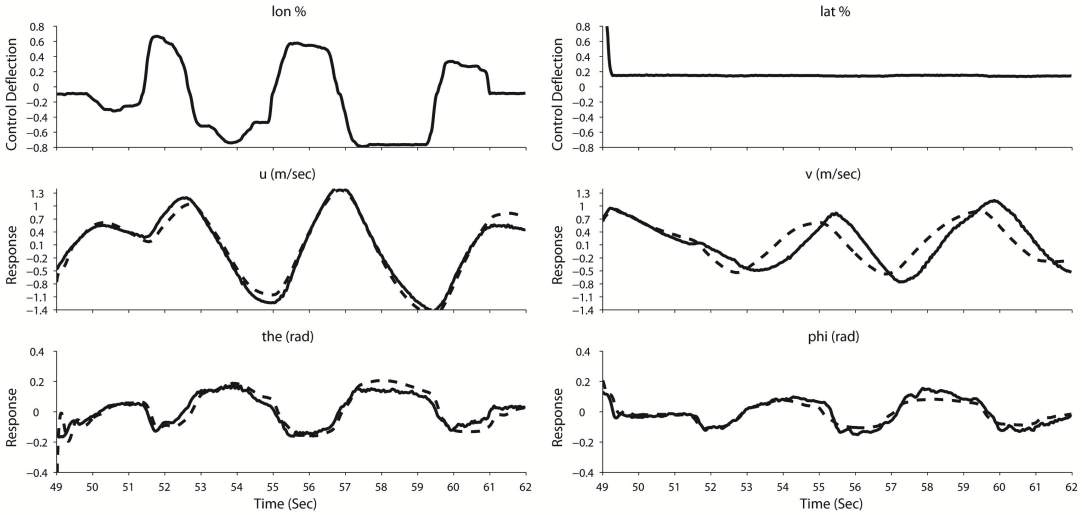


(a) velocity response to longitudinal input

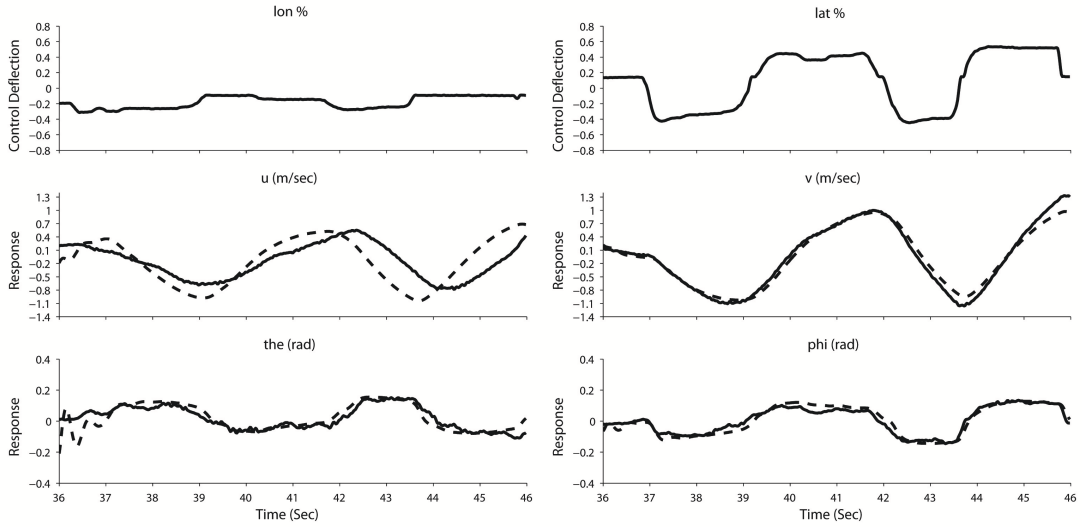


(b) velocity response to lateral input

Figure 3.7: Comparison between the frequency responses computed from the flight data (solid) and those derived from the identified model (dashed).



(a) attitude and velocity responses to longitudinal doublet



(b) attitude and velocity responses to lateral doublet

Figure 3.8: Comparison between the time responses predicted by the identified model and the responses obtained during flight testing.

3.4.3 Analysis and Interpretation of the Identified Derivatives

In this section we provide an interpretation of the derivatives in the parametric model of the helicopter based on physical principles. The estimate of the derivative based on first principles is not meant to provide a better model matching in

frequency responses obtained from the system identification procedure. The purpose however, is: **(i)** to provide physical understanding of the key derivatives and when possible relate the identified values to physical quantities, this also provides an understanding about how the derivatives are originated and interact with the model; and **(ii)** to have an initial starting point for the non-convex optimization problem discussed in Section 3.3.4 that finds the derivatives.

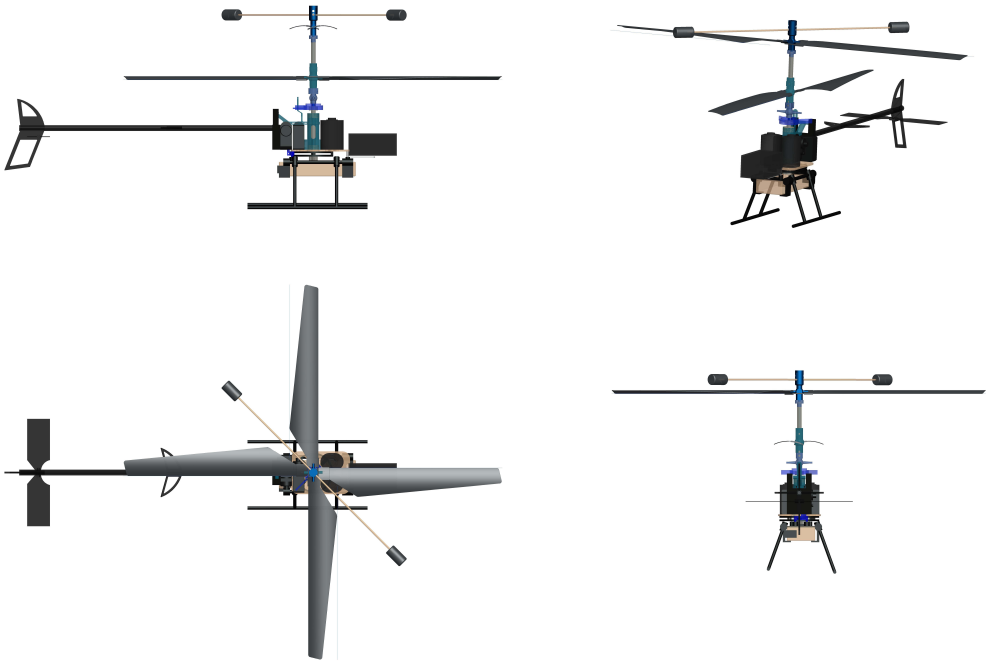


Figure 3.9: CAD modeling of Blade-CX2 which was used for calculating the C.G. and moment of inertia

We start this section by listing the physical parameters of Blade-CX2 helicopter in Table 3.2. The remainder of this section provides the analysis and interpretation for the identified parameters.

Body Force Derivatives

- X_θ and Y_ϕ

  These body force derivatives capture the effect of fuselage pitch and roll angles on the fuselage x- and y-forces. The main contribution to this effect is the rotation of gravity forces with respect to the body frame. The gravity

Parameter	Value
m, helicopter mass [kg]	0.224
I_{xx}, roll moment of inertia [kgm^2]	4.6792e-004
I_{yy}, pitch moment of inertia [kgm^2]	7.5782e-004
I_β, blade flapping inertia [kgm^2]	2.98634e-5
Ω, nominal rotor speed [rad/sec]	188
cl_α, blade lift curve slope (*obtained from load test experiment*)	4.58
$\bar{v}_i$, average inflow speed (*obtained from hotwire experiment*) [m/s]	3.0
G_{su}, gear ratio from stabilizer-bar flap angle to top rotor cyclic pitch	0.86
$G_{\delta\theta}$, gear ratio from servo deflection to blade cyclic pitch	0.17
R, rotor radius [cm]	15.8
$\bar{c}$, mean chord length [cm]	2.2
h_u, distance between upper and lower rotor [cm]	5.2
h_l, distance between lower rotor's hub and c.g [cm]	4.5
θ_p, average blade pitch angle [deg]	5.0

Table 3.2: Parameters of Blade-CX2 coaxial helicopter

vector in the body frame is

$$g^B = \begin{bmatrix} -g\sin(\theta) \\ g\cos(\theta)\sin(\phi) \\ g\cos(\theta)\cos(\phi) \end{bmatrix} \tag{3.22}$$

The derivatives X_θ and Y_ϕ can then be obtain via

$$\begin{aligned} X_\theta &= -g\cos(\theta_e) \\ Y_\phi &= -g\sin(\phi_e)\sin(\theta_e) + g\cos(\theta_e)\cos(\phi_e) \end{aligned} \tag{3.23}$$

where θ_e and ϕ_e are pitch and roll angles at equilibrium. Note that the force derivatives are normalized by the helicopter mass. Assuming small angle yields $X_\theta \cong -g$ and $Y_\phi \cong g$. In the optimization procedure for system identification, we fixed these derivatives to their theoretical values.

- X_a and Y_b

  These body rotor-force derivatives capture the effect of tilting of the rotor thrust vector with the flapping angles a and b. The total thrust vector in

the body frame of the helicopter is given by

$$T^B = \frac{T_l}{\sqrt{1-\sin(a)^2\sin(b)^2}} \begin{bmatrix} -\sin(a)\cos(b) \\ \sin(b)\cos(a) \\ -\cos(a)\cos(b) \end{bmatrix} + \frac{T_u}{\sqrt{1-\sin(c)^2\sin(d)^2}} \begin{bmatrix} -\sin(c)\cos(d) \\ \sin(c)\cos(d) \\ -\cos(c)\cos(d) \end{bmatrix} \tag{3.24}$$

where T_u and T_l are the upper rotor and lower rotor thrust magnitude, respectively. In addition, a, b, c and d are the rotor flapping angles which were introduced before. For single rotor helicopters, one can assume that in hover and slow flight conditions, the thrust vector is approximately equal to the weight of the helicopter and and directed perpendicular to the rotor plane and hence the derivatives X_a and Y_b can be often predicted [104]. However, for the lumped rotor formulation used for the identification of the coaxial Blade-CX2 helicopter, as it can be seen from equation (3.24), the thrust does not depend exclusively on the flapping angles a and b but also depend on the upper rotor flapping c and d. In fact, each of the lower and upper rotor thrust vector can have different directions while the helicopter is at hover (see, e.g., Figure 3.11). In addition, we observed that adding these derivatives to the model did not provide a better frequency domain model matching. In addition, the corresponding CIFER Cramer-Rao bound for these derivative were relatively large (more that 50 percent). Consequently, we dropped these derivatives from the parametric model.

Rotor Derivatives and Coupled Rotor Fuselage Mode

- τ and $\tau^{'}$

  Rotor time-constant is the key parameter of the tip-path-plane dynamics. Rotor time-constant is related to the lock number and rotor speed. The lock number, γ, which is the ratio of aerodynamic force to the inertial force acting on the blade is

$$\gamma = \frac{\rho \bar{c}\, c l_\alpha R^4}{I_\beta} = 3.76 \tag{3.25}$$

The theoretical value of main time-constant can be predicated via [13]

$$\tau = \frac{16}{\gamma\Omega} = 0.0226 \; sec \tag{3.26}$$

which is relatively close to the identified value of $\tau = 0.0276$. In addition, we conducted an experiment with a high frame rate camera to measure the rotor time-constant from a different approach. Figure 3.10 shows the sequence of lower blade flapping during the transient motion. It can be seen that the lower rotor reaches its maximum flapping less that 0.1 sec, which predicts the time-constant of $\tau \leq 0.63 \times 0.1 = 0.063$ sec (note that in this experiment, the input was not a pure step and therefore, the actual rotor time-constant is expected to be less than the predicted value).

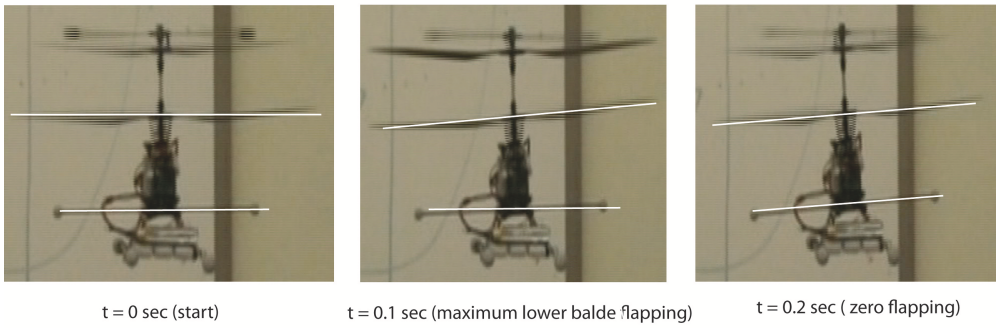
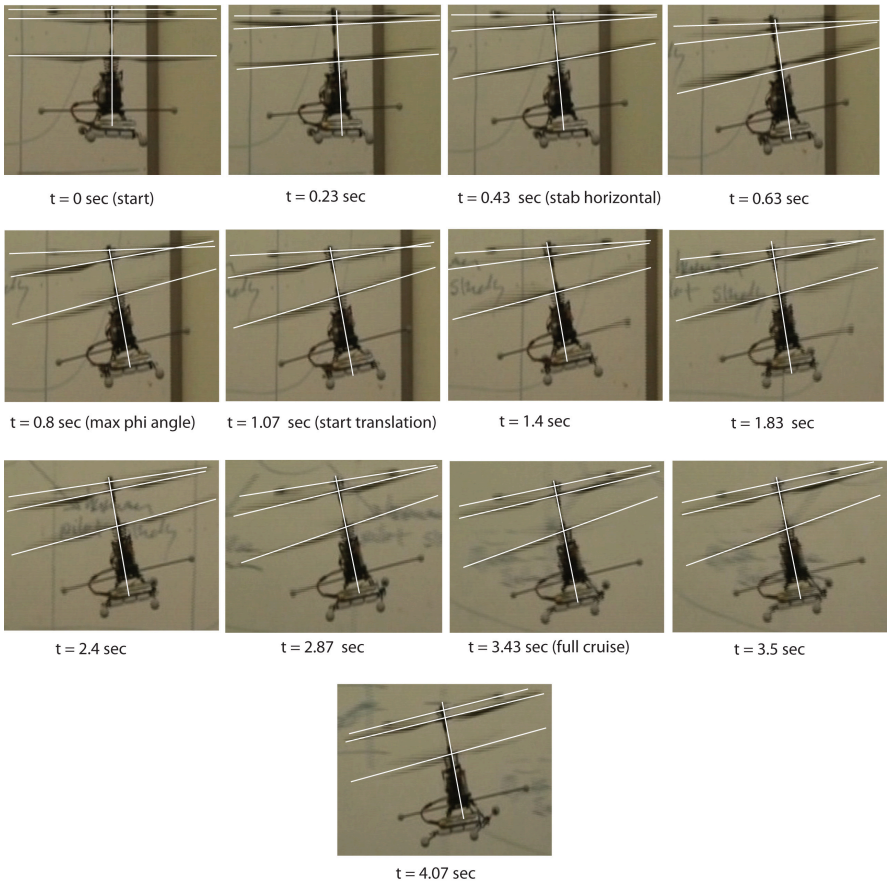
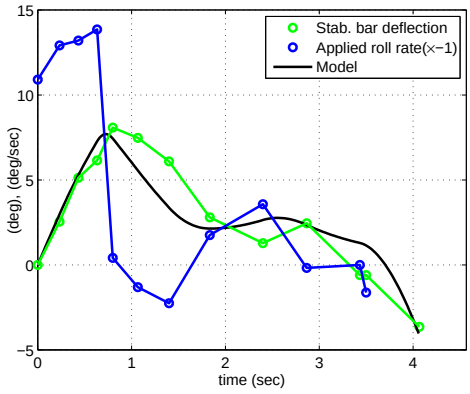


Figure 3.10: Sequence of images showing the evolution of lower rotor flapping over time.

In order to obtain an estimate of the stabilizer-bar time-constant τ_s, we conducted a similar experiment with the high frame rate camera in which the response of the rotors and stabilizer-bar was recorded for a longer period. In this experiment, a roll rate command applied to the system by the pilot and the response of the helicopter was captured. Figure 3.11-a shows the sequence of the motion. The resulting roll rate due to the control input and the stabilizer-bar deflection (the output) were extracted from the images in Figure 3.11-a by manual tracing. The extracted data is plotted in Figure 3.11-b along with the response of a first order system with time-constant $\tau_s = 1$. The predicted value of the merged upper-rotor stabilizer-bar time-constant is then equal to $\tau^{'} = \tau + \tau_s = 1.0276$ seconds. This is comparable with the identified value of $\tau^{'} = 0.56$ seconds.



(a) Sequence of helicopter's body and rotors motion after applying lateral cyclic input captured with a high frame rate camera.



(b) Extracted roll rate and stabilizer-bar deflection angle from the above video sequence. The response of the first order linear model fitted to the extracted data is also shown.

Figure 3.11: Estimating the stabilizer-bar time-constant.

- Blade Stiffness k_β

  In order to calculate the equivalent blade stiffness, we conducted an experiment in which the pilot excites the system with sinusoidal longitudinal (or lateral) input at its damped natural frequency. Since at this point the helicopters's velocity is almost zero, the moments acting on the fuselage are mainly created by the flapping angles. Figure 3.12, shows the primary flapping moments on the fuselage around the longitudinal axis. According to this figure, the following set of equations describes the system dynamics

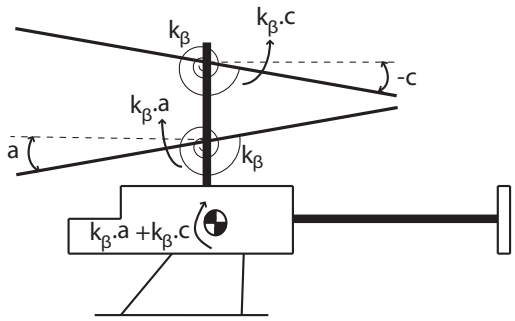


Figure 3.12: Primary flapping moments acting on the fuselage around the longitudinal axis.

$$\begin{aligned} I_{yy}\dot{q} &= k_\beta a + k_\beta c \\ \dot{a} &= -\frac{a}{\tau} - q \\ \dot{c} &= -\frac{1}{\tau'}c - C_q q \end{aligned} \tag{3.27}$$

where I_{yy} is pitching moment of inertia of the helicopter. It is to be noted that we assumed the two blades have the same stiffness and the moment is only created by the equivalent springs (we ignored the moment generated by the thrust vector tilt). Therefore, equation (3.27) can be written as

$$\frac{d}{dt}\begin{bmatrix} q \\ a \\ c \end{bmatrix} = \begin{bmatrix} 0 & k_\beta/I_{yy} & k_\beta/I_{yy} \\ -1 & -1/\tau & 0 \\ C_q & 0 & -1/\tau' \end{bmatrix}\begin{bmatrix} q \\ a \\ c \end{bmatrix} \tag{3.28}$$

Given the damped natural frequency of the system obtained from the experiment to be around $11\,rad/sec$, the oscillatory mode of equation (3.28)

predicts the value of $k_\beta = 0.091$.

It is known in the literature that the blade stiffness can be obtained by static experiments in which a set of known forces are applied at the blade tip and the corresponding blade deflection is measured. However, based on our experiments, we convinced that this method does not predict accurate blade stiffness for the Blade-CX2 helicopter. One reason might be due to the coaxial blade mechanism which makes it challenging to obtain an equivalent blade stiffness for the two rotors from the static experiments.

- M_a, L_b, M_c and L_d
  These derivatives represent the effect of the flapping on the pitching and rolling moments that the rotor imparts to the helicopter. The moments acting on the helicopter are mainly due to thrust vector tilt as well as hub stiffness effect. Assuming small angle approximation we get [104]

$$M = [T_l h_l + k_\beta]\, a + [T_u(h_l + h_u) + k_\beta]\, c \tag{3.29}$$

$$L = [T_l h_l + k_\beta]\, b + [T_u(h_l + h_u) + k_\beta]\, d \tag{3.30}$$

  where T_l and T_u are the magnitude of lower and upper rotor thrust, respectively. By taking partial derivatives we can obtain the following pitch and roll moment derivatives

$$\begin{aligned} M_a &= [T_l h_l + k_\beta]\, / I_{yy} \\ M_c &= [T_u(h_l + h_u) + k_\beta]\, / I_{yy} \\ L_b &= [T_l h_l + k_\beta]\, / I_{xx} \\ L_d &= [T_u(h_l + h_u) + k_\beta]\, / I_{xx} \end{aligned} \tag{3.31}$$

  where the moment derivatives are normalized by the helicopter moment of inertia (the numerical value for the moment of inertias were obtained form the CAD modeling and are included in Table 3.2). Note that we assume each rotor produces half of the total thrust (i.e., $T_u = T_l = mg/2$). In addition, k_β is the blade stiffness which was obtained in previous section to be $k_\beta = 0.091$. Based on the above considerations, the theoretical value for the flapping moment derivatives are $M_a = 185.3$, $M_c = 260.7$, $L_b = 300.1$

and $L_d = 422.2$. This is comparable with the identified value of $M_a = 189.6$, $M_c = 154.5$, $L_b = 265.4$ and $L_d = 186.2$.

Speed Derivatives and Phugoid Mode

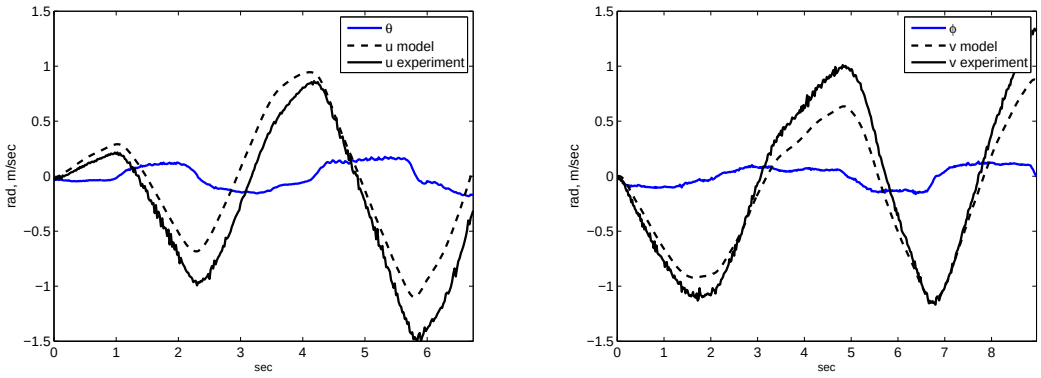
- X_u and Y_v

  These derivatives capture the effect of the body longitudinal and lateral speed on the body x and y forces. The change in speed causes variation in rotor flapping, rotor thrust as well as rotor drag force. Mathematical modeling of these effects is very complicated for coaxial rotors and direct measurement of the blade motion and the inflow velocity due to the helicopter change in speed are necessary for the purpose of modeling. In the absence of these measurements, we proposed another method for predicting these derivatives based on time domain simulation of the translational dynamics

$$\begin{aligned} \dot{u} &= X_u u + X_\theta \theta \\ \dot{v} &= Y_v v + Y_\theta \theta \end{aligned} \tag{3.32}$$

  Given the value of X_θ and Y_ϕ from previous part, equation (3.32) can be simulated by applying measured θ and ϕ as inputs. By comparing the simulated (u,v) versus the measured ones, we can predict the unknown derivatives. We obtain the values of $X_u = -0.4$ and $Y_v = -0.5$ which is comparable with the identified value of $X_u = -0.446$ and $Y_v = -0.604$. Figure 3.13, shows the response of the translational dynamics.

  In addition, these derivatives dictate the maximum speed based on the peak steady-state roll and pitch angles. Taking equation (3.32) at steady-state yields: $X_u = X_\theta \theta_{ss}/u_{ss}$ and $Y_v = Y_\phi \phi_{ss}/v_{ss}$. Hence, another approach to predict or verify these derivatives is by measuring the ratio of the pitch (roll) angle to the helicopter longitudinal (lateral) speed in steady-state conditions.



(a) Time response of the longitudinal translation model $\dot{u} = X_u u + X_\theta \theta$ versus experiment: the input is θ and the output is u.

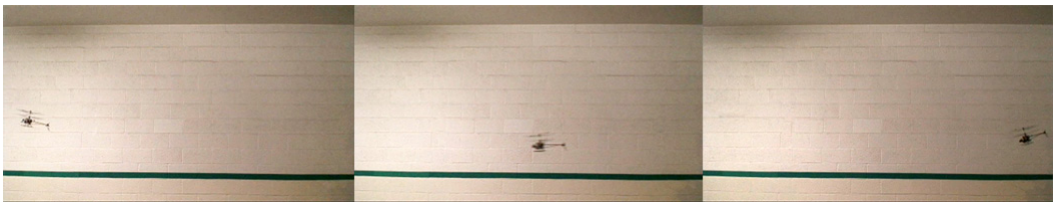
(b) Time response of the lateral translation model $\dot{v} = Y_v v + Y_\phi \phi$ versus experiment: the input is ϕ and the output is v.

Figure 3.13: Time domain analysis for predicting derivatives the X_u and Y_v.

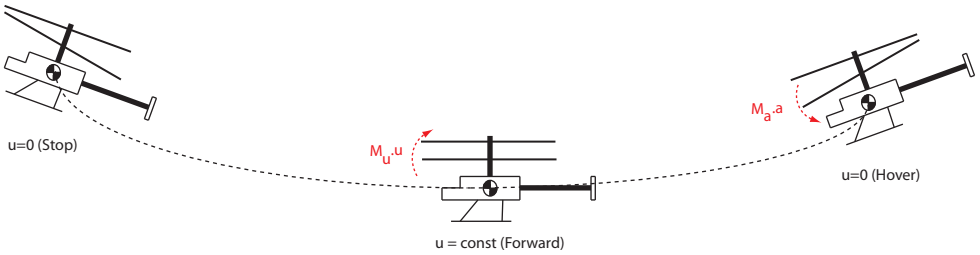
- A_u and B_v

  The rotor flapping derivatives A_u amd B_v describe the effect of body translational speed on the longitudinal and lateral flapping.

  The effect of derivative A_u is associated with an effect called *speed stability* [105] which is created due to increase in relative speed of the advancing blade and decrease in the relative speed of retreating blade in forward flight. As a result, the counter-clockwise rotating blade tends to flap up on the nose producing a nose up pitching moment with respect to helicopter center of gravity which decreases the helicopter's speed. This behavior is characterized by *Phugoid* mode which is illustrated in Figure 3.14-b. As the pilot applies longitudinal cyclic pitch command to the helicopter at hover, the created longitudinal flapping produces a negative pitching moment (due to derivative M_a) which causes the helicopter to pitch down and gain acceleration. However, in forward flight, the derivative A_u captures a positive pitching moment which slows down the helicopter to full stop.



(a) Experiment conducted to find the period of the Phugoid mode.



(b) The helicopter starts from hover to reach to a relatively constant velocity, however due to effect of A_u derivative, it is slowed down to a full stop

Figure 3.14: Estimating the period of the Phugoid mode.

The phugoid mode can be characterized by analyzing the following force and moment equations

$$\begin{aligned} \dot{u} &= X_u u + X_\theta \theta \\ \dot{\theta} &= q \\ \dot{q} &= M_a a + M_c c \end{aligned} \tag{3.33}$$

In addition, due to the long period of this mode compared to the rotor-fuselage mode, we can ignore the flapping transients and consider the steady-state value for the flapping motion, i.e.,

$$\begin{aligned} a &= -\tau q + \tau A_u u \\ c &= \tau^{'} C_q q \end{aligned} \tag{3.34}$$

Combining equation sets (3.33) and (3.34), we obtain the following system of equations

$$\frac{d}{dt}\begin{bmatrix} u \\ \theta \\ q \end{bmatrix} = \begin{bmatrix} X_u & X_\theta & 0 \\ 0 & 0 & 1 \\ M_a A_u \tau & 0 & -M_a \tau - M_c C_q \tau^{'} \end{bmatrix} \begin{bmatrix} u \\ \theta \\ q \end{bmatrix} \tag{3.35}$$

The approximate second-order characteristic equation for the low frequency oscillation becomes [111, 13]

$$\lambda^2 - \left(X_u - \frac{X_\theta A_u \tau}{(M_a \tau + M_c C_q \tau')^2} \right) \lambda - \frac{X_\theta M_u}{(M_a \tau + M_c C_q \tau'} = 0 \qquad (3.36)$$

We conducted an experiment to fly the helicopter at constant velocity by starting from hover in order to measure the period of the phugoid mode. Figure 3.14-a shows snapshots of the experiment. The period of the Phugoid mode obtained from this experiment was $T_{ph} = 4\,sec$. Given the period of phugoid mode, the characteristic equation (3.36) predicts the value of 3.5 for A_u. Finally, based on the symmetry we can conclude that $A_u = -B_v = 3.5$.

In addition, the effect of derivative B_u is called *coning effect* or transverse flow effect [105]. Rotor coning causes the vertical component of forward speed to create upwash on the blade over the nose and downwash on the blade over the tail. Thus, due to this effect, a counter-clockwise rotating blade flaps up on the left side in translational flight. It can be shown that the derivative B_u is proportional to the coning angle and since the coning angle is very small in Blade-CX2 helicopter we ignore this derivative. This effect is usually also most noticeable in fast forward flight. We also found that the identified values for off-axis derivatives (A_v,B_u) very small and have high values in their respective Cramer-Rao bounds. Based on this analysis it is reasonable to drop these derivatives from the parametric model.

The identified values for on-axis speed derivatives came out to be $A_u = 2.29$ and $B_v = -2.14$, which is comparable with the value we obtained based on the Phugoid mode analysis.

Input Derivatives

- A_{lon}, A_{lat}, B_{lon} and B_{lat}
  In order to study the effect of lateral and longitudinal cyclic pitch on the flapping, one can obtain quasi steady solutions (first harmonic) for the blade lateral and longitudinal flapping from the blade flapping differential equation

as follows [13]

$$\begin{aligned} a &= \frac{1}{1+{S_\beta}^2}(S_\beta\theta_{lat}-\theta_{lon}) \\ b &= \frac{1}{1+{S_\beta}^2}(S_\beta\theta_{lon}+\theta_{lat}) \end{aligned}$$

where θ_{lon} and θ_{lat} are the longitudinal and lateral cyclic input, respectively. S_β is the blade stiffness defined as follows

$$S_\beta \triangleq \frac{8({\lambda_\beta}^2-1)}{\gamma} = 0.268 \tag{3.37}$$

and λ_β is the flapping frequency ratio which is defined to be the ratio of the blade flapping frequency to the rotor rotational speed and is given by

$$\lambda_\beta \triangleq 1+\frac{k_\beta}{I_\beta\Omega^2} = 1.042 \tag{3.38}$$

The flapping derivatives with respect to cyclic inputs can then be obtained by taking partial derivatives as follows

$$\begin{aligned} \frac{\partial a}{\partial\theta_{lon}} &= -\frac{\partial b}{\partial\theta_{lat}} = \frac{-1}{1+{S_\beta}^2} \\ \frac{\partial a}{\partial\theta_{lat}} &= \frac{\partial b}{\partial\theta_{lon}} = \frac{S_\beta}{1+{S_\beta}^2} \end{aligned} \tag{3.39}$$

which shows that the cross coupling is more significant when the blade stiffness is higher. The flapping derivatives with respect to the pilot control input can be computed by multiplying the gearing ratio from the servo deflection to the blade cyclic pitch $G_{\delta\theta}$, i.e.,

$$\begin{aligned} \frac{\partial a}{\partial\delta_{lon}} &= G_{\delta\theta}\frac{\partial a}{\partial\theta_{lon}}, \quad \frac{\partial b}{\partial\delta_{lat}} = G_{\delta\theta}\frac{\partial b}{\partial\theta_{lat}} \\ \frac{\partial a}{\partial\delta_{lat}} &= G_{\delta\theta}\frac{\partial a}{\partial\theta_{lat}}, \quad \frac{\partial b}{\partial\delta_{lon}} = G_{\delta\theta}\frac{\partial b}{\partial\theta_{lon}} \end{aligned} \tag{3.40}$$

Now, according to our definition, the flapping derivatives are normalized by the flapping time-constant. In addition, the manufacturer has rotated the swash plate of Blade-CX2 helicopter the amount of $\psi_s = 45$ degrees to reduce the effect of coupling discussed in Section 3.2.1. Hence, by applying

the rotation matrix, the flapping derivative can be computed via

$$\begin{bmatrix} A_{lon} & A_{lat} \\ B_{lon} & B_{lat} \end{bmatrix} = \begin{bmatrix} \frac{1}{\tau}\frac{\partial a}{\partial \delta_{lon}} & \frac{1}{\tau}\frac{\partial a}{\partial \delta_{lat}} \\ \frac{1}{\tau}\frac{\partial b}{\partial \delta_{lon}} & \frac{1}{\tau}\frac{\partial b}{\partial \delta_{lat}} \end{bmatrix} \begin{bmatrix} cos(\psi_s) & sin(\psi_s) \\ -sin(\psi_s) & cos(\psi_s) \end{bmatrix} = \begin{bmatrix} -4.29 & -2.48 \\ -2.48 & 4.297 \end{bmatrix} \tag{3.41}$$

This is comparable with the identified value of $A_{lon} = -4.157$, $B_{lon} = -2.232$, $A_{lat} = -2.908$ and $B_{lat} = 4.900$.

Table 3.3: Comparison between the value of identified parameters based on system identification technique versus the value obtained based on the physics-based analysis.

Parameter	Sys. ID approach	Physics-based approach
τ	0.0276	0.0226
$\tau^{'}$	0.56	1.0276
X_θ	–	-9.810
Y_ϕ	–	9.810
X_u	-0.446	-0.4
Y_v	-0.604	-0.5
A_u	2.289	3.5
B_v	-2.141	-3.5
M_a	189.600	185.3
M_c	154.500	260.7
L_b	265.400	300.1
L_d	186.200	422.2
A_{lon}	-4.157	-4.29
A_{lat}	-2.908	-2.48
B_{lon}	-2.232	-2.48
B_{lat}	4.900	4.29

Finally, to sum up this section, we show the comparison between the identified values of the derivatives and those predicted based on our first principles analysis in this section in Table 3.3. The relatively close results based on these two different approaches shows that the physics-based approach is a powerful method for predicting the derivatives of small-scale rotorcraft if done properly. It also confirms that the model structure is physically meaningful.

Chapter 4

Controller Design and Closed-loop Performance Analysis

The first part of this section deals with modeling the uncertainties in the Blade-CX2 helicopter and follows with the design of a velocity tracking control system based on a nested attitude-velocity architecture. The μ analysis is applied to characterize the effects of the uncertainties on the robustness and performance. This system is then embedded within a path following architecture that regulates the cross track position error as shown in Figure 1.5. In the second part of the section, we introduce methods to evaluate the control system performance within the path-following architecture. We first provide experimental evaluation based on traditional time responses. Next, we extend the sensitivity and complementary sensitivity concepts to nonlinear dynamics. The closed-loop system that combines the vehicle dynamics with the control augmentation is described as a generic nonlinear closed-loop system. Doing so allows to apply the evaluation technique to other forms of control law and types of rotorcraft platforms. In addition, being able to *experimentally* characterize the performance of the closed-loop system can also arise in situations where no mathematical models of the vehicle and control system are available, e.g., these models may be restricted, inaccurate, or too cumbersome to be identified. The procedure used to experimentally characterize the closed-loop system performance can be used without having to separately identify each component.

4.1 Modeling Uncertainty

System identification techniques normally provide an estimate of the system unknown parameters along with the uncertainty in the identified parameters in the form of error covariance or error percent. In the CIFER toolset [106] used for system identification, uncertainty percent is based on the Cramer-Rao bound. The Cramer-Rao bound is determined from the diagonal element of the inverse Hessian matrix associated with the cost function used for fitting the parametric model to the experimental frequency responses. The Cramer-Rao bound provides a lower bound on the identified parameter's standard deviation. As stated in [21], a factor of 2 is needed to find a reasonable estimate of standard deviation:

$$CR_i = 2\sqrt{(H^{-1})_{ii}} \approx \sigma_i \tag{4.1}$$

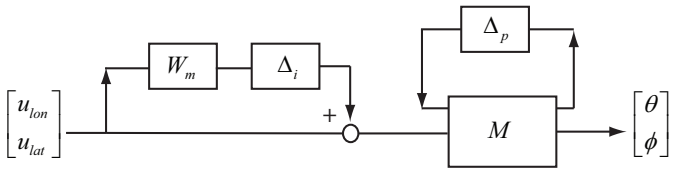
CIFER's Cramer-Rao bound is used as an uncertainty measure for the identified stability derivative (shown in Table 3.1). This parametric uncertainty is modeled using Linear Fractional Transformation (LFT). Consider a nominal system given by M, the uncertainty in that system is described by the Δ. LFT gives the input-output of the model with its uncertainty as

$$\begin{aligned} y &= F_u(M, \Delta) && (4.2)\\ &= (M_{22} + M_{21}\Delta_p(I - M_{11}\Delta_p)^{-1}M_{12})u && (4.3) \end{aligned}$$

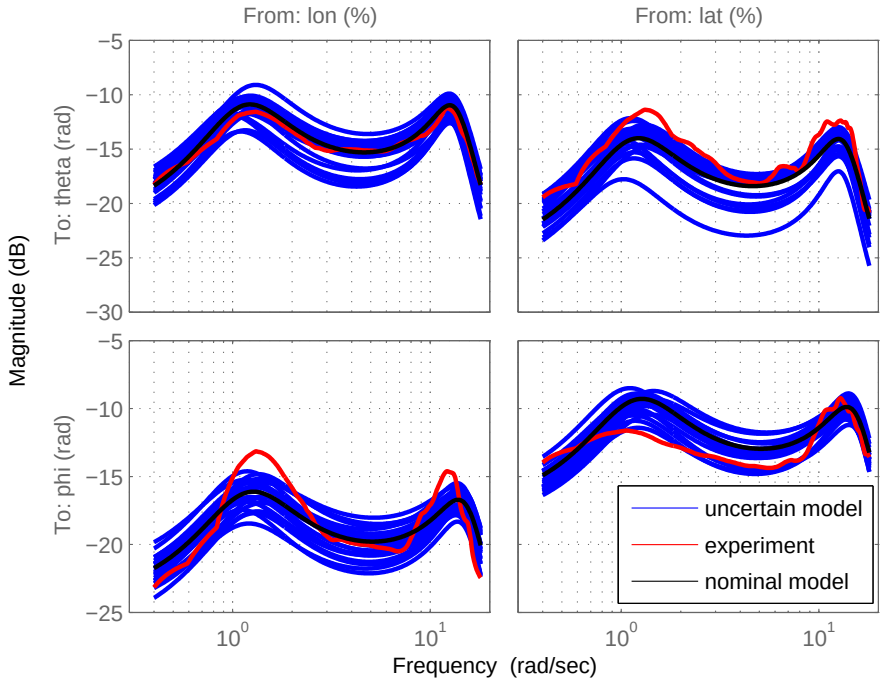
In addition, we use input multiplicative uncertainty to capture the actuator uncertainty as well as uncertainty in cross coupling between the lateral and longitudinal inputs. We found that accounting for this effect is critical since the actuators and linkages wear out and the mechanical design of the swash-plate mechanisms is not very tight in these low-cost models. Miniature helicopters also tend to get more abuse during manipulation, hard landings and occasional collisions. The LTI dynamic multiplicative uncertainty is given by

$$G_\Delta(j\omega) = G(j\omega)[I + W_m(j\omega)\Delta_i(j\omega)],\ |\Delta_i(j\omega)| \leq 1 \tag{4.4}$$

where ω is the frequency, $G(j\omega)$ is the nominal system and $W_m(j\omega)$ is the relative weighting function.



(a) uncertainty model.



(b) uncertain system frequency response.

Figure 4.1: Bode magnitude plot of the helicopter attitude dynamics (with inputs $[u_{lon}, u_{lat}]$ and outputs $[\theta, \phi]$). The responses include the experimental frequency responses (red), the uncertain model (blue), and the response associated with the nominal identified model (black).

Figure 4.1 illustrates the bode magnitude plot of the helicopter pitch and roll response to cyclic inputs. The plot includes the uncertain model along with the experimental frequency responses data and the nominal identified model. The uncertain plant shown are generated using 20 random samples of the original uncertain plant. The results show that the experimental frequency responses are within the frequency response envelope resulting from the parametric and structured uncertainty. We use these frequency responses to analyze the robust stability and robust performance of the controller.

4.2 Control Architecture

This section describes the control augmentation for the Blade-CX2 helicopter that are necessary for the path following controller. The details are specific to the helicopter but the overall input-output response characteristics are common to other helicopter and rotorcraft. In helicopters, the cyclic blade pitch is used to control roll and pitch attitude (via the moments generated by the longitudinal and lateral tip-path-plane (TPP) angles). The rotor thrust is approximately perpendicular to the TPP. Therefore, a change in attitude (superposed with TPP) acts as a type of thrust vectoring. The difference in time constants associated with the rotor and fuselage rotation and the fuselage translation are typically sufficiently large to allow the use of the attitude angle as a pseudo-control input for the translational motion.

A nested control architecture is common in helicopters and other rotorcraft. An attitude inner-loop provides fast and accurate pitch and roll dynamics. A velocity outer-loop uses the attitude angles as pseudo control inputs to control the propulsive x and y forces. We use classical loop-shaping technique to shape the inner-loop frequency responses. For the outer velocity loop we use dynamic inversion technique with desired dynamics using the reduce-order velocity-attitude dynamics of the helicopter. Figure 4.2 shows the block diagram of the nested velocity-attitude control loop architecture. The attitude and velocity controller as well as robustness analysis are explained in details in Subsections 4.2.1 and 4.2.2, respectively. Subsection 4.2.3 describes how the velocity control system is incorporated within the path following to enable desired tracking performance.

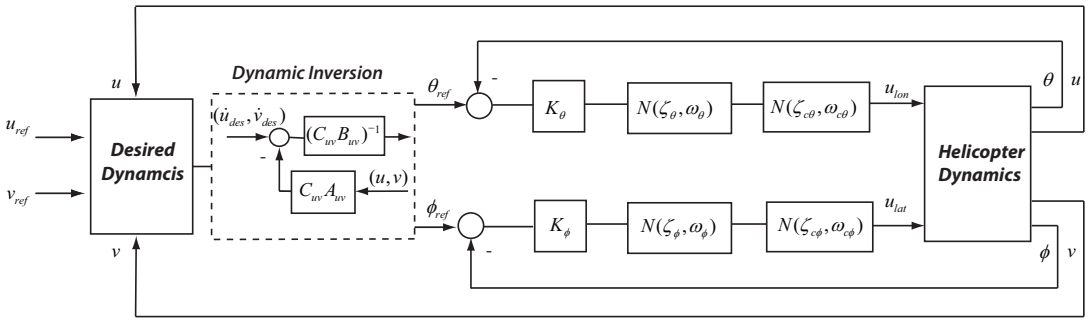


Figure 4.2: Nested velocity control architecture.

4.2.1 Attitude Controller

The inner-loop is an attitude controller which relies on pitch and roll feedback measurements. Angular rate damping is provided mechanically by the helicopter's top rotor and stabilizer-bar. Figure 4.3 shows the experimental frequency responses for attitude dynamics. As can be seen, for the Blade-CX2 helicopter, the Phugoid mode is lightly damping in comparison to other miniature helicopter (e.g. the R-50 [35]). The same observation can be made about the coupled Fueselage-Rotor-Stabilizerbar (FRS) mode. Finally, the pseudo rate feedback provided by the stabilizer-bar generates enough phase lead that a PD or lead type compensator is not required.

For the nested velocity control architecture, it is important to achieve the highest possible bandwidth BW for the inner, attitude control loop. The BW of the velocity loop should not be higher than about half the attitude loop BW to ensure sufficient time-scale separation between the two loops. One way to speed up the attitude control system is to avoid including integral action in that loop. Ultimately, however, the BW is limited by the natural frequency of the FRS mode as well as the time-delay. According to [112], it is recommended to limit the crossover frequency ω_c of the system to

$$\omega_c < 1/\theta_d \tag{4.5}$$

where θ_d is the time delay. Given the approximate $\theta_d = 60\,msec$ delay in our setup, the bandwidth is limited to about $16\,rad/sec$.

The lightly damped FRS mode in the attitude dynamics can be compensated by a second order notch filter matching the damping and natural frequency of the FRS mode in each axis

$$N(\zeta, \omega) = \frac{s^2 + 2\zeta\omega s + \omega^2}{s^2 + 2\omega s + \omega^2} \tag{4.6}$$

The parameters ω and ζ were chosen as the natural frequency and damping ratio of the FRS mode in each axis

$$\begin{aligned} \omega_\theta &= 12.35\,rad/sec, \quad \zeta_\theta = 0.2 \\ \omega_\phi &= 12.86\,rad/sec, \quad \zeta_\phi = 0.2 \end{aligned} \tag{4.7}$$

As can be seen from Figure 4.3, for the Blade-CX2 helicopter, the off-axis FRS mode has a frequency very close to the on-axis FRS mode. Therefore, the above notch filters are capable of eliminating the coupling. However, some off-axis effects are observed around the frequency of $8\,rad/sec$. To remove the effect of this coupling we use off-axis notch filter with the parameters

$$(\omega_{c\theta}, \zeta_{c\theta}) = (\omega_{c\phi}, \zeta_{c\phi}) = (8, 0.4) \tag{4.8}$$

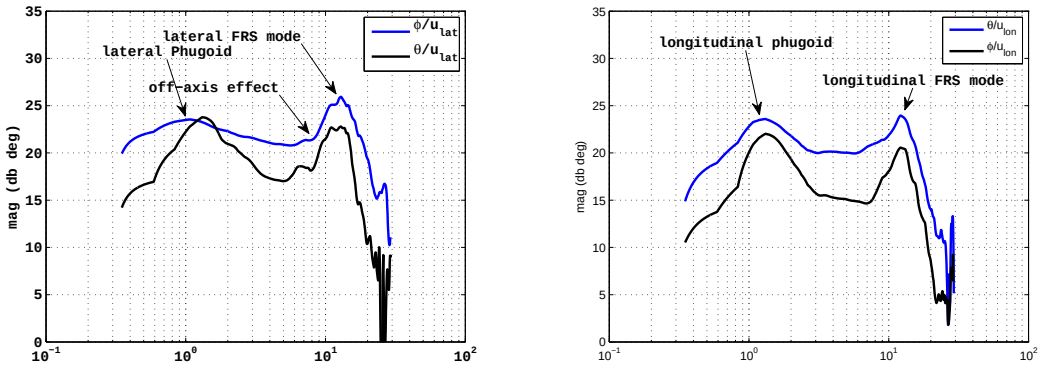
The gains K_θ, K_ϕ in block diagram depicted in Figure 4.2 are used to adjust the BW. The maximum BW is determined based on the minimum amount of phase-margin (PM) and gain-margin (GM) needed to achieve sufficient robustness. With the following set of gains

$$\begin{aligned} K_\theta &= -0.1180\ \%/deg \\ K_\phi &= 0.106\ \%/deg \end{aligned} \tag{4.9}$$

the system achieves a bandwidth of about $2.5\,rad/sec$ for the attitude loop with $100\,deg$ of PM and/or $12\,db$ GM.

Figure 4.4 shows the resulting loop-shapes along with the controller and open-loop frequency responses. Notice from the loop-gains that the bandwidth could still be increased using larger feedback gains (K_θ, K_ϕ). The available PM comes from the pseudo rate feedback effect of the stabilizer-bar. However, this would be detrimental for the GM as the GM of less then $10\,db$ would compromise the robustness.

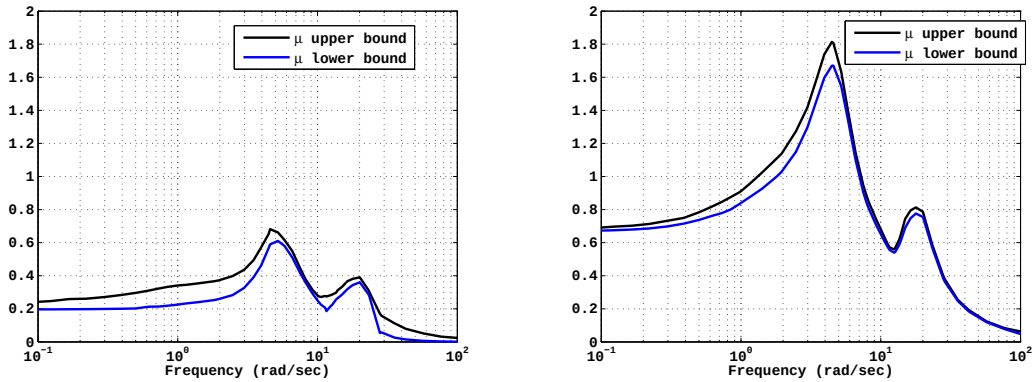
To evaluate the robust stability and robust performance of the attitude control system with respect to the modeling uncertainties shown in Figure 4.1, which were discussed in Section 4.1, we use μ analysis [113]. The result of μ analysis is shown in Figure 4.5. In regard to robust stability, the μ analysis reveals that the attitude controller has μ value less than one across all frequencies. This means that the inner-loop is robustly stable with respect to the given parametric and structured uncertainties. On the other hand, in regard to robust performance, the μ analysis shows a μ value greater than unity. The velocity controller therefore will not achieve robust performance.



(a) Attitude response to lateral input.

(b) Attitude response to longitudinal input.

Figure 4.3: Experimental attitude frequency responses highlighting the lightly damped lateral and longitudinal Fueselage-Rotor-Stabilizerbar (FRS) modes as well as the lightly damped Phugoid mode.



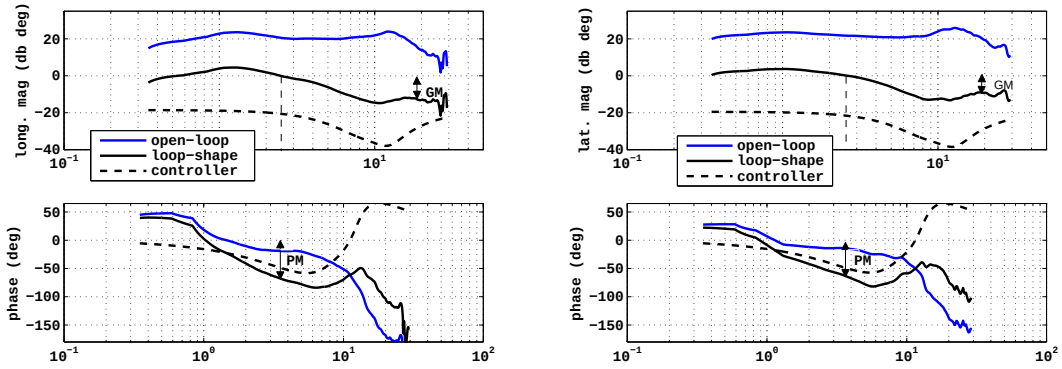
(a) μ for robust stability.

(b) μ for robust performance.

Figure 4.5: Result of the μ analysis for the attitude controller.

4.2.2 Velocity Controller

The velocity control system follows a nested loop design where the pseudo-control (the attitude angles) are determined from a dynamic inversion. The nested control architecture is based on the time-scale separation between the high frequency and low frequency modes. The low-frequency dynamics are the fuselage translational dynamics with states $[u, v, \theta, \phi]$. The high-frequency ones correspond to the attitude fuselage angular rates coupled with the rotor tip-path-plane dynamics with states $[p, q, a, b, c, d]$. According to the overall dynamic model given in



(a) Pitch angle loop-shape.

(b) Roll angle loop-shape.

Figure 4.4: Loop shapes obtained using a classical loop-shaping procedure. Notice how the notch filter eliminates the FRS coupling and ensures that a sufficient phase margin is achieved.

equation (3.13), the dynamics governing the low frequency states are

$$\dot{u} = X_u u + X_\theta \theta \tag{4.10}$$

$$\dot{v} = Y_v v + Y_\theta \theta \tag{4.11}$$

This system can be written in the following state-space form

$$\frac{d}{dt}\begin{bmatrix} u \\ v \end{bmatrix} = A_{uv}\begin{bmatrix} u \\ v \end{bmatrix} + B_{uv}\begin{bmatrix} \theta \\ \phi \end{bmatrix} \tag{4.12}$$

where here $[u,\, v]$ are the the Controlled Variable (CV) [114]. The basic dynamics inversion equation for computing attitude commands is

$$\begin{bmatrix} \theta \\ \phi \end{bmatrix}_{ref} = (C_{uv}B_{uv})^{-1}\left(\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix}_{des} - C_{uv}A_{uv}\begin{bmatrix} u \\ v \end{bmatrix}\right) \tag{4.13}$$

The 'desired dynamics' are determined using feedback from the CV as well as the reference velocity. In general these are composed of a PI-term combined with a

feed-forward term [38]

$$\frac{dCV_{des}}{dt} = K_f \frac{dCV_{cmd}}{dt} + K_b \left(f_c \, CV_{cmd} - CV + z \right) \tag{4.14}$$

$$\frac{dz}{dt} = f_i K_b (CV_{cmd} - CV) \tag{4.15}$$

The proportional gain K_b sets the bandwidth, the integral gain f_i is used for desensitization and disturbance rejection, and the command gain f_c is use to achieve the desired command response. In our design we did not use the feedforward gain K_f. We also considered a pre-filter with time constant τ to smooth the command signal. Figure 4.6 shows the blockdiagram for our 'desired dynamics'. The numerical value of the controller gains in our implementation are

$$f_c = 1, \; f_i = 0.01, \; K_b = 2 \tag{4.16}$$

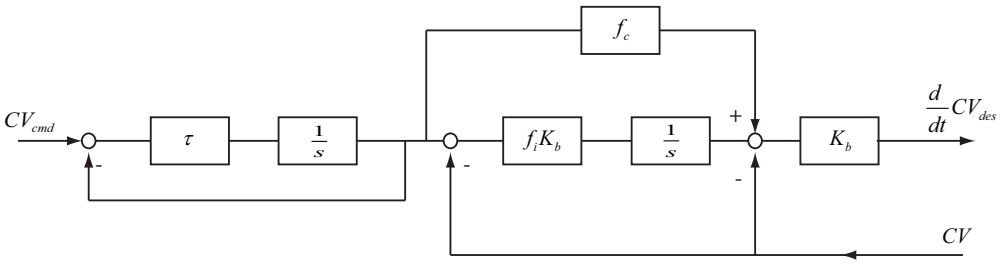
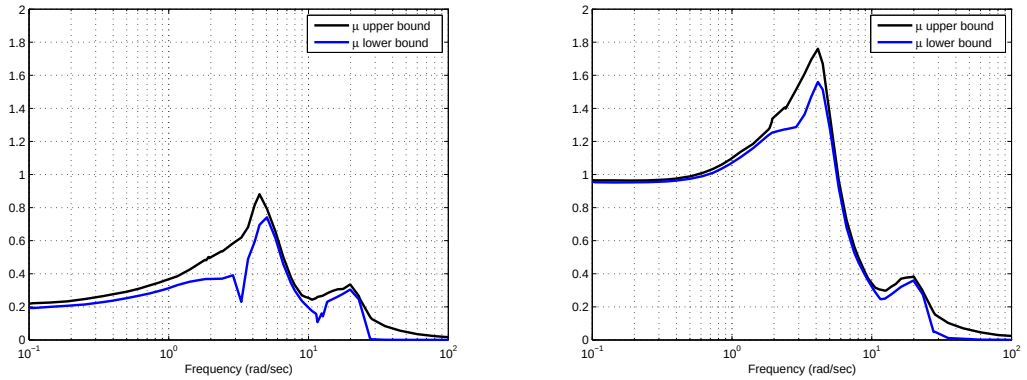


Figure 4.6: Desired dynamics for Dynamic Inversion controller.

The μ analysis for robust stability against the parametric and structured uncertainty indicates that the velocity controller achieves a μ value less than one over the entire frequency range (see Figure 4.7). The robust performance μ analysis, however, shows that the velocity controller results in a μ value greater than one. The velocity controller therefore will not achieve robust performance. Although a higher level of performance might be expected for the Blade-CX2 helicopter, the achievable performance is limited by several factors including the time-delay (around 60 msec) associated with the experimental setup; the relatively low bandwidth actuators (around 12 rad/sec); and the powerful tuning of the stabilizer-bar. All of these factors contribute in making it difficult to achieve robust performance. It is important to keep in mind that helicopters like the Blade-CX2 are hobby helicopter that were developed primarily for indoor flight and stable enough to allow

operation by unexperienced pilots.



(a) μ upper-bound for robust stability.

(b) μ upper-bound for robust performance.

Figure 4.7: Result of the μ analysis of the velocity control system.

4.2.3 Path Following Controller Architecture

As explained in Section 1.5.3 the receding horizon trajectory optimization provides a reference path over finite horizon. The reference path is specified by a set of waypoint locations and velocity vectors at each waypoint $(p_i, v_i),\ i = 1, ..., H$ in inertial frame. This reference path corresponds to the trajectory of a rigid-body system, therefore the linear velocity tracking system will not provide adequate performance. The nonlinear effects that are associated with the operating point need to be accounted for. For the analysis of the path following problem, we consider a simplified mass-point model of the helicopter dynamics (see Appendices A.1 and A.2). This model abstracts out the details of the rotor dynamics. This simplification is possible thanks to the control augmentation provided by the attitude and velocity control loops.

The primary effect occurring through the rigid-body dynamics is the normal acceleration arising in the presence of path curvature $a_n = V^2/\rho$. The analysis given in the Appendices A.1 and A.2 show that the radial and tangential (so-called cross-track and track) error can be adequately regulated by two decoupled PD controllers. Figure 4.8 shows the path following tracking architecture derived from the mass-point analysis. The error between the current helicopter position and velocity and the reference waypoints and velocities is parameterized in terms of cross-track error. This error and the reference velocity are transformed into

body coordinates. The body-frame velocity reference is applied to the velocity controller. The normal error vector, which is in the direction of centrifugal acceleration, is passed through a PD element and applied as feedforward command signal for the attitude controller. The feedforward guarantees that an adequate amount of thrust is generated in the direction normal to the path to compensate for the effect of centrifugal acceleration arising in the presence of path curvature.

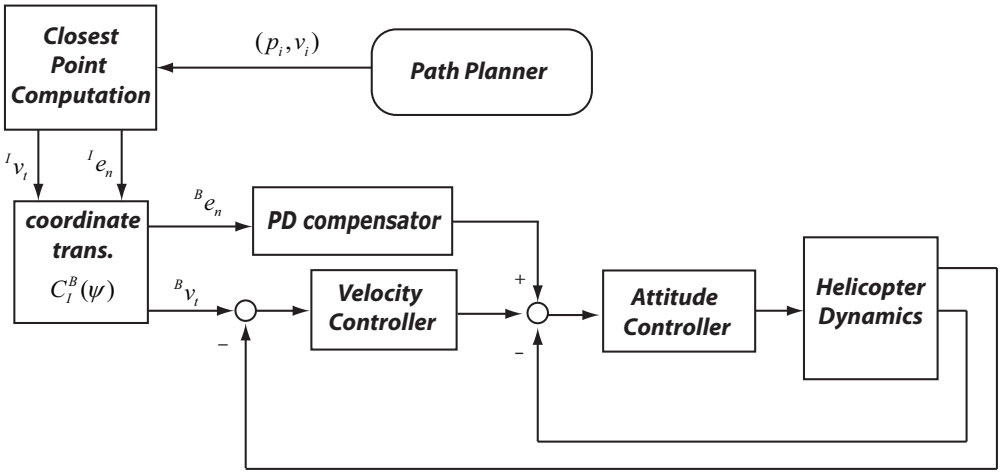


Figure 4.8: Path following controller architecture.

The remaining of this section describes the *Closest-Point Algorithm* (based on [49]). A spline description of the path is used to determine the tangent and normal to the path and helps deal with possible discontinuities. A continuous spline description of the path is computed by fitting a cubic spline to the sequence of waypoints. We use a parametric representation of the cubic spline in terms of the path arc length s

$$p(s) = as^3 + bs^2 + cs + d, \tag{4.17}$$

where s is obtained by linear approximation of the path length between two subsequent waypoints p_i and p_{i-1}

$$s_i = s_{i-1} + ||p_i(s) - p_{i-1}(s)|| \tag{4.18}$$

The vectors a, b, c and d are then computed via a cubic spline interpolation algorithm (see, e.g., [115]). For each value of s the tangent $t(s)$ and normal $n(s)$

vectors are obtained as follows

$$\begin{aligned} t(s) &= 3as^2 + 2bs + c \\ n(s) &= t \times q \times t/|t^4| \\ q(s) &= 6as + 2b \end{aligned} \tag{4.19}$$

Given the helicopter position in inertial frame, the closest reference point s^* on the path is obtained knowing that the error vector from the vehicle position to the reference point e has to be normal to the tangent vector

$$e_t(s) = e(s) \cdot \frac{t(s)}{||t(s)||} \tag{4.20}$$

where $e(s) = p(s) - [x,\, y]^T$ (see Figure 4.9). To find the reference point that satisfies the this condition, the following equation can be used [49]

$$s_n = s_{n-1} + \frac{e_t(s_{n-1})}{||t(s_{n-1})||} \tag{4.21}$$

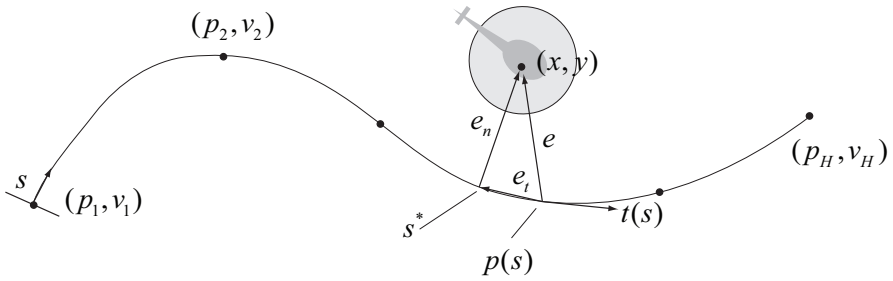


Figure 4.9: Illustration of closest point computation.

The closest reference point $s^\star$ is then used to compute the cross-track position error vector ${}^I e_n(s^\star)$ and the desired tangent velocity vector ${}^I v_t(s^\star)$ (the superscript I stands for inertial frame). These vectors are then transformed into the helicopter body frame B according to

$$\begin{bmatrix} {}^I e_n(s^\star) \\ {}^I v_t(s^\star) \end{bmatrix} = C_I^B(\psi) \begin{bmatrix} {}^B e_n(s^\star) \\ {}^B v_t(s^\star) \end{bmatrix} \tag{4.22}$$

where ψ is the body frame heading angle. Finally, these signals are incorporated into the the control system as illustrated in Figure 4.8.

4.3 Experimental Evaluation

Autonomous guidance calls for control performance requirements that are not fully captured by the traditional robust control analysis or the traditional handling qualities metrics [85]. This section provides experimental evaluation of the closed-loop control system as well as the particular requirements of the autonomous guidance system that was descried in Section 1.5.2. The closed-loop vehicle is treated as a generic closed-loop system and all the methods described next can in principle be applied to any closed-loop system formed by a particular rotorcraft and control augmentation. First the traditional evaluation of the attitude and velocity control laws is described and then, we identify the characteristics that are needed for guidance: the absolute maneuvering envelope and the tracking error characteristics of the path following controller.

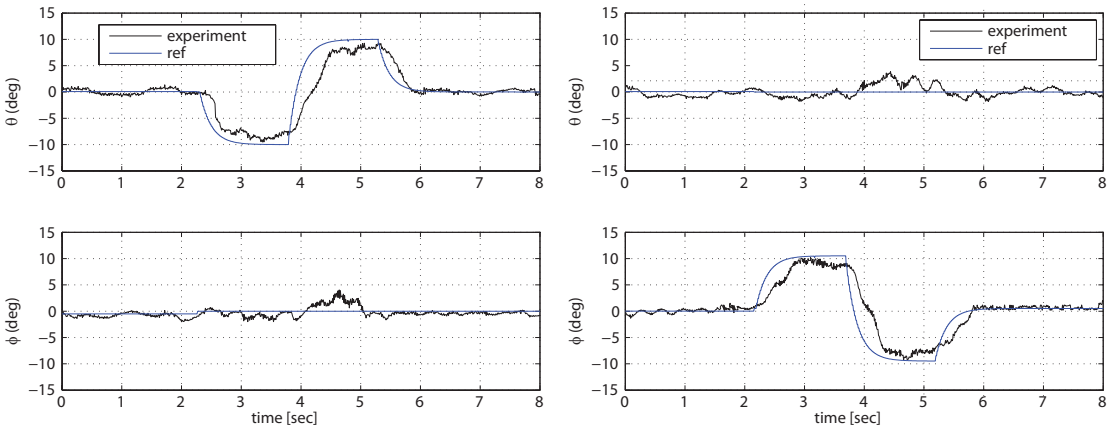
4.3.1 Evaluation of the Velocity and Attitude Controllers

The performance of the velocity and attitude controller is evaluated by applying doublet reference test signals. The controllers are implemented and run at 100 Hz. However, due to limitations of the RC transmitter, the control signals can only be sent to the helicopter as fast as 50 Hz. The controllers were discretized using a used zero-order hold (ZOH) transformation.

The inner-loop attitude controller, was evaluated by applying a doublet reference of 10 deg magnitude and 1.5 sec pulse duration. The tracking performance of the velocity control system was evaluated by applying a doublet velocity command of 1 m/sec magnitude and 2 sec pulse duration. Figure 4.10 shows the tracking performance of the inner-loop and Figure 4.11 shows the tracking performance of the velocity controller.

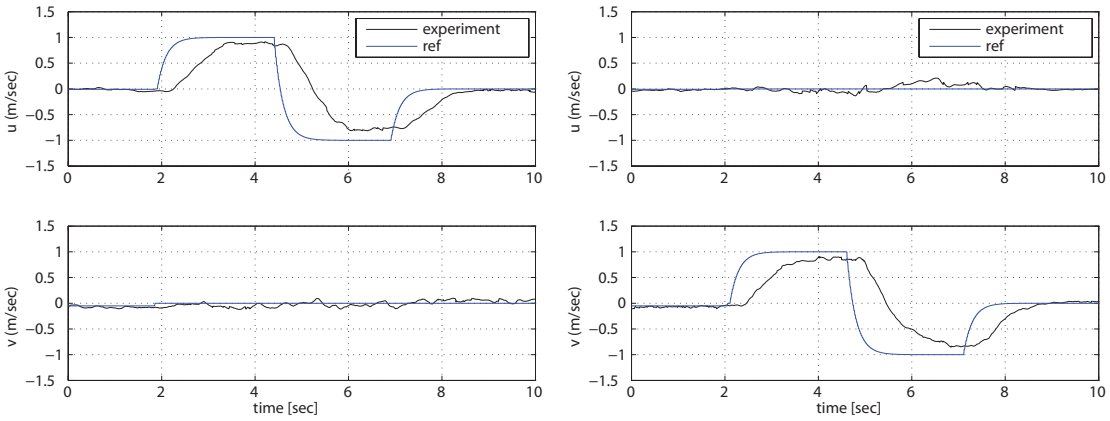
4.3.2 Evaluating Closed-loop Absolute Maneuvering Envelope

Knowing the absolute maneuvering envelope for the closed-loop system is necessary to avoid generating infeasible trajectories. The values of normal acceleration that the helicopter can sustain determines the upper-bound on the maximum curvature. Based on the centrifugal effects, for the same normal acceleration, a higher curvature can be achieved at lower velocity magnitude and vice versa. The absolute limits in normal acceleration depends on the maximum attitude that can



(a) longitudinal doublet. (b) lateral doublet.

Figure 4.10: Flight test results for attitude controller.



(a) longitudinal doublet. (b) lateral doublet.

Figure 4.11: Flight test results for velocity controller.

be held and the thrust. The curvature-velocity plot is a convenient way to depict the maneuvering envelope since it has a direct geometric interpretation [116].

Consider a general helicopter trajectory as shown in Figure 4.12 where v_t and a are total horizontal velocity and acceleration vectors in inertial frame, respectively. Given these two vectors, we can compute the angle ξ between the velocity and acceleration vector as follows

$$\xi = \sin^{-1} \frac{a \times v_t}{||a||.||v_t||} \tag{4.23}$$

Given the angle ξ, the normal acceleration a_n and curvature of the path can be

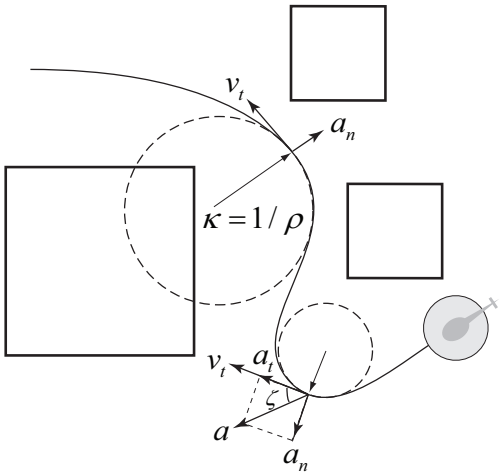


Figure 4.12: Parametrization of a general trajectory amid obstacles.

computed via

$$||a_n|| = ||a|| \sin(\xi) \tag{4.24}$$

$$\kappa = \frac{||a_n||}{||v_t||^2} \tag{4.25}$$

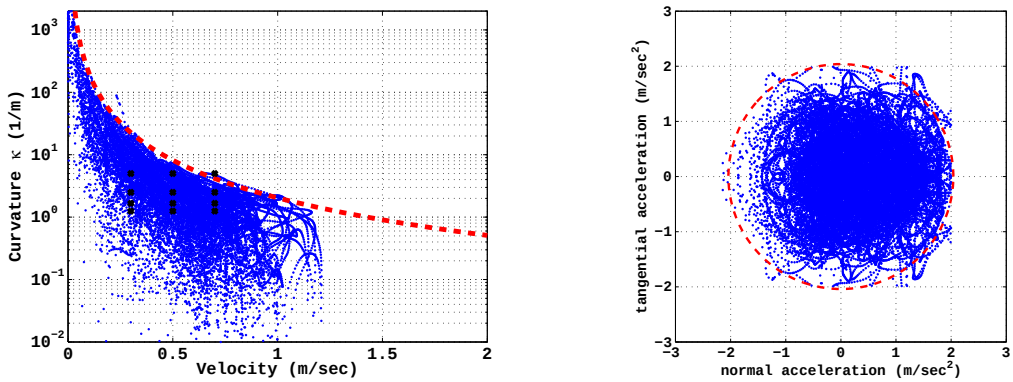
The upper-bound on curvature-velocity curve can be computed from the maximum roll angle ϕ_{max} for the bare airframe and using the assumption that the helicopter stays level altitude. The thrust is given by $T = mg/\cos(\phi_{max})$. The maximum acceleration based on the normal acceleration is given by

$$a_{max} = g \tan(\phi_{max}) \tag{4.26}$$

where g is the gravitational acceleration. Hence, the following relation provides the upper-bound for the curvature-velocity space

$$\kappa = \frac{g \tan(\phi_{max})}{||v_t||^2} \tag{4.27}$$

Figure 4.13-a shows the experimental curvature-velocity characteristic of the Blade-CX2 helicopter under velocity control obtained from equations (4.23) to (4.25). The red line shows the upper-bound (4.27). The black dots show the curvature-velocity coordinate for 12 experiments used for the tracking performance evaluation. In addition, Figure 4.13-b shows the normal and tangential acceleration of the vehicle during the same flight and the corresponding upper bound.



(a) Absolute envelope.

(b) Normal and tangential acceleration.

Figure 4.13: (a) Curvature-velocity plot of Blade-CX2 helicopter in closed-loop: The curvature-velocity points are constrained by the absolute total acceleration limit (red line). The black dots are the the curvature-velocity coordinate for the tracking evaluation experiments. (b) Normal and tangential acceleration characteristic of the Blade-CX2 helicopter in closed-loop.

4.3.3 Performance and Robustness Analysis of the Velocity Controller

Traditional performance and robustness characteristic are based on the sensitivity $|S(j\omega)|$ and complementary sensitivity functions $|T(j\omega)|$. The sensitivity function describes the behavior of the tracking error e as a function of the frequency of the reference r. The complementary sensitivity function describes how a measurement noise n affects the output y. However, if the closed-loop system contains nonlinearities, the frequency responses for $|S(j\omega)|$ and $|T(j\omega)|$ cannot be interpreted in this traditional manner.

The notion of a steady-state response to harmonic excitation is undefined for general nonlinear systems. A general nonlinear system can have multiple coexisting periodic and even chaotic responses to a harmonic excitation. However, for the special class of *convergent* nonlinear systems such a notion is still valid. Convergent systems are nonlinear systems for which any bounded input produces a uniformly bounded globally asymptotically stable solution [117] (which is called a steady-state solution). The convergence property can usually be achieved through a feedback controller.

Suppose that that the helicopter closed-loop velocity control system is uniformly convergent. Then we can define a *generalized sensitivity and complementary sensitivity functions* as follows [118]

$$S(v,\omega) = \frac{||e_{v\omega}||_2}{||r_{v\omega}||_2}, \quad T(v,\omega) = \frac{||y_{v\omega}||_2}{||r_{v\omega}||_2} \tag{4.28}$$

where v is the harmonic excitation magnitude and ω is its frequency. Notice that due to nonlinearity of the system, $S(v,\omega)$ and $T(v,\omega)$ depend not only on the excitation frequency ω but also on the amplitude v. In addition, for MIMO systems the frequency responses also depend on the direction of the input signal. Therefore, the *generalized sensitivity and complementary sensitivity functions* for MIMO systems can be define as follows

$$S(v,\omega) = \max_{r_{v\omega}\neq 0} \frac{||e_{v\omega}||_2}{||r_{v\omega}||_2}, \quad T(v,\omega) = \max_{r_{v\omega}\neq 0} \frac{||y_{v\omega}||_2}{||r_{v\omega}||_2} \tag{4.29}$$

By choosing the velocity reference signal $r_{a\omega}$ for the velocity control system as

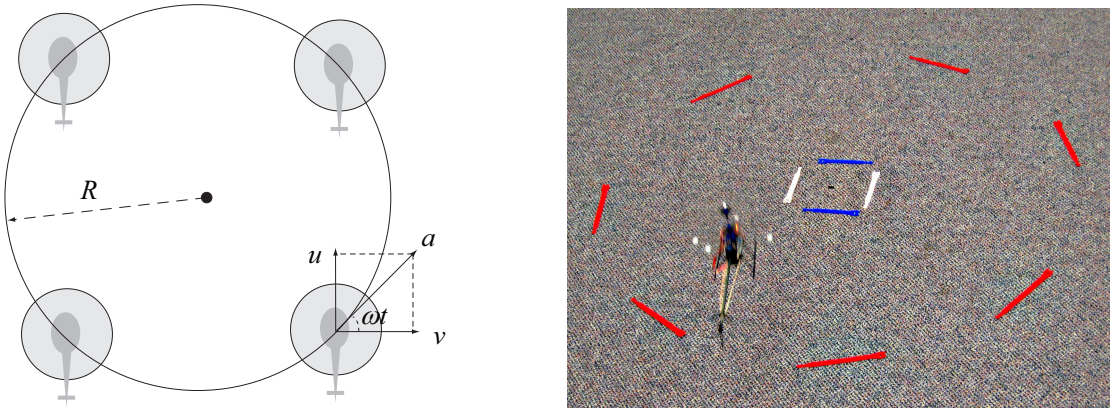
$$r_{a\omega} = \begin{bmatrix} a\,cos(\omega t) \\ a\,sin(\omega t) \end{bmatrix} \tag{4.30}$$

one can find the maximum gain for $S(a,\omega)$ and $T(a,\omega)$ according to equation (4.29) as the direction of input vector is varied for the given input magnitude a and frequency ω.

By varying the frequency and magnitude of the reference command it is possible to evaluate the trajectory tracking performance around a particular equilibrium trajectory. Given the space constraints in our indoor facility it is not possible to evaluate the tracking performance for the full range of operating points. Nevertheless, it is possible to achieve the same effect by following circular trajectories and holding the heading constant. The reference command $r_{v\omega}$ in equation (4.30) corresponds to the circular trajectory with constant heading as shown in Figure 4.14. The radius of the trajectory can be obtained via $R = v/\omega$.

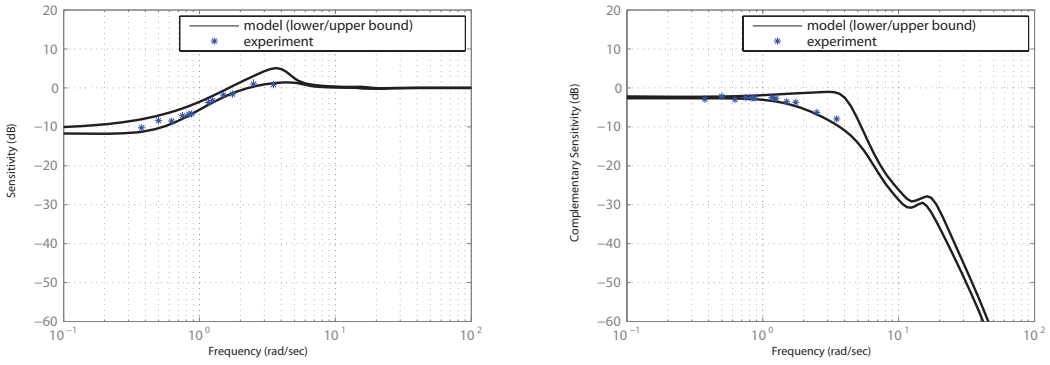
Figure 4.16 shows the experimental tracking performance when applying the velocity reference command of equation (4.30) for different magnitudes and frequencies corresponding to the 12 experimental curvature-velocity points shown in Figure 4.13-a. Note that due to the space limitation in our indoor flight test

facility, we were only able to achieve the frequencies between 0.35 rad/sec to 3.5 rad/sec. As can be seen in Figure 4.16, the executed circular trajectories are drifting over time due to noise and disturbances. Based on these flight tests, we computed the generalized sensitivity and complementary sensitivity function of the overall velocity control system at different input magnitudes and frequencies according to equation (4.29). Figure 4.15 shows the comparison between the nominal linear model sensitivity and complementary sensitivity function and the ones obtained from the experiment.



(a) (b)

Figure 4.14: Circular trajectory with constant heading, which corresponds to applying reference command of equation (4.30) to the velocity control system.



(a) sensitivity function. (b) complementary sensitivity function.

Figure 4.15: Sensitivity and complementary sensitivity function obtained from the linear model as well as experimental data obtained from the circle tests at different velocities and frequencies.

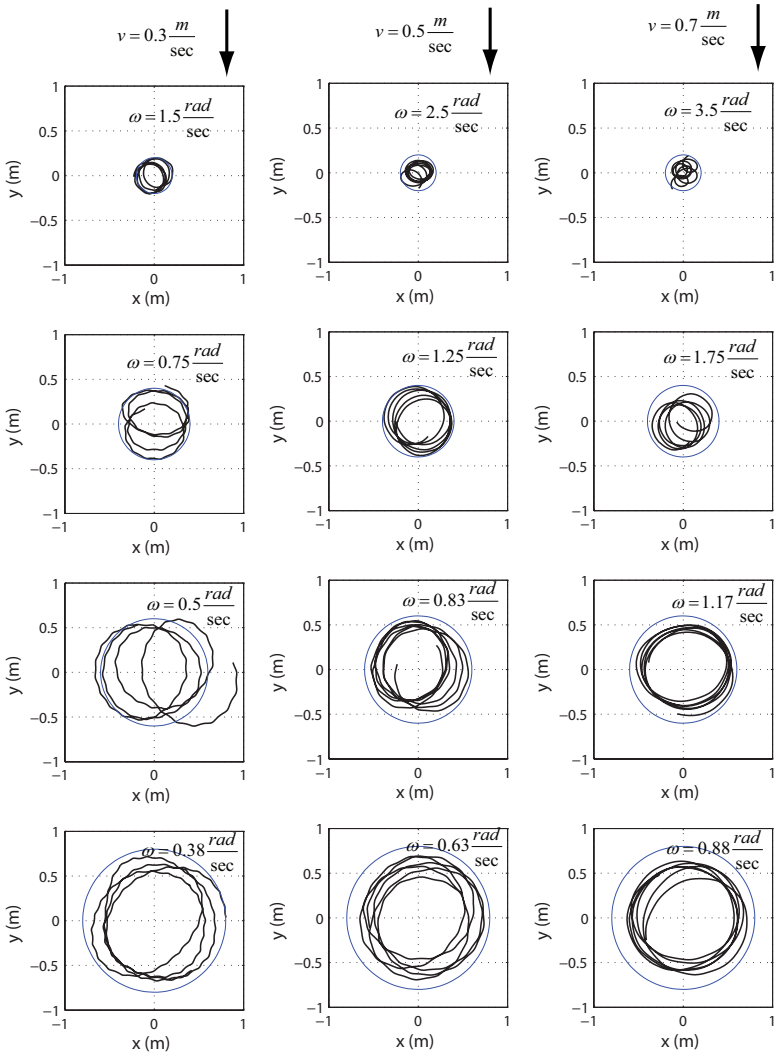


Figure 4.16: Experimental tracking performance based on circular trajectories obtained by applying reference velocity command of equation (4.30) with different magnitude and frequency to the velocity control system.

4.3.4 Performance Evaluation of Path Following System

The flight test results for the same 12 test points with the path following system are depicted in Figure 4.17. Compared to Figure 4.16, where the path following loop is not active, we see no drift in the trajectories. The trajectories are followed with small deviations due to the effects of disturbances. To characterize the tracking performance the flight data is used to generate histograms of the relative tracking error (see Figure 4.18). The mean and variance of the tracking error are modeled by fitting each histogram using Gaussian density functions. These values

are depicted as functions of velocity and normal acceleration in Figure 4.19. Linear interpolation was used to cover the entire velocity and normal acceleration range. From these plots we can clearly see that tracking error is primarily a function of normal acceleration with a small speed effect.

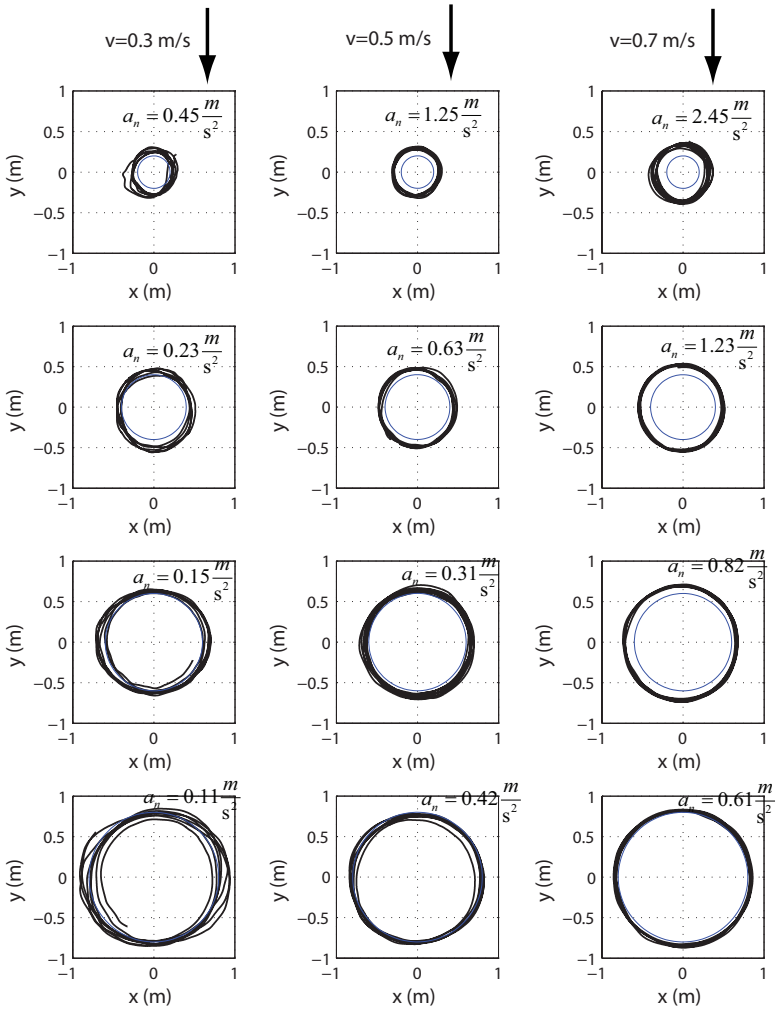


Figure 4.17: Experimental tracking performance of the path following controller at different velocities and turn radius.

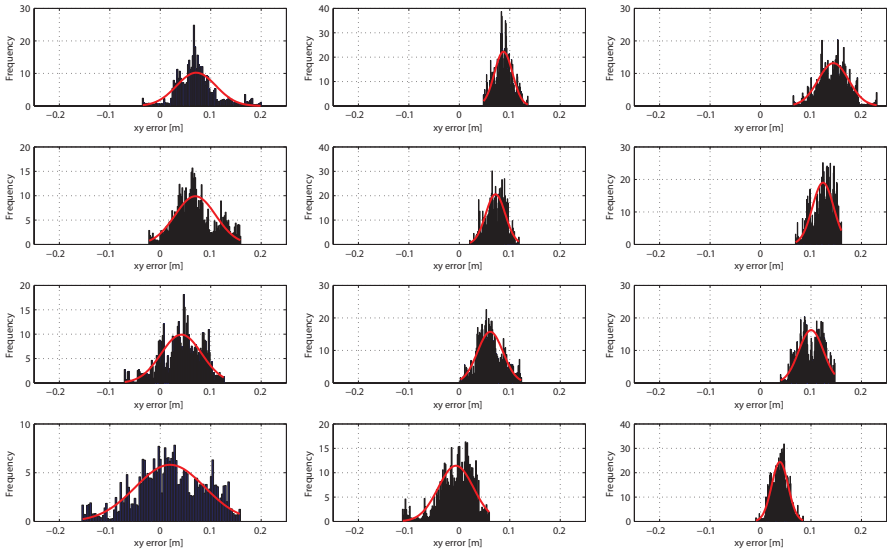
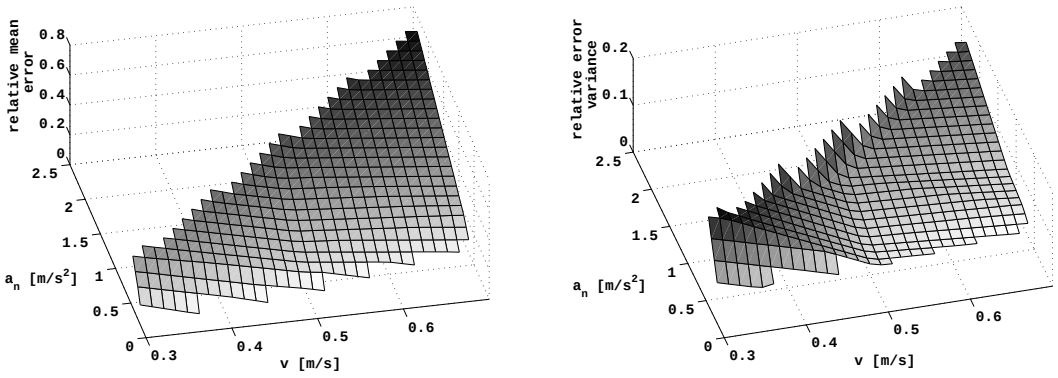


Figure 4.18: Histogram of the relative tracking error along with the fitted normal density function.



(a) Mean. (b) Variance.

Figure 4.19: Statistical properties of the relative tracking error in the circle tracking experiment as a function of tangential velocity and normal acceleration.

4.3.5 Tracking Error Model

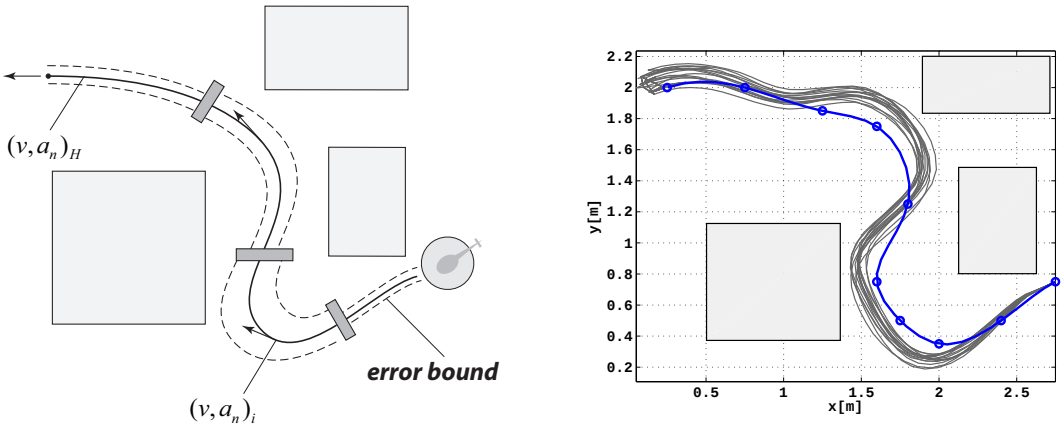
To ensure safe operation when operating amidst obstacles, it is critical to be able to predict the tracking error bound (see Figure 4.20). The goal is to have a tracking error model that can be used in real-time to predict the tracking error bound based on a pre-specified path generated by a path planner. Figure 4.21 shows the block-diagram of error prediction model.

The error model is based on a simplified first-order model of the inner-loop.

Un-modeled effects and modeling uncertainties and disturbances are described by noise parameters η_x and η_y. According to the bock-diagram in Figure 4.21, the differential equation governing the path tracking dynamics are as follows

$$\begin{aligned}
\theta_{ref} &= X_\theta^{-1}\left[\dot{u}_{des} - X_u u\right] \\
\phi_{ref} &= Y_\phi^{-1}\left[\dot{v}_{des} - Y_v v\right] \\
\dot{\theta} &= -1/\tau\,\theta + 1/\tau\,\theta_{ref} + K_{p_x} e_x + K_{d_x}\dot{e}_x + \eta_x \\
\dot{\phi} &= -1/\tau\phi + 1/\tau\,\phi_{ref} + K_{p_y} e_y + K_{d_y}\dot{e}_x + \eta_y \\
\dot{u} &= X_u u + X_\theta \theta \\
\dot{v} &= Y_v v + Y_\phi \phi \\
\dot{x} &= u \\
\dot{y} &= v
\end{aligned} \tag{4.31}$$

where $e = [e_x,\, e_y]$ is the normal tracking error vector which is computed according the *closed-point algorithms* in Section 4.2.3. The noise parameters η_x and η_y are zero-mean normal Gaussian random variables. They are determined by matching the predicted error statistics with those obtained from the experiments. The gains K_{p_x}, K_{d_x}, K_{p_y} and K_{d_y} are the PD controller gains. Figure 4.22 shows a comparison between the histogram of the predicted relative error with the experimental results obtained from the circle experiments.



(a) (b)

Figure 4.20: (a) Illustration of a path in an obstacle-field environment in the presence of tracking errors. (b) 20 Monte-carlos simulations of the tracking error model showing the waypoints specifying the reference path and the spline fit. Each waypoint has also an attributed reference velocity.

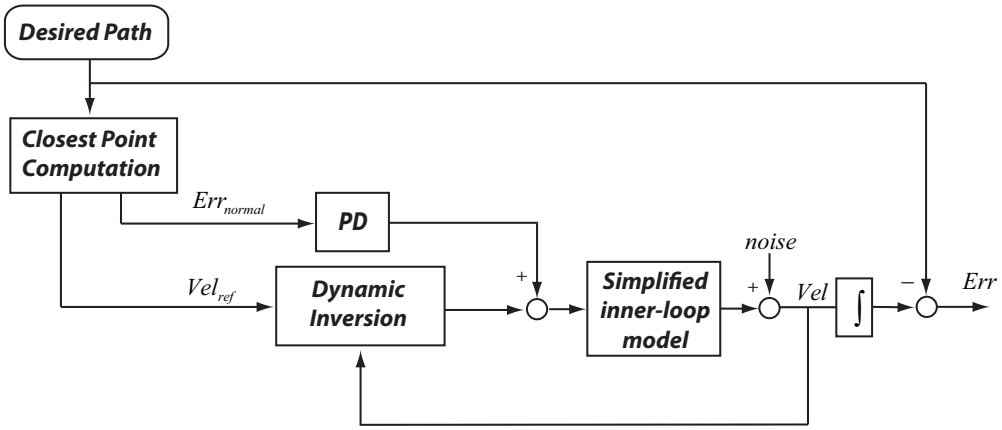


Figure 4.21: Block diagram of the tracking error prediction model.

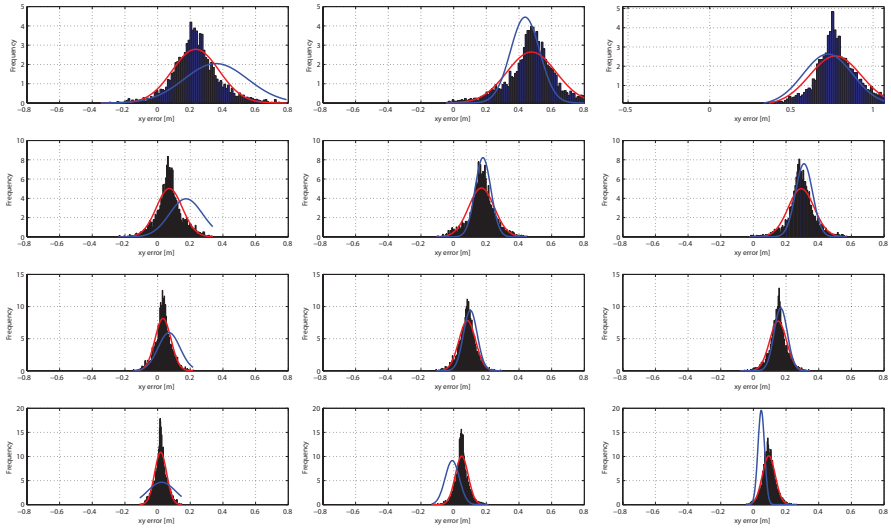


Figure 4.22: Histogram of the predicted relative tracking error along with the experimental statistics (blue).

The differential equation governing the path tracking dynamics can be further simplified. Ignoring the inner-loop transient response, the attitude angles of the helicopter become

$$\begin{aligned} \theta &= \theta_{ref} = X_\theta^{-1}\left[\dot{u}_{des} - X_u u\right] + K_{p_x} e_x + K_{d_x}\dot{e}_x + \eta_x \\ \phi &= \phi_{ref} = Y_\phi^{-1}\left[\dot{v}_{des} - Y_v v\right] + K_{p_y} e_y + K_{d_y}\dot{e}_x + \eta_y \end{aligned}$$

substituting the above equations into the velocity dynamics and assuming perfect

dynamic-inversion, we get

$$\begin{aligned} \dot{u} &= \dot{u}_{des} + K_{p_x} e_x + K_{d_x} \dot{e}_x + \eta_x \\ \dot{v} &= \dot{v}_{des} + K_{p_y} e_y + K_{d_y} \dot{e}_y + \eta_y \\ \dot{x} &= u \\ \dot{y} &= v \end{aligned} \tag{4.32}$$

As we expected from analysis in Appendix A.1, the driving factors in the path following dynamics are the desired tangential acceleration (first term in the right hand side) and the normal tracking error (second and third terms). This model is simple enough to be used online.

The tracking error bounds can be computed from the equation set (6.35) running Monte-Carlo simulations (as shown in Figure 4.20) based on sample values of the random variables η_x and η_y. The vector $[\dot{u}_{des}, \dot{v}_{des}]$ is computed by differentiating the reference velocity vector (tangent to the path).

Chapter 5

Guidance in Known Environment using Spatial Value Function

The trajectory planning for dynamic systems can be formulated as an Optimal Control Problem (OCP). However, such problems are intrinsically NP-hard ([119]), yet to enable necessary interactive capabilities and situational awareness require continuous trajectory update. *Receding Horizon* (RH) optimization can help reduce the computational load by using a finite prediction horizon. To take full advantage of the RH formulation in autonomous guidance, the immediate challenges are to develop Cost-to-go (CTG) functions that are more closely related to the original OCPs value function (VF). In this section we describe the CTG function as an approximate value function derived from a finite-state approximation of the vehicle dynamics (called *Spatial Value Function* (SVF)) [90]. We also demonstrate the integration of the RH trajectory optimization with SVF through real-time guidance experiments with the Blade-CX2 miniature helicopter.

5.1 Autonomous Guidance as Optimal Control Problem (OCP)

The control task in a general trajectory optimization problem is to determine a control history $u(t)$ which will drive the vehicle from its current state x_0 to a desired goal state x_{goal} while minimizing a chosen performance objective of the

form

$$J_\infty(\mathbf{x}) = \int_0^\infty g(\mathbf{x}, \mathbf{u})dt. \tag{5.1}$$

where g is the instantaneous cost function. The mathematical formulation is given as

$$\begin{aligned} \min_{\mathbf{u}} \quad & J_\infty(\mathbf{x}) \\ \text{subject to} \quad & \\ \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{u}) \\ \mathbf{x}(t=0) &= \mathbf{x}_0 \\ \mathbf{x}(t=\infty) &= \mathbf{x}_{goal} \\ \mathbf{u} &\in U \\ \mathbf{x}_p(t) &\in X_{free} \\ \mathbf{x}_v(t) &\in X_{vhc} \end{aligned} \tag{5.2}$$

where $\mathbf{f}$ is the vector differential equation describing the vehicle's equations of motion, and U is the set of admissible controls. The state vector $\mathbf{x}$ is partitioned in vehicle states $\mathbf{x}_v$ and vehicle position $\mathbf{x}_p$. X_{vhc} is the set of admissible vehicle states; X_{free} represents the admissible region of the environment (e.g. obstacle free).

The optimal command history $\mathbf{u}^*(t)$ for the current state $\mathbf{x}$ is the one that minimizes the objective

$$\mathbf{u}^*_\infty(t) = \text{argmin}\{J^*_\infty(\mathbf{x})\}. \tag{5.3}$$

To solve this OCP all its elements must be specified and then converted into a numerical optimization problem i.e., a nonlinear program (NLP). However, the corresponding NLP is NP-hard and can not be solved in real-time. Therefore, approximate OCP is typically used for real-time implementation.

5.2 Receding Horizon Optimization

In Receding Horizon (RH) optimization, the optimization horizon is truncated to a finite duration H. The problem is then solved repeatedly to obtain the control

action based on the most up-to-date state information. The time interval between optimizations, the trajectory update rate, is usually fixed. The minimum update rate is limited by how quickly the online optimization can be solved (or at least how quickly it converges to a reasonable solution). Therefore efficient optimization algorithms and the computational capabilities are critical for system with fast dynamics.

The technique that is the most consistent with the original infinite-horizon form is to approximate the discarded tail of the optimization by a CTG function. The optimal control in the resulting RH+CTG scheme is the one that minimizes the composite cost

$$\mathbf{u}_T^* = \text{argmin}\{J_T(\mathbf{x}(\mathbf{t})) + J_{CTG}(\mathbf{x}(t+T))\}, \tag{5.4}$$

where $J_T(\mathbf{x}(\mathbf{t}))$ is the finite-horizon cost

$$J_T(\mathbf{x}(\mathbf{t})) = \int_0^T g(\mathbf{x}(\mathbf{t}), \mathbf{u}) d\tau, \tag{5.5}$$

and $J_{CTG}(\mathbf{x}(t+T)$ represents the cost of the discarded tail of the trajectory or the CTG function which is a function of the vehicle state attained at horizon end $\mathbf{x}(\mathbf{T})$. Note that if $J_{CTG}(\mathbf{x})$ is identical to the optimal, infinite-horizon cost $J_\infty^*(\mathbf{x})$, the RH- and the infinite-horizon solutions are identical, i.e., the optimality gap is zero. $J_\infty^*(\mathbf{x})$ is also called the optimal value function (VF) $V^*(\mathbf{x})$ [120].

5.3 Cost-to-go (CTG) for Trajectory Optimization

Ideally, the CTG should be chosen as an approximation of the VF associated with the guidance problem's cost function (energy, time, etc) [90]. The optimal VF defines a manifold in $\mathbb{R}^n$ (n is the number of state variables) and is a solution to the Hamilton-Bellman-Jacobi (HBJ) differential equation [120, 121]. However, for many problems of practical interest, this equation can not be solved analytically, therefore computational techniques have to be used. In addition, to accommodate changes occurring when operating in partially known or dynamically changing environments, it is critical to develop computationally efficient algorithms to compute VF.

5.3.1 Finite-State Vehicle Motion Model

Finite-state approximation of Vehicle motion provides an avenue to compute approximate VF. With a finite-state representation of the vehicle dynamics, the trajectory optimization becomes a sequential decision problem which can be solved as a dynamic program [122]. One form of finite-state representation is the *Motion Primitive Automaton* (MPA) which was originally proposed as part of a hybrid guidance system [123, 124]. The finite-state MPA essentially corresponds to a quantization of the system dynamics [125]. As a result of using the finite-state MPA, the value function will be suboptimal (it gives the true optimal solution but for the approximate optimization problem). We use grid-based finite-state MPA which is a particular form of quantization of the vehicle dynamics where the MPs are constrained to share a common spatial grid [126]. Using MPs defined on a fixed spatial grid prevents from having to perform costly interpolations during the value iteration. Furthermore, by limiting the heading resolution to $\pi/4$ it is further possible to take advantage of symmetry properties.

To construct a set of MPAs, the closed-loop performance parameters such as minimum turn radius at different speeds, maximum climb and maximum cruise speed are used [127]. The reduced state-space (x,y,z,Ψ) is discretized: the horizontal grid size d_{xy} is set to minimum turn radius for the lowest non-zero speed, the vertical grid size d_z is set to be consistent with the lowest non-zero horizontal speed and the lowest non-zero ascent/descent speed and finally the heading increment $d\Psi$ is set to have a resolution of $\pi/4$.

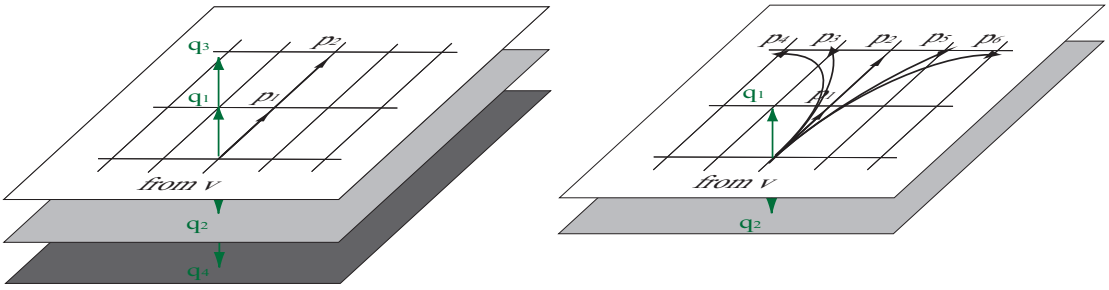


Figure 5.1: Set of horizontal and vertical components of MPA of the feasible flight envelope.

Any motion primitive must be a combination of one horizontal MP and one vertical MP. Corresponding horizontal component MPs and vertical component MPs are illustrated in Figure 5.1 [90]. The use of MPA significantly reduces the

complexity of value functions computation. In fact, using MPAs for approximation of vehicle dynamics, makes the approximate VF to be a manifold in $\mathbb{R}^3$ which is sub-manifold of optimal VF in $\mathbb{R}^n$. The approximate VF is known as *Spatial Value Function* (SVF) is defined as follows [90]

Definition 1 Spatial Value Function (SVF)
Given the finite-state vehicle dynamics model and the quantized environment state, SVFs are the true optimal VF for the finite state vehicle model and are suboptimal OCP's value functions.

5.3.2 CTG Computation Algorithm

The algorithm for the CTG Computation is shown in the Algorithm 1. For a given elevation map, goal, motion primitives and cell resolutions, Algorithm 1 computes the CTG and the velocity vector at each cell in the map. The algorithm is based on Dijkstra shortest path computing algorithm with priority queue data structure for speedy retrieval and maintenance of the cells. We use Fibonacci heap for implementation of the priority queue which improves the asymptotic running time of the algorithm. Each cell has the CTG value and three velocities (v_x, vy and v_z) along each coordinate axis. The velocities are the actions associated with the value function at each cell and indicate the required velocities that the vehicle must have at each cell.

When planning in the presence of uncertainty, the above CTG computation procedure is done for the a-priori map of the environment. Then, as we will explain in Chapter 6, the CTG is updated as the true environment unveils to the vehicle.

5.4 Experimental Demonstration

In this section some the of important features of guidance using Spatial Value Functions (or CTG function) are demonstrated through experiment. The experiments have done in our lab facility explained in Chapter 2. To exercise the guidance scheme, we consider three different obstacle field navigation tasks: (i) Navigation between multiple goals (ii) Guidance with terminal state constraint (iii) 3D guidance tasks. Subsections below demonstrate the experimental results for each task.

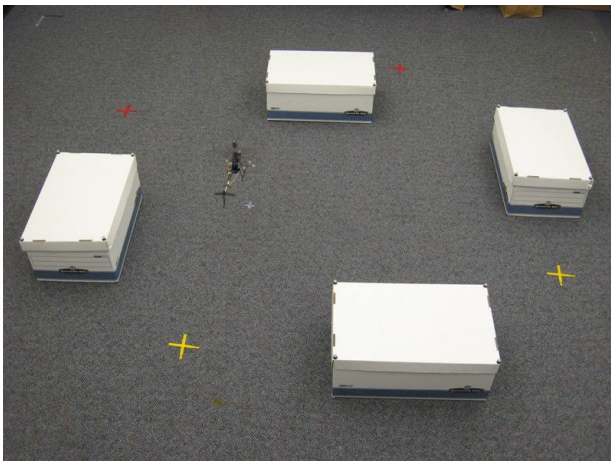
Algorithm 1: CTG Computation

Input: free and occupied cells is the map, vehicle motion primitives, goal state.
Output: CTG value and the corresponding velocity vector for each free cell.

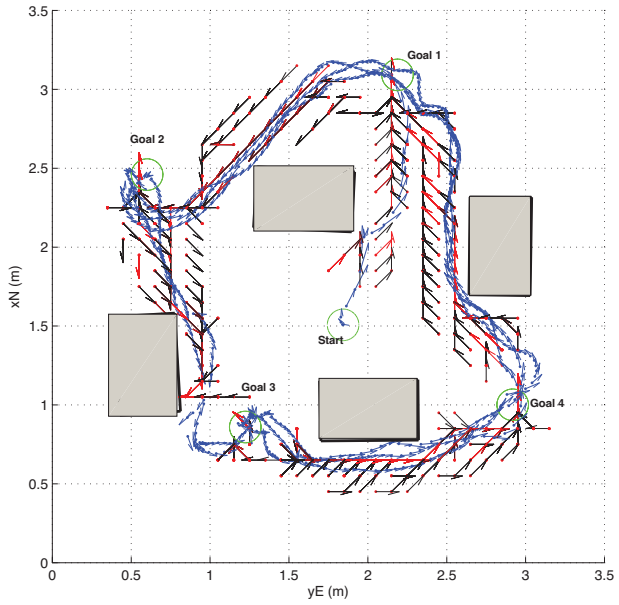
```
forall cell q ∈ Map do
    q.ctg = ∞;
    q.v_x = q.v_y = q.v_z = ∞;
    if q is occupied then
        q.isfree = false;
    else
        q.isfree = true;
        InsertToPriorityQueue(PQ, q, q.ctg);
while PriorityQueue not empty do
    p ← ExtractAndDeleteMin(PQ);
    find children of p (children{p}) through motion primitives;
    forall r ∈ children{p} do
        if r.isfree == true then
            if r.ctg > p.ctg + c(r, p) then
                AdjustPriorityQueue(PQ, r, p.ctg + c(r,p));
                compute q.v_x, q.v_y, q.v_z based on motion primitives;
```

5.4.1 Task 1: navigation between multiple goals

In this task we conducted an experiment in which the helicopter navigates autonomously amid obstacles between four different goals as shown in Figure 5.2. As the helicopter reaches a goal (to be more precise, a circular region of radius 10 cm around the goal), the task manager automatically switches to the next goal and loads the corresponding CTG function. We run this experiment over several cycles to illustrate how the system 'adapts' its trajectory to its current state, continuously generating the optimal trajectory for the current conditions. Variability in the performance is inevitable due to disturbances including the effects of the helicopter's own wake. Even with these effects, the behavior from one cycle to the next is relatively consistent. Note that if the trajectories were precomputed, the performance would be sub-optimal in the presence of disturbances. Figure 5.2 (b), shows the xy trajectory of the helicopter along with its velocity vector and the velocity vectors at the intermediate waypoints which have the best CTG value.



(a) The helicopter navigates amid obstacles between different goals over several cycles.



(b) Blue vectors are the helicopter velocity and the red vectors are the velocity vectors at the intermediate waypoints which have the best CTG.

Figure 5.2: Experimental evaluation of the autonomous guidance system.

5.4.2 Task 2: guidance with terminal state constraints

The purpose of this task is to show that the trajectory can be greatly influenced by the terminal state constraint. To be more specific, it is know from optimal control theory that trajectory optimization is a Two Point Boundary Value Problem (TPBVP) in which different terminal constraint result in various solution to the

problem. For the purpose of demonstration, here we choose the goal's heading as a terminal constraints for the problem. However, it can be the terminal velocity as well. Figure 5.3 (a,b), illustrate the trajectories in which the vehicle starts from different starting locations and flies to the goal with different terminal headings.

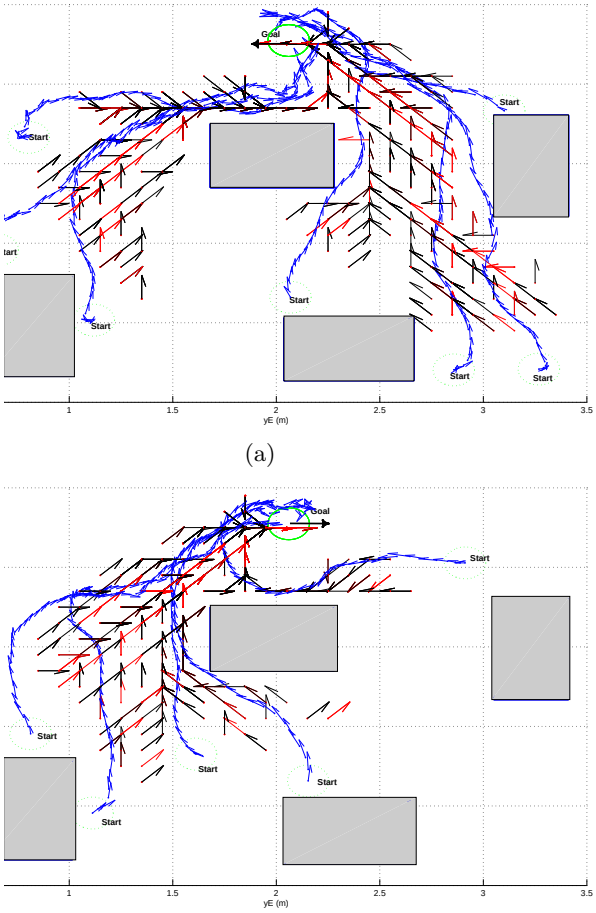
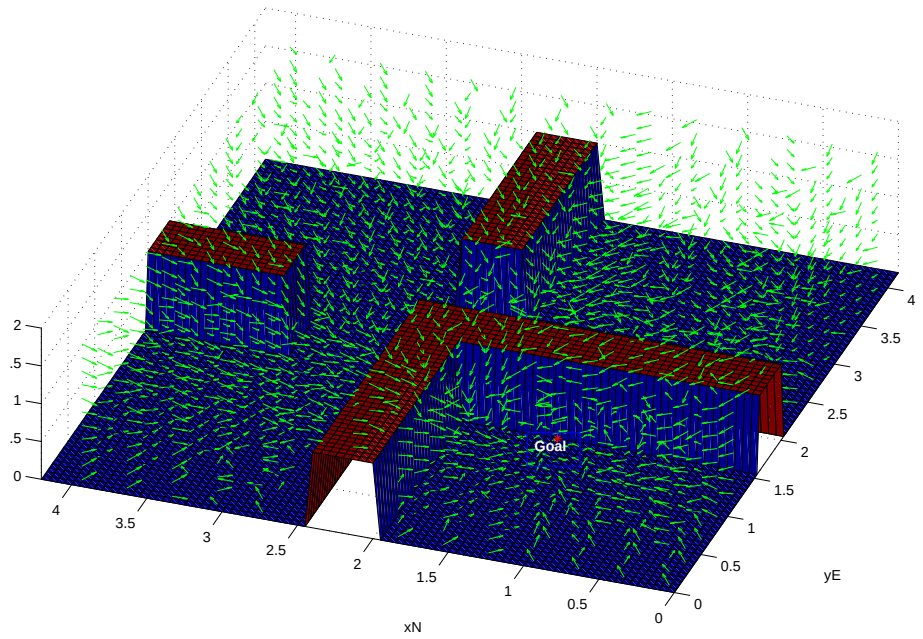


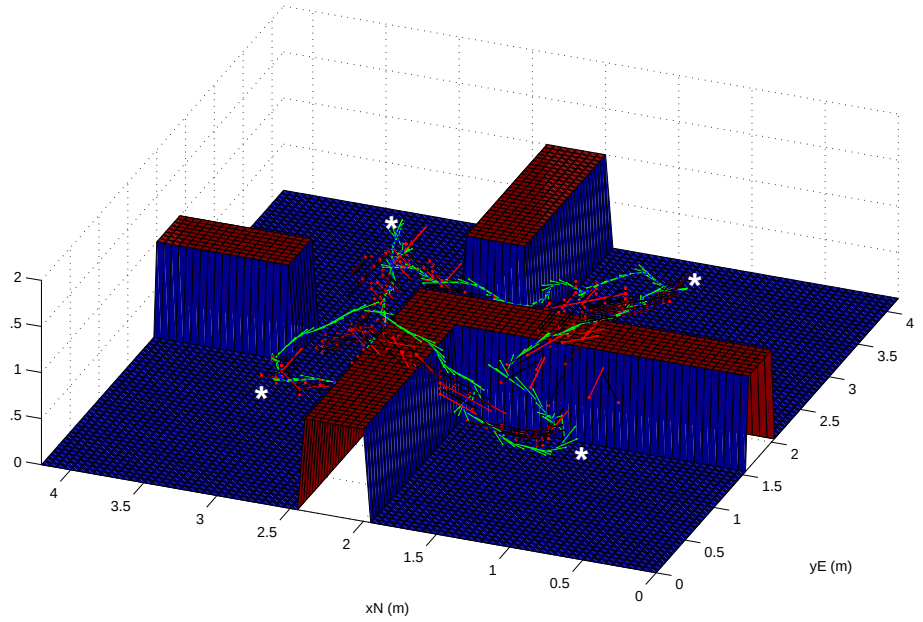
Figure 5.3: Experimental helicopter trajectory toward the goal for different starting points: top and button plots show how the goal's heading (i.e., terminal constraint) affects the trajectory.

5.4.3 Task 3: guidance in 3D environment

Finally, in order to exploit the full capability of the aerial vehicle in autonomous guidance, 3D guidance solution is inevitable. For 3D tasks, the environment decomposition and CTG map generation is done based on MPAs in 3D. Figure 5.4, shows the experimental trajectory as well as the selected intermediate waypoints when the helicopter navigates between different goals.



(a) 3D vector field generated by MPAs



(b) Experimental helicopter trajectory (green) and velocity vectors at the intermediate waypoints which have the best cost-to-go value (red)

Figure 5.4: Experimental results for guidance experiment in a 3D environment.

Chapter 6

Sensory Predictive Guidance in Partially Known Environment

This chapter addresses the problem of autonomous navigation through a partially known or unknown cluttered environment. The proposed hierarchical guidance framework is developed as an extension to the proposed guidance algorithm based on *Receding Horizon* optimization with *Spatial Value or Cost-to-Go* function that was discussed in Chapter 5. To accommodate plan updates when operating in partially known or unknown environments, we provide methods to update the CTG and the environment map using the exteroceptive onboard sensory information. The closed-loop performance characterizations which was addressed in Chapter 4 is used here to ensure a tight integration between the planner and the inner-loop controller. More specifically, the RH trajectory optimization accounts for 1) *absolute maneuvering envelope* as well as 2) *nominal closed-loop model*. Also to ensure safe path tracking, the path tracking error model which was developed in Section 4.3.5 is used in real-time to predict the tracking error bound based on a pre-specified path generated by the RH path planner in each planning cycle. Finally, a reactive planning technique is integrated with the guidance framework to provide higher level of safety and reliability as well as a last resort solution for dealing with uncertainty and obstacle avoidance.

We start the chapter by overviewing the mathematical framework for planning under uncertainties. Then, we describe the simulation scenario for the *Sensory Predictive Guidance* (SPG). Figure 1.7 gives an overview of SPG. The multi-layer hierarchical guidance system relies on the interaction of several subsystems. In

this chapter each subsystem and interaction between them are described in details.

6.1 Mathematical Foundation for Planning Under Uncertainties

In this section we lay out mathematical foundations used for the planning problem in uncertain environment. The material is adapted from Reference [128]. Let the state of the system at time step t denoted by s_t and the state of the environment at system state s_t denoted by $q(s_t)$. The pair $(s_t, q(s_t))$ represents to the state of the overall system in the uncertain environment. Both s_t and $q(s_t)$ are obtained through quantization of the vehicle dynamics and environment respectively, and as a result belong to finite state spaces. The state transition matrix given a control action u_t is defined as

$$s_{t+1} = f(s_t, u_t, \theta_t) \tag{6.1}$$

where $\theta_t \in \Theta(s_t, u_t)$ is the disturbance acting on the system. Given $p(\theta_t \,|\, s_t, u_t)$ the probability of the state transition matrix is obtained as follows

$$p(s_{t+1} \,|\, s_t, u_t) = \sum_{\theta_t \in \Theta'} p(\theta_t \,|\, s_t, u_t) \tag{6.2}$$

in which

$$\Theta' = \{\theta_t \in \Theta(s_t, u_t) \,|\, s_{t+1} = f(s_t, u_t)\} \tag{6.3}$$

In fact, the calculation of $p(s_{t+1} \,|\, s_t, u_t)$ involves accumulating all the probability masses that could lead to s_{t+1} under the action of various uncertainties θ_t.

In addition, we assume that the random process that describes the environment is spatially uncorrelated and temporarily stationary. Assume also that the probability of the current system state only depends on the state and control input at previous time instant. Under these assumptions the following holds

$$p((s_{t+1}, q(s_{t+1})) \,|\, s_0, q(s_0), u_0, \ldots, s_t, q(s_t), u_t) = p(s_{t+1} \,|\, s_t, u_t)\, p^*(q(s_{t+1})), \tag{6.4}$$

where $p^*(q(s_t))$ is the true probability of the environment at system state s_t.

Each $q(s_t)$ corresponds to a cell in the quantized environment q_i. Therefore, q_i is a binary variable that does not change over time. In addition, $z_{1:t}$ represents the set of all the environment measurements up to time t that was obtained by the vehicle on-board sensor. Based on the assumption that the environment random process is spatially uncorrelated, one can write the following equation for the environment probability

$$p(q \mid z_{1:t}) = \prod_i p(q_i \mid z_{1:t}) \tag{6.5}$$

where $p(q_i)$ refers to the probability that the cell q_i is occupied.

Based on the above definitions and assumptions, it is now possible to state the path planning problem in uncertain environment that will be addressed in this chapter:

Given the initial states $(s_0, q(s_0))$ and a positive cost incremental function $C((s_t, q(s_t)), (s_{t-1}, q(s_{t-1})), u_{t-1})$ that captures the cost of transition from the state $(s_{t-1}, q(s_{t-1}))$ to $(s_t, q(s_t))$ under control action u_{t-1}, the optimal control policy in uncertain environment minimizes the following stochastic cost function

$$J = \mathbb{E}\left\{ \sum_{t=1}^{\infty} \gamma^t C(\,(s_t, q(s_t)), (s_{t-1}, q(s_{t-1})), u_{t-1}\,) \mid (s_0, q(s_0)) \right\} \tag{6.6}$$

Here the expectation $\mathbb{E}\{\}$ is taken over all the transition costs (or payoff values) from time $t-1$ to time t. Each individual transition cost is multiplied by a factor γ^t, called the discount factor where $\gamma \in [0; 1]$. Smaller values of γ makes the more recent costs more important than the later ones.

The optimal control policy is the one that minimizes the total expected cost (6.6) with respect to all control policies. According to the optimality principle, the optimal control policy is given by [128]

$$\begin{aligned} u^*(s_t, q(s_t)) \;=\; & \arg\min_u \sum_{(s_{t+1}, q(s_{t+1}))} p((s_{t+1}, q(s_{t+1})) \mid s_0, q(s_0), u_0, \ldots, s_t, q(s_t), u_t) \\ & [C(\,(s_{t+1}, q(s_{t+1})), (s_t, q(s_t)), u_t\,) + \gamma J^*(s_{t+1}, q(s_{t+1}))] \end{aligned} \tag{6.7}$$

which according to (6.4) is equal to

$$u^*(s_t, q(s_t)) = \arg\min_{u} \sum_{(s_{t+1}, q(s_{t+1}))} p(s_{t+1} \mid s_t, u_t) p^*(q(s_{t+1})) \\ [C(\,(s_{t+1}, q(s_{t+1})), (s_t, q(s_t)), u_t\,) + \gamma J^*(s_{t+1}, q(s_{t+1}))] \tag{6.8}$$

where $J^*(s_{t+1}, q(s_{t+1}))$ is the cost-to-go from the state $(s_{t+1}, q(s_{t+1}))$. Also $p^*(q(s_{t+1}))$ is the true probability of the environment which can be estimated on-line using the on-board sensor measurements. It can be shown that if the environment probability approaches its true value, the cost-to-go function approaches its optimal value [128]. The dependency of the optimality equation (6.8) on the disturbances θ_t is implicit through the expression for $p(s_{t+1} \mid s_t, u_t)$ which according to (6.2) and (6.3) uses $p(\theta_t \mid s_t, u_t)$ and state transition equation f. Figure 6.1 illustrates the dynamic programming in uncertain environments.

Equation (6.8) is a stochastic dynamic programming problem, which is computationally intractable for most real problems (due to large number of state variables i.e., curse of dimensionality). In addition, the problem formulation (6.8) is restricted to finite-state approximation of the vehicle dynamics and does not consider the differential equations of motion and the corresponding state/input constraints. The approximate approach used in the paper is to estimate the environment probability $p^*(q(s_t))$ as well as the cost-to-go function $J^*(s_{t+1}, q(s_{t+1}))$ separately. This is known as *certainty-equivalence* [80] in which the models for the environment and the cost-to-go are learned continuously and at each planning cycle the current models are used to compute an optimal policy. In addition, for the execution phase, in order to generate a dynamically feasible action for the vehicle, a *Receding Horizon* optimization is solved.

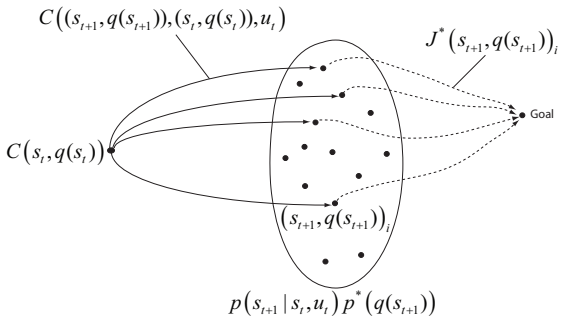


Figure 6.1: Illustration of dynamic programming in uncertain environment.

6.2 SPG Simulation Scenario Overview

This section describes the simulation scenario for the sensory predictive guidance. We explain the results of this simulation scenario throughout this chapter. In order to fully demonstrate the capabilities of the *sensory productive guidance* framework, we have chosen an a-priori unknown full 3D indoor environment. Many guidance algorithms in the literature have been demonstrated in either 2D (planar) or 2.5D. In 2.5D representations, a 2D grid is used to store the height of each cell. Therefore, a general drawback of guidance algorithms based on 2.5D representations are that they cannot operate in environments involving bridges or tunnels as well as indoor environments. To demonstrate the full 3D capabilities of our guidance system, we have graphically modeled the Akerman Hall at the University of Minnesota (see Figure 6.2-a) at sub-meter accuracy and used for our simulations. The rotorcraft in the simulations is our test platform Blade-CX2 coaxial indoor helicopter.

In addition, to sense the environment we assume that the helicopter carries a 2D laser scanner (see e.g. Hokuyo UGR-04LX-UG01 Laser Rangefinder [129]) that is mounted on a servo that rotates from [-90 90] degrees about the helicopter z axis at the frequency of 1 Hz. This provides a full 3D point cloud of the surrounding environment. The range of the laser scanner is 4 meters. The point could is generated using the OpenSceneGraph [130] ray casting method to find the intersection of the laser rays with the graphical objects in the model.

To demonstrate the full 3D capabilities of the guidance algorithm, the starting location of simulation scenario chosen to be in the basement of Akerman Hall and the goal is located in the hangar area in the first floor. The helicopter has no a-priori knowledge of the environment and only knows the location of the goal.

6.3 Updating Environment Probability from Depth Sensor Readings

The prior environment probability (or a-prior map) is usually given based on the map accuracy. This map usually does not contain enough details about the environment to ensure safe path planning. However, the initial map can be improved using on-board sensor measurements. We formalize the problem of environment

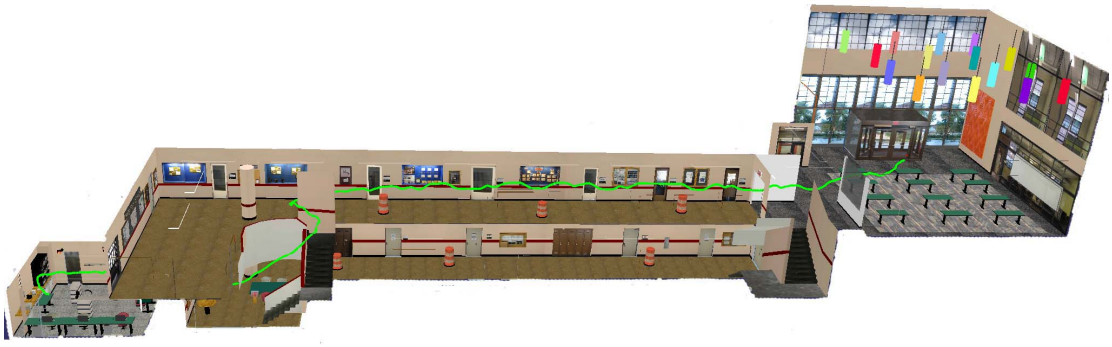


Figure 6.2: In the 3D simulation scenario, Blade-CX2 helicopter flies autonomously in Akerman Hall at the University of Minnesota.

probability state estimate from a series of measurements i.e., $p(q \mid z_{1:t})$, where $z_{1:t}$ are the measurements up to time step t. As mentioned earlier, each cell in the environment is a binary state that does not change over time. This estimation problem can be solved using a binary Bayes filter [131]. Given the fact that

$$p(x \mid y, z) = \frac{p(y \mid x, z)p(x \mid z)}{p(y \mid z)}, \tag{6.9}$$

we have

$$\begin{aligned} p(q_i \mid z_{1:t}) &= \frac{p(z_t \mid q_i, z_{1:t-1})p(q_i \mid z_{1:t-1})}{p(z_t \mid z_{1:t-1})} \\ &= \frac{p(z_t \mid q_i)p(q_i \mid z_{1:t-1})}{p(z_t \mid z_{1:t-1})}, \end{aligned} \tag{6.10}$$

where q_i is the ith cell of the quantized environment. In the above equation we assume that the measurements at each time steps are independent. Also according to Bayes rule we get

$$p(z_t \mid q_i) = \frac{p(q_i \mid z_t)p(z_t)}{p(q_i)}. \tag{6.11}$$

Substituting (6.11) into (6.10) yields

$$p(q_i \mid z_{1:t}) = \frac{p(q_i \mid z_t)p(z_t)p(q_i \mid z_{1:t-1})}{p(q_i)p(z_t \mid z_{t-1})}, \tag{6.12}$$

and since q_i is a binary variable we have: $p(\sim q_i \mid z_{1:t}) = 1 - p(q_i \mid z_{1:t})$. Now dividing $p(q_i \mid z_{1:t})$ by its negate will remove the term $p(z_t \mid z_{t-1})$ from (6.12) and

yields

$$\begin{aligned} \frac{p(q_i \mid z_{1:t})}{1 - p(q_i \mid z_{1:t})} &= \frac{p(q_i \mid z_t)}{p(\sim q_i \mid z_t)} \frac{p(q_i \mid z_{1:t-1})}{p(\sim q_i \mid z_{1:t-1})} \frac{\sim p(q_i)}{p(q_i)} \\ &= \frac{p(q_i \mid z_t)}{1 - p(q_i \mid z_t)} \frac{p(q_i \mid z_{1:t-1})}{1 - p(q_i \mid z_{1:t-1})} \frac{1 - p(q_i)}{p(q_i)}. \end{aligned} \tag{6.13}$$

Solving for $p(q_i \mid z_{1:t})$, we get the following equation for the probability of the cell q_i given all measurements up to the current time

$$p(q_i \mid z_{1:t}) = 1 - \left\{ 1 + \frac{p(q_i \mid z_t)}{1 - p(q_i \mid z_t)} \frac{p(q_i \mid z_{1:t-1})}{1 - p(q_i \mid z_{1:t-1})} \frac{1 - p(q_i)}{p(q_i)} \right\}^{-1}. \tag{6.14}$$

Note that in equation (6.14), $p(q_i)$ is the prior occupation probability of the cell q_i, $p(q_i \mid z_t)$ is the measurement update and finally $p(q_i \mid z_{1:t-1})$ is the belief probability of the cell based on the previous measurements. Hence, equation (6.14) has the structure of a recursive Bayes filter.

In order to avoid numerical instability for probabilities close to one or zero, log-odds representation of equation (6.14) is used in practice [131]. The odds of an event is the ratio of its probability divided by the probability of its negate). Therefore, the log-odd of the state x is defined as

$$\Lambda = \log \frac{p(x)}{1 - p(x)}. \tag{6.15}$$

Taking the log of both side of equation (6.13) yields

$$\Lambda_{t,i} = \Lambda_{t-1,i} + \log \frac{p(q_i \mid z_t)}{1 - p(q_i \mid z_t)}. \tag{6.16}$$

where $\Lambda_{t,i}$ is the log-odds probability of the ith cell in the environment at time step t and $\Lambda_{0,i} = \frac{p(q_i)}{1-p(q_i)}$ corresponds to the initial probability of occupation of the cell q_i. The term $p(q_i \mid z_t)$ is the *risk map update* function [132]. This function is unknown and has to be computed on-line based on to the incoming measurements z_t. This requires building an approximation for the probability of the ith cell occupancy after a single observation on the cell q_i. Therefore, $p(q_i \mid z_t)$ will directly depend on the type of sensor. For the case of range sensor devices (such as laser range finder or depth map obtained from stereo vision) the following risk map

update equation was suggested [133]

$$p(q_i \mid z_t) = \frac{-c_1}{\sigma\sqrt{2\pi}} \left(1 + exp\left[\frac{2\pi(\rho_i + 2\sigma)}{\sigma\sqrt{3}}\right]\right)^{-1} + \frac{c_2}{\sigma\sqrt{2\pi}} \left(exp\left[-\frac{\rho_i^2}{2\sigma^2}\right]\right), \quad (6.17)$$

where ρ_i is the difference between the corresponding measured depth of the cell q_i (i.e., D_i) and the distance between the cell and the vehicle r_i (see Figure 6.3)

$$\rho_i = D_i - r_i. \quad (6.18)$$

The variance of ρ_i, σ^2, depends on both the uncertainty in the vehicle location as well as the on-board sensor's accuracy. Also the constants c_1 and c_2 are used to scale each contribution in equation (6.17) to achieve a desired risk map that is appropriate for the typical depth of the obstacle in the environment.

To efficiently determine the cells which need to be updated, a ray-casting operation is performed using the 3D variant of Bresenham's algorithm [134, 135]. Bresenham's algorithm is an efficient way to enumerate all of the points between two endpoints. Once the risk map update $p(q_i \mid z_t)$ is computed, the occupation probability of cell q_i can be recursively estimated according to equation (6.16) as new measurements become available. Algorithms 2 summarizes the environment probability update procedure in 3D.

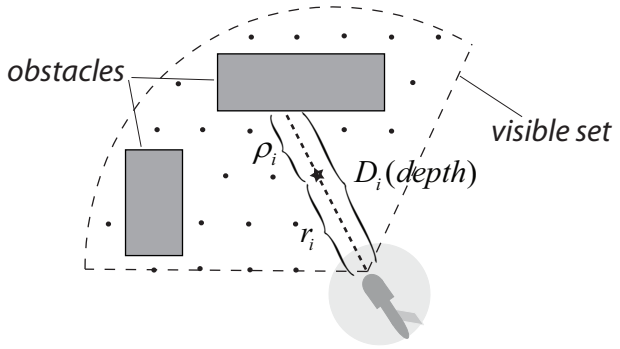


Figure 6.3: Map update using depth map sensors.

Figure 6.4 shows the result of applying Algorithm 2 for updating occupancy probability of each cell using measurements from the laser scanner as the vehicle makes progress toward the goal. Initially, all the cells in the unknown environment are assigned an occupation probability of 0.5. The map update algorithm recursively updates the probability of each node in the visible set. The blue color

grids in Figure 6.4 represent probability of occupation close to zero (free) and red color represent probability of occupation close to 1 (not free). The probability of the undiscovered area of the map is unchanged (green color).

Algorithm 2: Environment Map Update in 3D

Input: vehicle pose (x_R, y_R, z_R), laser measurements, prior occupancy probability $\Lambda_{t-1,i}$.

Output: occupancy probability for all visible cells at current time $\Lambda_{t,i}$.

forall *3D laser beams*, $(i = 1, \ldots, N_{beam})$ **do**

Record the end point of the beam (x_i, y_i, z_i)

Run the 3D Bresenham's raycasting algorithm [135] to get all the points (x_{pj}, y_{pj}, z_{pk}) lie between (x_R, y_R, z_R) to (x_i, y_i, z_i), $(j = 1, \ldots, N_{points})$

forall *points* (x_{pj}, y_{pj}, z_{pk}), $(j = 1, \ldots, N_{points})$ **do**

$r_i \leftarrow \sqrt{(x_{pj} - x_i)^2 + (y_{pj} - y_i)^2 + (z_{pj} - z_i)^2}$.

$p(q_i \mid z_t) \leftarrow p(q_i \mid z_t) = \frac{-c_1}{\sigma\sqrt{2\pi}} \left(1 + exp\left[\frac{2\pi(\rho_i + 2\sigma)}{\sigma\sqrt{3}}\right]\right)^{-1} + \frac{c_2}{\sigma\sqrt{2\pi}} \left(exp\left[-\frac{\rho_i^2}{2\sigma^2}\right]\right)$.

$\Lambda_{t,i} \leftarrow \Lambda_{t-1,i} + \log \frac{p(q_i \mid z_t)}{1 - p(q_i \mid z_t)}$.

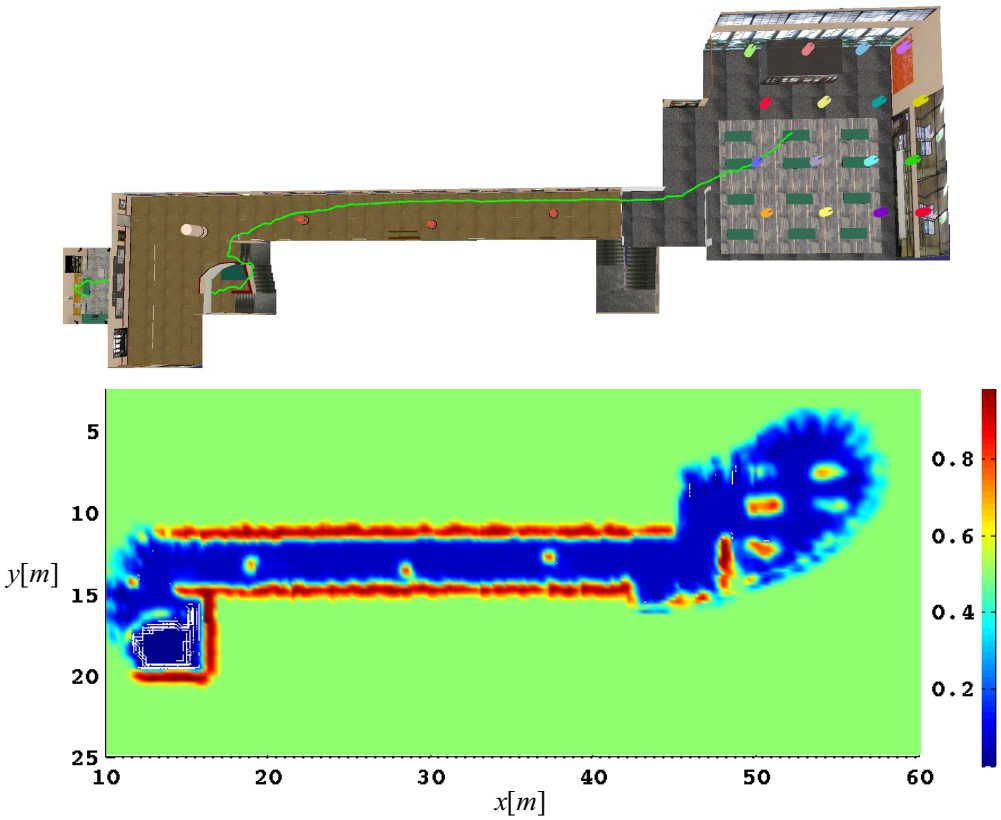


Figure 6.4: Occupancy probability obtained while helicopter flew the green path in the top plot with no a-priori knowledge about the environment.

6.4 Updating Global Cost-to-go (CTG) Function

The CTG function is an approximation of the optimal value function (VF) associated with the guidance problem's cost function (energy, time, etc) [90]. The optimal VF defines a manifold in $\mathbb{R}^n$ (n is the number of state variables) and one of the issues of computing the optimal VF is the curse of dimensionality. Therefore, a first step toward making a more tractable formulation is to introduce practical approximation for the optimal VF. As discussed in Chapter 5 we use quantization of the state of vehicle dynamics as well the state of the environment. Given the quantized environment, the goal state and the MPAs, the Dijkstra's backward search algorithm (see Algorithm 1) is used to compute the SVF and the corresponding *Velocity Vector Field* (VVF) at each cell of the environment. VVFs are basically optimal states (actions) associated with the SVF for the finite state vehicle model. Both the SVF and the VVF are stored in the memory. Note that the term CTG refers to the SVF. We use these two term interchangeably throughout the paper.

To accommodate changes occurring when operating in partially known or unknown environments, we use dynamic version of Dijkstra's algorithm [61]. Rather than re-calculating the entire CTG map when changes in the map are detected, a reduced set of cells are checked and their CTG values are updated in each iteration. In this approach, first the CTG map is computed for the entire environment based on a-priori knowledge of the terrain and the obstacles. When an obstacle is detected by the vehicle's on-board sensors, the occupancy map is registered and updated according to Algorithm 2. The CTG update algorithm then first detects the environment cells in the vehicle's visible set which their occupancy has changed. These cells are inserted into a priority queue. Second, the algorithm for updating the CTG function is applied (outlined in Algorithms 3). The algorithm, maintains two cost values for every cell in the graph: 1) CTG_{cur}, which is the current CTG value (initially obtained from the apriori map) and 2) RHS, which is one step look-ahead cost value based on the neighboring cells

$$RHS(u) = \min_{s' \in succ(u)} \{C(u, s') + CTG_{cur}(s')\} \tag{6.19}$$

where $succ(u)$ is the set of all successor neighboring cells of u based on the MPAs. $C(a, b)$ is the cost of going from cell a to b according to the motion primitives. The algorithm maintains a priority queue of the inconsistent cells (both *over-consistent* and *under-consistent*) as the on-board sensor discovers the new cells. The cell u is *over-consistent* if its current CTG is larger than their neighboring cells (i.e., $CTG_{cur}(u) > RHS(u)$).

In this case, the cell u and all its neighboring cells should adjust their cost in the priority queue unless they are consistent (i.e., $RHS = CTG_{cur}$). Once consistent, they are removed from the priority queue. In addition, cell u is *under-consistent* if $CTG_{cur}(u) < RHS(u)$. In this case, the CTG of the cell is updated to infinity (large number) and then the priority queue is updated for all the neighboring cells. The assignment of an infinite cost to a cell in this step converts the under-consistent cell into either consistent or over-consistent (which will be re-adjusted in the priority queue in the next iterations). The process is completed either when the priority queue becomes empty or the CTG of the current vehicle location become smaller than the CTG of the top cell in the priority queue (the minimum cost cell in the priority queue).

It is to be noted that the algorithm presented here, uses the CTG map of the a-priori environment for initialization of the cost for each cell and does not use heuristics to focus the search toward the goal as implemented in D* Lite algorithm [64]. In other words, we are interested in updating the CTG for the entire visible set as the vehicle explores the environment for new obstacles. This provides reactive and adaptive guidance capabilities when operating in partially known environment.

Figure 6.5 shows the result of applying Algorithm 3 for updating the CTG as the vehicle progresses toward the goal. Initially, the entire unknown environment is treated as an empty 3D box with a given goal location. The CTG of the a-priori environment is then generated for the 3D box. The CTG update algorithm, then updates the CTG of each node as it becomes visible by the laser scanner. Figure 6.5-c shows the corresponding VVF.

Algorithm 3: CTG Map Update

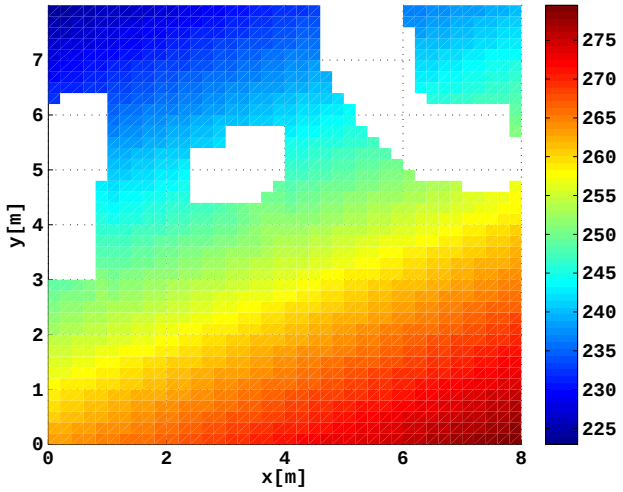
Input: vehicle pose, cells in the visible set, prior and current occupancy map, a-priori CTG map.

Output: updated CTG value for the cells in the visible set.

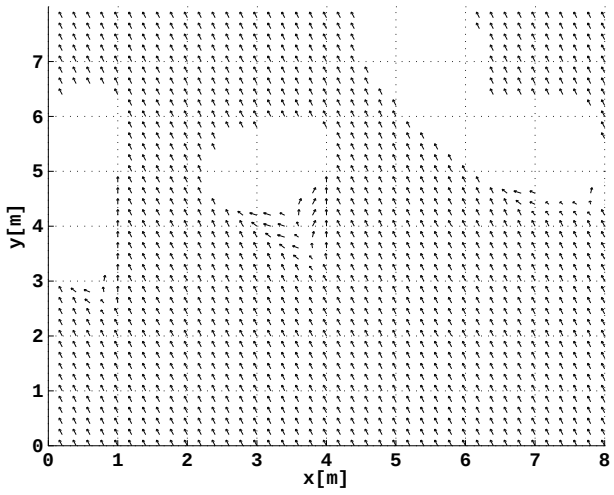
forall *cell* $q \in Visible\,Set$ **do**
  if *(*$q.prob < 0.2$ *and* $q.isfree == 0$*) or (*$q.prob > 0.8$ *and* $q.isfree == 1$*)* **then**
    $q.rhs \leftarrow \infty$;
    $q.isfree \leftarrow \sim q.isfree$;
  $q.rhs \leftarrow \min_{p \in succ(q)}\{c(q,p) + p.ctg\}$
  if $q.rhs \neq q.ctg$ **then**
    InsertToPriorityQueue(PQ, q, $\min\{q.rhs, q.ctg\}$)

while $CurrentPose.ctg > \min\{p.rhs, p.ctg\}$ **do**
  p $\leftarrow$ ExtractAndDeleteMin(PQ);
  $p.rhs \leftarrow \min_{s \in succ(p)}\{c(p,s) + s.ctg\}$;
  if $p.rhs < p.ctg$ **then**
    $p.ctg \leftarrow p.rhs$;
    forall $r \in pred\{p\}$ **do**
      $r.rhs \leftarrow \min_{s \in succ(r)}\{c(r,s) + s.ctg\}$;
      if $r.rhs \neq r.ctg$ **then**
        AdjustPriorityQueue(PQ, r, $\min\{r.rhs, r.ctg\}$);
      else
        if $r \in PQ$ **then**
          DeleteFromPriorityQueue(PQ, r)
  else
    $p.ctg \leftarrow \infty$;
    forall $r \in pred\{p\}$ **do**
      $r.rhs \leftarrow \min_{s \in succ(r)}\{c(r,s) + s.ctg\}$;
      if $r.rhs \neq r.ctg$ **then**
        AdjustPriorityQueue(PQ, r,$\min\{r.rhs, r.ctg\}$);
      else
        if $r \in PQ$ **then**
          DeleteFromPriorityQueue(PQ, r)

(a) Vehicle's immediate environment in obstacle field.



(b) CTG in the local window around the vehicle.



(c) Corresponding velocity vector field (VVF).

Figure 6.5: CTG and VVF are updated as the information about environment becomes available through an exteroceptive sensor.

6.5 Control and Path Following System

As mentioned in Section 1.5.3, our 2DOF guidance architecture is wrapped around an inner-loop control system. We used a nested architecture for the control system with an attitude inner-loop and a velocity outer-loop which is augmented with a path following controller to track reference trajectories as illustrated in Figure 1.5. The reader is referred to Chapter 4 for more details about the control system in horizontal flight. Due to satisfactory decoupling between the lateral-longitudinal dynamics and altitude-heading dynamics in Blade-CX2 helicopter, we used classical controllers to control the altitude and heading as shown in Figure 6.6. The reference vertical velocity $v_{z_{ref}}$ and the reference altitude z_{ref} are provided by the path following layer. The heading rate command is computed based on the location of *Active WayPoint* (AWP) (see Section 6.9) and the current rotorcraft heading ψ so that the rotorcraft always points toward the AWP as follows

$$\dot{\psi}_{ref} = K_{\psi}\left(tan^{-1}(\frac{X_{AWP}-x}{Y_{AWP}-y}) - \psi\right) \tag{6.20}$$

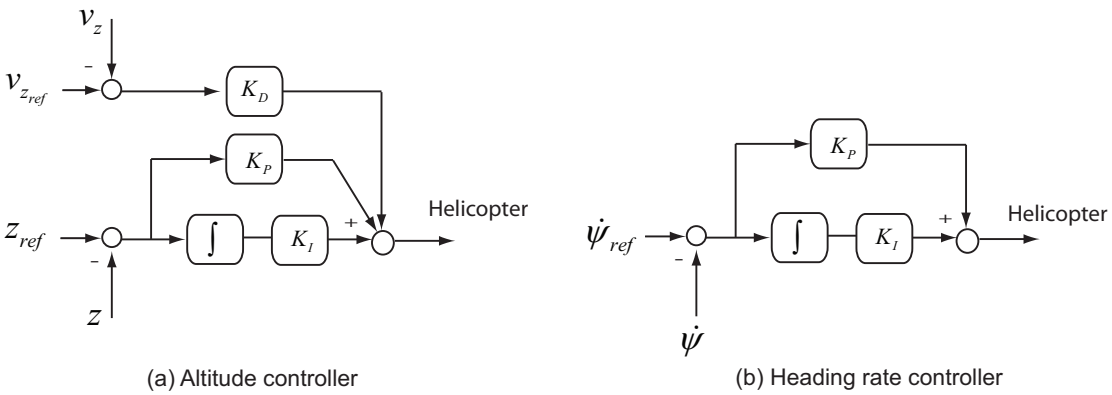


Figure 6.6: Altitude and heading rate controllers for Blade-CX2 rotorcraft.

6.6 Receding Horizon (RH) Optimization

In 2-DOF guidance architecture discussed in Section 1.5.3, a feasible open-loop trajectory is generated by solving a finite-horizon constrained optimal control problem starting from the current state to a *Active Waypoint* (AWP). A variety of performance objective can be chosen for the RH optimization problem, but the minimum-time is preferred as it pushes the performance to the constraints. To account for the discarded

tail of the optimization, the SVF described in Section 6.4 is used to approximate the value function from the AWP to the goal state. Without appropriate SVF, arrival to the goal in bounded time cannot be guaranteed.

A variety of optimization tools can be used for the online RH planner. *Mixed Integer Linear Programing* (MILP) (see e.g., References. [136, 137]) approach is used here. A minimum-time trajectory from the current state to the AWP is generated by minimizing the following cost [90]

$$\begin{aligned} J &= \sum_{i=1}^{H}\{M(H+1-i)(1-b_i)+(H-i)(R_{err}+Z_{err}) \\ &+ \alpha(|v(i)_{x_{cmd}}|+|v(i)_{y_{cmd}}|+|v(i)_{z_{cmd}}|)\} \end{aligned} \tag{6.21}$$

subject to

$$\ell_a([x(i)-x_{awp},\, y(i)-y_{awp}]) \leq M(1-b_i)+\epsilon_p \tag{6.22}$$

$$\ell_a(\left[v_x(i)-v_{x_{awp}},\, v_y-v_{y_{awp}}\right]) \leq M(1-b_i)+\epsilon_v \tag{6.23}$$

$$-M(1-b_i)-\epsilon_p \leq z(i)-z_{awp} \leq M(1-b_i)+\epsilon_p \tag{6.24}$$

$$-M(1-b_i)-\epsilon_v \leq v_z(i)-v_{z_{awp}} \leq M(1-b_i)+\epsilon_v \tag{6.25}$$

$$\sum_{i=1}^{H} b_i = 1 \tag{6.26}$$

$\ell_a(.)$ is a linear approximation of the norm of a vector $r=[x_1,\, x_2]$ defined as follows [136]

$$\ell_a(r) = s \,:\, s \geq L_1x_1+L_2x_2,\ \forall(L_1,L_2)\in\mathcal{L} \tag{6.27}$$

where $\mathcal{L}$ is the finite set of unit vector whose directions are distributed between $0^\circ - 360^\circ$. Also, M is a large positive number. Constraints (6.22) and (6.23) are added to ensure that the position and velocity errors at the arrival time are less than ϵ_p and ϵ_v, respectively (usually small positive numbers). The variables b_i, $i=1\cdots H$, are the binary decision variables. When the binary variable b_i is zero the arrival condition is relaxed. In addition, the parameter r_{err} is an upper-bound on the position error

$$\ell_a([x(i)-x_{awp},\, y(i)-y_{awp}]) \leq R_{err} \tag{6.28}$$

$$-Z_{err} \leq z(i)-z_{awp} \leq Z_{err} \tag{6.29}$$

The first term in the cost function J in equation (6.21) minimizes the arrival time at

the AWP. Reaching the AWP in minimum time is enforced by penalizing the cost at the beginning of the planning horizon ($i = 1$) more than the end of the horizon ($i = H$). Once it reaches the AWP, the corresponding binary variable b_i becomes one. The second term in the cost function, minimizes r_{err} which is the upper-bound of the position error between the current vehicle's position and the AWP. Finally, the last term in the cost function penalizes the input energy with the small coefficient α (around 0.0001) to prevent high frequency artifacts in the control input.

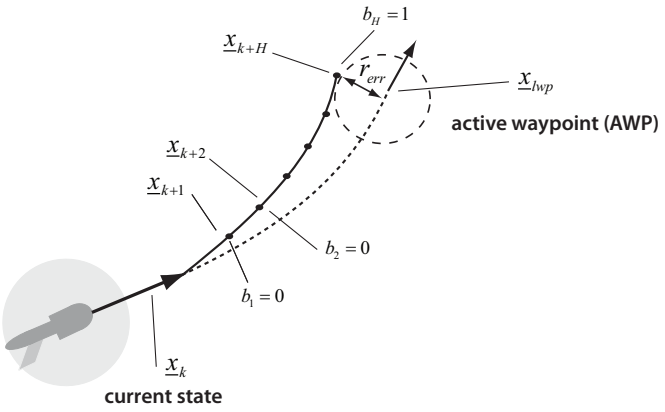


Figure 6.7: Illustration of on-line planning process using MILP.

The vehicle dynamics for on-line trajectory planning are approximated by a linear discrete-time system. This model represents the closed-loop dynamics of the rotorcraft under the velocity controller

$$\mathbf{x}(k+1) = A_d\mathbf{x}(k) + B_d\mathbf{u}(k) \tag{6.30}$$

The state vector consists of position, velocity and acceleration in each direction and the inputs of the system are the control system velocity commands

$$\begin{aligned} \mathbf{x} &= [x,\, v_x,\, a_x,\, y,\, v_y,\, a_y, z,\, v_z]^T \\ \mathbf{u} &= [v_{x_{cmd}},\, v_{y_{cmd}},\, v_{z_{cmd}}]^T \end{aligned} \tag{6.31}$$

In addition, we impose the constraints that dictates the closed-loop maximum applied

velocity commands as well as maximum achievable acceleration

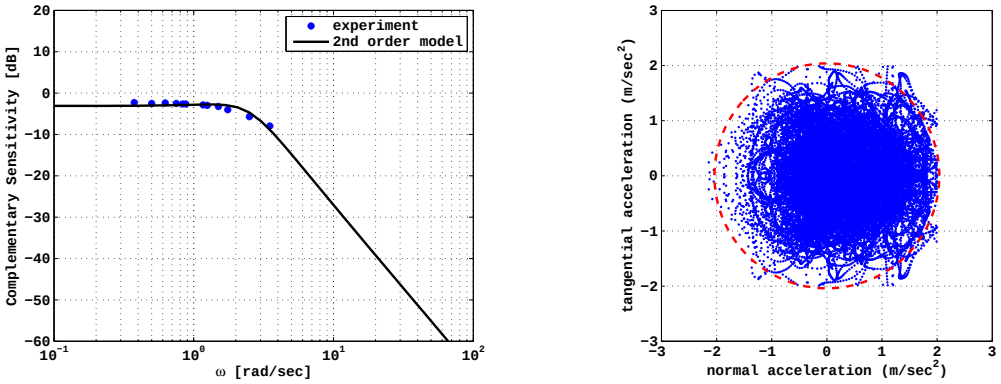
$$\ell_a([v_x(i)_{cmd},\, v_y(i)_{cmd}]) \;\leq\; v_{cmd_{max}} \tag{6.32}$$

$$\ell_a([a_x(i),\, a_y(i)]) \;\leq\; a_{max} \tag{6.33}$$

$$v_{z_{min}} \leq v_z(i) \;\leq\; v_{z_{max}} \tag{6.34}$$

The MILP problem is formulated in AMPL [138] and is solved by a Branch and Bound algorithm provided by the CPLEX software [139] which is interfaced with AMPL.

The nonlinear tracking performance of the closed-loop system under the velocity control can be characterized experimentally by varying the reference velocity magnitude and frequency over the entire curvature-velocity envelope as discussed in Section 4.3.3. The singular value of the Blade-CX2 helicopter closed-loop transfer function across frequencies are shown in Figure 6.8-a. The closed-loop linear model in equation (6.30) is determined by fitting a transfer function to the experimental data in Figure 6.8-a. In addition, the maximum acceleration a_{max} in equation (6.33) is obtained by recording the normal and tangential acceleration of the vehicle under closed-loop control. Figure 6.8-b shows the experimental data and the corresponding upper-bound.



(a) Fitting the linear model of (6.30) to the singular value of the closed-loop transfer function across frequencies obtained from experiment.

(b) Normal and tangential acceleration characteristic of the Blade-CX2 helicopter in closed-loop and the corresponding upper-bound.

Figure 6.8: Characterization of the helicopter closed-loop model for RH optimization.

6.7 Tracking Error Prediction

The tracking performance of the Blade-Cx2 inner-loop control system was experimentally validated across its flight envelope by tracking circular trajectories with different

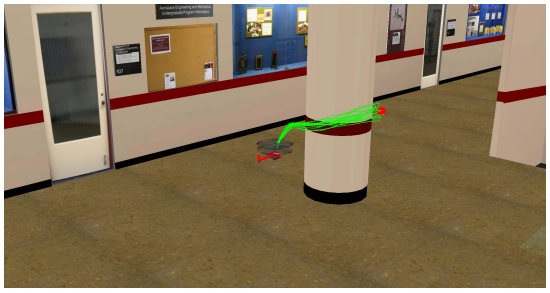
speeds and normal accelerations as explained in Section 4.3.4. The experimental trajectories were shown in Figure 4.17. To characterize the tracking performance, the flight data was used to generate histograms of the relative tracking error (see Figure 4.18). To ensure safe operation when operating amidst obstacles, it is critical to be able to predict the tracking error bound. In Section 4.3.5 we developed a tracking error model that can be used in real-time to predict the tracking error bound based on a pre-specified path generated by the RH path planner. The error model is based on a simplified first-order model of the inner-loop. The effects of modeling uncertainties and disturbances are described by noise parameters η_x, η_y and η_z. The 3D extension of the differential equation governing the path tracking dynamics are as follows

$$\begin{aligned}
\dot{u} &= \dot{u}_{des} + K_{p_x} e_x + K_{d_x} \dot{e}_x + \eta_x && (6.35)\\
\dot{v} &= \dot{v}_{des} + K_{p_y} e_y + K_{d_y} \dot{e}_y + \eta_y y && (6.36)\\
\dot{v} &= \dot{w}_{des} + K_{p_z} e_z + K_{d_z} \dot{e}_x + \eta_z \\
\dot{x} &= u \\
\dot{y} &= v \\
\dot{z} &= w
\end{aligned}$$

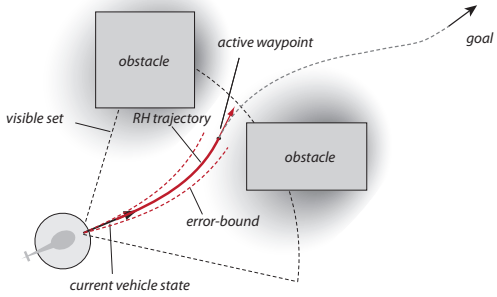
The noise parameters η_x, η_y and η_z are zero-mean Gaussian random variables. Their covariance are adjusted so that the error response predicted by the model have the same statistical behavior as the flight test results obtained from the circle experiments over the dynamics flight envelope of the helicopter as illustrated in Figure 4.22. The gains K_p and K_d are the path following controller's PD gains. The driving factors in the path following dynamics are the desired tangential acceleration (first term in the right hand side) and the normal tracking error (second and third terms). The tracking error bounds are determined from a Monte-Carlo simulation of equation set (6.35) by sampling the random variables η_x, η_y and η_z. Finally, the vector $[\dot{u}_{des},\, \dot{v}_{des}\, \dot{w}_{des}]$ is computed using numerical differentiation of desired velocity vector (tangent to the path) given the path geometry. Figure 6.9 illustrates example of Monte-Carlo trajectories (green color) to the AWP (red ball) in the simulation environment. Given $\{q(x_k, y_k, z_k) : k = 1, \ldots, M\}$, the set of all the points in the Monte-Carlo trajectories, the nominal RH trajectory is considered to be safe if

$$prob(q(x_k, y_k, z_k)) < \alpha \qquad (6.37)$$

where α is the threshold probability for the free space ($\alpha \cong 0.3$).



(a) Monte-Carlo simulation of trajectories to the AWP (red ball) in the simulation environment. The trajectories describe the path following error.



(b) Accounting for the tracking error bound allows to control the trade-off between safety and performance.

Figure 6.9: Tracking error prediction.

6.8 Horizon Length for RH Optimization

The RH optimization part of the guidance system repeatedly solves a *Two-Point Boundary Value* (TPBV) problem with *open final time* subject to the vehicle closed-loop dynamics constraint as discussed in Section 6.6. The MILP minimum-time formulation is a variable horizon scheme depending on how far the AWP is located with respect to the vehicle.

The horizon length H should be large enough to allow the system's trajectory to reach the AWP. Once it reaches the AWP, the arrival binary variable becomes one and the corresponding minimum time and optimal trajectory will be determined. Ideally, one would like to have a very large horizon length H so that the system's trajectory can always reach the AWP. However, a larger horizon length requires longer computation time. From the other hand, too small horizon lengths prevent the trajectory to reach the AWP. This represents a trade-off between the computation load and the resolution of the trajectory. In the remainder of this section, a lower-bound for the planning horizon is determined analytically.

In general it is hard or even impossible to obtain analytical solution for TPBV problems [140]. However, for some dynamic systems it is possible to use approximate dynamics to obtain analytical solution. For the rotorcraft evolving over a finite-time horizon, we can approximate its dynamics as a uniformly accelerated point mass model. The open final time TPBV problem for the mass-point helicopter model under constant

nominal acceleration a_{nom} is given by

$$\min J = t_f \tag{6.38}$$

$$subject\ to$$

$$\begin{aligned} \dot{u} &= a_{nom}cos(\theta) \\ \dot{v} &= a_{nom}sin(\theta) \\ \dot{x} &= u \\ \dot{y} &= v \end{aligned} \tag{6.39}$$

$$with$$

$$\begin{aligned} u(0) = V_o cos(\gamma_o), \quad x(0) &= x_o, \quad x(t_f) = x_f \\ v(0) = V_o sin(\gamma_o), \quad y(0) &= y_o, \quad y(t_f) = y_f \end{aligned} \tag{6.40}$$

The Hamiltonian function $H(t)$ is then

$$H(t) = \lambda_u a_{nom} cos(\theta) + \lambda_v a_{nom} sin(\theta) + \lambda_x u + \lambda_y v \tag{6.41}$$

and the augmented cost function is

$$\Phi = t_f + \eta_x(x(t_f) - x_f) + \eta_y(y(t_f) - y_t) \tag{6.42}$$

According to Euler-Lagrange necessary condition for optimality we must have

$$\begin{aligned} \dot{\underline{\lambda}} &= -H_{\underline{x}} \\ \underline{\lambda}^T(t_f) &= \Phi_{\underline{x}} \end{aligned} \tag{6.43}$$

where $\underline{x}$ and $\underline{\lambda}$ are state vector and Lagrange multiplier vector, respectively. This leads to

$$\begin{aligned} \dot{\lambda}_u &= -\lambda_x, \quad \lambda_u(t_f) = 0 \\ \dot{\lambda}_v &= -\lambda_y, \quad \lambda_v(t_f) = 0 \\ \dot{\lambda}_x &= 0, \quad \lambda_x(t_f) = \eta_x \\ \dot{\lambda}_y &= 0, \quad \lambda_y(t_f) = \eta_y \end{aligned} \tag{6.44}$$

In addition, the derivative of the Hamiltonian with respect to the input must be zero

along the optimal trajectory (i.e., $H_\theta(t) = 0$), which yields

$$\lambda_v a_{nom} cos(\theta) + \lambda_u a_{nom} sin(\theta) = 0 \Rightarrow tan(\theta) = \frac{\lambda_u}{\lambda_v} \tag{6.45}$$

After solving (6.44) and combining with (6.45), we get

$$\theta_{opt} = tan^{-1} \frac{\eta_x}{\eta_y} = const. \tag{6.46}$$

Since according to the above results, the optimal input θ_{opt} is constant along the optimal trajectory, we can integrate the equations of motion and obtain the following trajectories

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = a_{nom} \begin{bmatrix} cos(\theta_{opt}) \\ sin(\theta_{opt}) \end{bmatrix} t^2/2 + V_o \begin{bmatrix} cos(\gamma_o) \\ sin(\gamma_o) \end{bmatrix} t + \begin{bmatrix} x_o \\ y_o \end{bmatrix} \tag{6.47}$$

Combining equation (6.47) with (6.46) and after performing some manipulations, it can be shown that the minimum final time t_f is the smallest positive real root of the following polynomial equation

$$\begin{aligned} t_f^4 a_{nom}^2 - 4t_f^2 V_o^2 + 8t_f V_o(y_f sin\gamma_o + x_f cos\gamma_o) - 8t_f V_o(y_o sin\gamma_o + x_o cos\gamma_o) \\ -4(x_f^2 + y_f^2) - 4(x_o^2 + y_o^2) + 8x_f x_o + 8y_f y_o = 0 \end{aligned} \tag{6.48}$$

and the optimal input is given by

$$\theta_{opt} = tan^{-1} \frac{y_f - V_o sin(\gamma_o) t_f - y_o}{x_f - V_o cos(\gamma_o) t_f - x_o} \tag{6.49}$$

Using this model, we can easily compute a lower bound on the cost from the current vehicle state to the AWP. This *cost-to-come* is given by the value of the optimal cost (6.48) in the absence of obstacles. Notice that the mass-point model used in this analysis is free of state/input constraints. Therefore, the *cost-to-come* in the presence of obstacles and state/input constraints can not be lower than the obstacle-free cost, since the presence of obstacles and constraint translates into additional cost [55].

In addition, we need to take into account the time to climb/descend to the AWP located at altitude z_f when operating in a 3D environment. It is reasonable to assume that the vehicle flies in vertical direction at a constant nominal climb rate w_{nom}. As a result, the time to climb/descend is given by

$$t_v = \frac{z_f - z_o}{w_{nom}} \tag{6.50}$$

where z_o in the starting altitude. Finally, the total *cost-to-come* t_{ctc} is

$$t_{ctc} = \max\{t_f, t_v\} \tag{6.51}$$

The minimum horizon length for the RH optimization can be computed from the lower bound on the *cost-to-come* via $H = t_{ctc}/T$ where T is the planning sampling time.

6.9 Active Waypoint (AWP) Selection

The RH-CTG framework enables to efficiently compute a control action while taking into account the global elements of the planning problem [90, 141]. The control action is determined by running the RH optimization problem to reach a AWP in the vehicle's reachable set. Global planning considerations are accomplished via the spatial value functions (SVF). The AWP acts as the interface between the RH planner and the SVF. It defines the domain over which the RH planner performs its optimization and therefore is an essential component to the planning performance. It also defines the process that allows to assimilate the sensory data from on-board sensors. The SVF provides the computational gateway for situational awareness of the immediate environment. The SVF that accurately reflect the effect of the immediate environment on the value of possible actions allows to properly select the AWP. Hence, proper selection of AWP plays a key role in enabling safe and adaptive operation.

Some AWPs in the reachable set result in a safe but conservative trajectories while other AWP candidates may result in more time-optimal but unsafe trajectories. Therefore, it is necessary to use a cost function for AWP selection that takes into account the trade-off between performance and safety. The strategy that we use here to select AWP is based on stochastic dynamic programming equation in Section 6.1 for finite-state vehicle models. We assume deterministic motion primitives therefore there is no uncertainty in control action for the finite-state vehicle model, i.e., $p(s_{t+1}|s_t, u_t) = 1$. The uncertainties in control action associated with the inner-loop control system is considered in the form of bounds on tracking error as discussed in Section 6.7. Algorithm 4 summarizes the AWP selection procedure. The AWP is determined during each planning cycle based on information about the immediate environment and cost-to-go function that combines the sensory updates and the longer-term, global planning problem.

Algorithm 4: Active Waypoint (AWP) Selection

Input: vehicle pose, motion primitives, occupancy probability, constant $\gamma \in [0\ 1]$.

Output: Active Waypoint (AWP).

Initialize r = current vehicle pose.

while $r \in Visible\,Set$ **do**

  forall *cell* $s_i \in successor(r)$ **do**

    Compute transition cost from r to s_i, $C(s_i, r)$ based on MPs.

    Compute the total cost $J_i = p(\,q(s_i)\,)\,\{\,C(s_i, r) + \gamma\, CTG(s_i)\,\}$.

  Pick the successor $s_i^\star$ with the minimum cost J_i.

  $r \leftarrow s_i^\star$.

AWP = r.

6.10 Reactive Planner

Reactive planning refers to the methodologies of changing the robot's path to overcome unexpected obstacles. The exteroceptive sensor carried by the rotorcraft cannot reach far enough, or behind obstacles, to account for all a-priori unknown information within the planning process. In our guidance framework the reactive planner is tightly integrated with the global planner to provide higher level of safety as well as a last resort solution for collision avoidance. The reactive planning module is engaged if 1) A safe AWP cannot be found within the visible set or 2) the trajectory error prediction process labels the RH path as unsafe.

The reactive planner in our guidance framework seeks the safest free space in the rotorcraft's vicinity for obstacle avoidance. We use *Vector Field Histogram* (VHF) approach [142] to find the safest region around the vehicle's momentary location. In VHF, a moving window of size $w_s \times w_s$ cells accompanies the vehicle at each time (called the active region). For each cell (i, j) in the active region, the direction $\beta_{i,j}$ and magnitude $m_{i,j}$ are defined as

$$\begin{aligned} \beta_{i,j} &= tan^{-1}\frac{y_j - y_r}{x_i - x_r} \\ m_{i,j} &= p_{i,j}\,(a - b\,d_{i,j}) \end{aligned} \tag{6.52}$$

where (x_i, y_j) is the location of the cell (i, j), (x_r, y_r) is robot location, $d_{i,j}$ is the distance between the robot and the cell (i, j), a and b are positive constants and finally $p_{i,j}$ is the occupancy probability of the cell. Since we implemented the map update process,

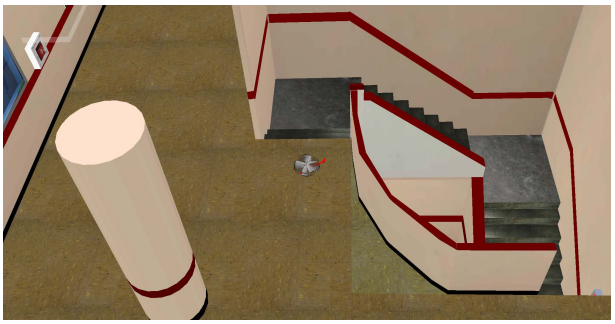
which provides the occupancy probability of each cell in the environment, we use the obstacle representation based on occupancy probability instead of histogram grid in the original formulation of VHF[142].

The next step is to construct the polar histogram which has resolution of α degrees and consists of n sectors such that $n = 360^\circ/\alpha$. For each sector of the histogram k, the polar obstacle density h_k is

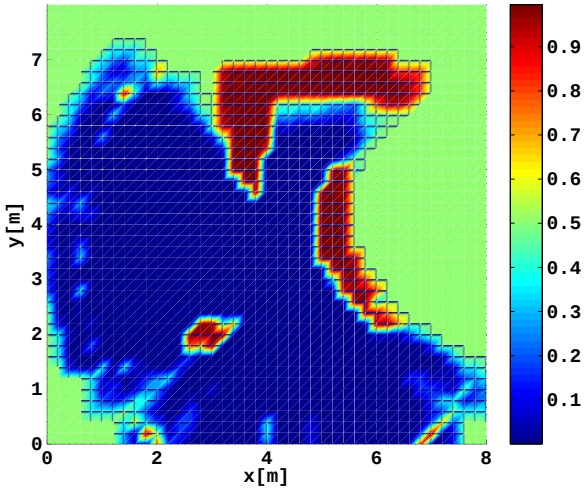
$$h_k = \sum_{i,j} m_{i,j} \tag{6.53}$$

The resulting polar histogram is checked to select the most suitable sector among all sectors with sufficiently low polar obstacle density. The AWP is then placed along that direction.

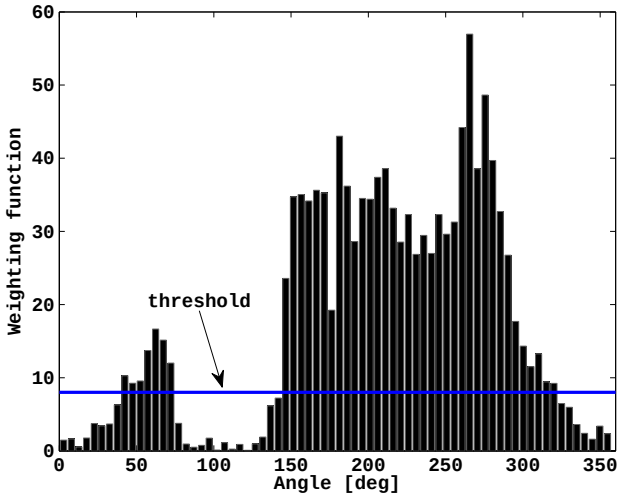
Figure 6.10-a shows the obstacle course in the helicopter's immediate environment. Figure 6.10-b is the occupancy probability of cells in the active window. Figure 6.10-c shows the polar histogram indicating the directions that have a polar obstacles density below a certain threshold. Once the safest horizontal direction for travel was selected, the same procedure is done to obtain the safest vertical direction by forming the polar histogram of the plane in the best horizontal direction.



(a) Obstacle course in the helicopter immediate environment.



(b) Corresponding occupancy probability in the active window.



(c) The resulting polar histogram which is used to find a sector with sufficiently low polar obstacle density.

Figure 6.10: Example of polar histogram in VFH method which determines the candidate directions for free space.

6.11 Overall Guidance System Integration

The integration of the main subsystems of *Sensory Predictive Guidance* was previously shown as the block-diagram of Figure 1.7. Algorithm 5 gives a functional overview of the overall guidance system.

Algorithm 5: Functional description of the overall guidance system processes.

Input: Current vehicle state, a-priori CTG map.

Output: Velocity reference commands at each iteration.

while *Not reached to the target* **do**

1 • Implement Algorithm 2 to update the environment map;

2 • Implement Algorithm 3 to update the CTG for the vehicle visible set;

3 • Implement Algorithm 4 to find AWP;

4 • **if** *AWP not found* **then**
    Engage reactive planner to find a safe AWP;

5 • Run the RH optimization from the current vehicle state to AWP to find the reference trajectory (Section 6.6);

6 • Predict the reference trajectory tracking error-bound (Section 6.7) and check for collision;

7 • **if** *reference trajectory not safe* **then**
    Engage reactive planner to find a safe AWP;
    Goto line 5;

8 • Engage the path following controller to follow the reference path until the next planning cycle;

6.12 Computation Time

Even though tactical UAV helicopters carry powerful on-board computers, the limited on-board computation power has been always a challenge in UAV guidance problems. In this section we characterize the computation time for the main modules in our guidance framework by recording the real-time taken for each module in each iteration during the execution. The analysis was done on a desktop computer with the following specifications:

```
Simulation Computer:
CPU: Intel core (TM) i7 920
```

```
clock rate: 2668 MHz
RAM: 3 GB
Cache size: 8192 KB
Cores: 8
Operating system: Linux 2.6.32-36 generic i686
```

We are only interested in the CPU computation time for the main time consuming processes in our guidance framework which are *Map-update* (Algorithm 2), *CTG-update* (Algorithm 3) and *RH-planner* (Section 6.6) as the rest of the processes have relatively much smaller CPU computation time.

Figure 6.11 shows the CPU usage for the *CTG-update* process as a function of number of newly seen terrain elements that are passed to the cost-to-go update process. As it can be seen, the number of terrain elements is not a factor in computation cost by itself. In other words depending on the relative location of new terrain elements in the map the computation time might vary.

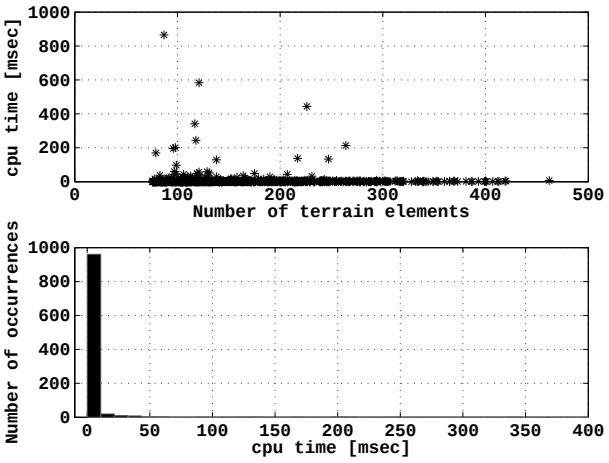


Figure 6.11: CPU computation time for CTG update process, top plot shows the variation on CPU-time w.r.t number of terrain element, bottom plot shows the variation of CPU-time in a histogram.

Figure 6.12 shows the CPU usage for the *Map-update* process as a function of number of newly seen terrain elements. As it is expected, the computation time is proportional to the number of terrain elements. Finally, Figure 6.13 shows the computation time to run the RH optimization problem using MILP in each planning cycle. Therefore, according to our analysis, the RH optimization problem takes the longest CPU time among all other processes. However, it runs at a slower rate ($\sim$ 0.5 to 1 Hz). In

addition, all these processes run in a parallel distributed way and the execution of each process does not depend on the termination of the other ones.

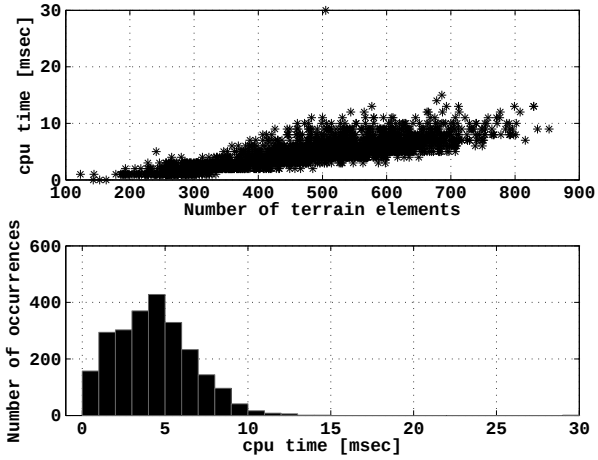


Figure 6.12: CPU computation time for occupancy map process, top plot shows the variation on CPU-time w.r.t number of terrain element, bottom plot shows the variation of CPU-time in a histogram.

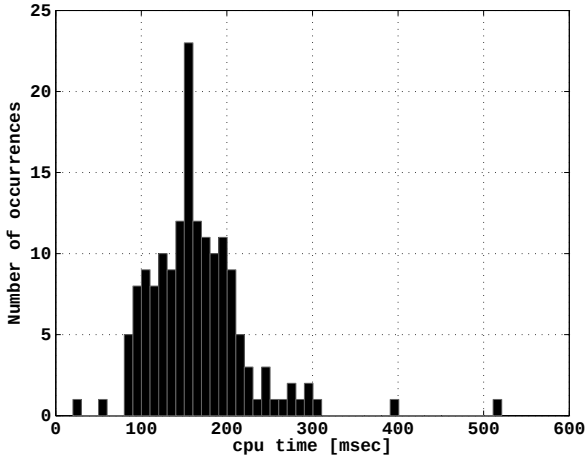


Figure 6.13: CPU computation time for the RH process which runs the MILP optimization problem in each planning cycle.

6.13 Simulation Results

As mentioned in Section 2.2.2 the 3D simulation is built around RIPTIDE [91] and our own custom C++ processes. Figure 2.7 shows the functional block diagram of the whole simulation environment.

Figure 6.14 show snapshots of the simulation scenario in this paper where the helicopter starts its autonomous flight from the basement of the Akerman Hall move toward the goal in the hangar area in the first floor. The first column in Figure 6.14 shows the outside view, the second column shows the mapping process, the third column shows the laser scanner scanning the surrounding environment and finally the forth column is the on-board view. Table 6.1 shows the key parameters of the guidance system that was used for the simulation.

Table 6.1: Key parameters of the guidance system used for the simulations.

Parameter	Value
Sensory range of onboard sensor	4 [m]
RH update rate	2 [Hz]
Inner-loop control system update rate	20 [Hz]
Max horizontal velocity	0.45 [m/s]
Max horizontal acceleration	2 [m/s^2]
Max vertical velocity	0.3 [m/s]
Environment grid size	0.2 [m]
Probability threshold for occupancy	0.7
Risk map update coefficients (eq. 6.17)	$c_1 = 0.2$, $c_2 = 0.2$, $\sigma = 0.4$

To evaluate the performance of the proposed guidance system we performed successive simulations of the same scenario. In the first simulation the vehicle has no a-prior knowledge about the environment. In the next simulations the vehicle uses the knowledge about the environment and cost-to-go that were obtained from the previous simulations. Figure 6.15 shows the plot of the 3D trajectories of these simulations. In this figure we also showed the shortest path from the starting point to the goal which was obtained by propagating wave fronts in four quadrants [143].

Figure 6.16-a shows the time-to-go of the trajectory points in each run over the course of the simulation time. Time-to-go is a measure of performance for the guidance system. As it can be seen, in the first simulation, the vehicle detours slightly to a dead-end area due to the limited perception horizon before it finds its way which causes an increase in the overall time-to-go. The data points indicated by asterisks are the corresponding CTG for each simulation which can be served as an upper-bound for the

actual time-to-go. The CTG of shortest path is also plotted in the figure. Since the shortest path does not consider dynamic constraints, it can be served as a lower-bound for the actual time-to-go. Figure 6.16-b shows the histogram of the distance of the trajectory points to the nearest obstacle in the environment as a function of time. The nearest distance is calculated by first converting the 3D environment probability map to a 3D binary image and then computing the Euclidean distance transform for the 3D binary image. The figure shows that the mean distance to the nearest obstacles slightly improves after the first simulation however it remains almost the same for the next simulations. This is due to the fact that the parameters of the risk map update function (equation 6.17) are held constant for all the simulations.

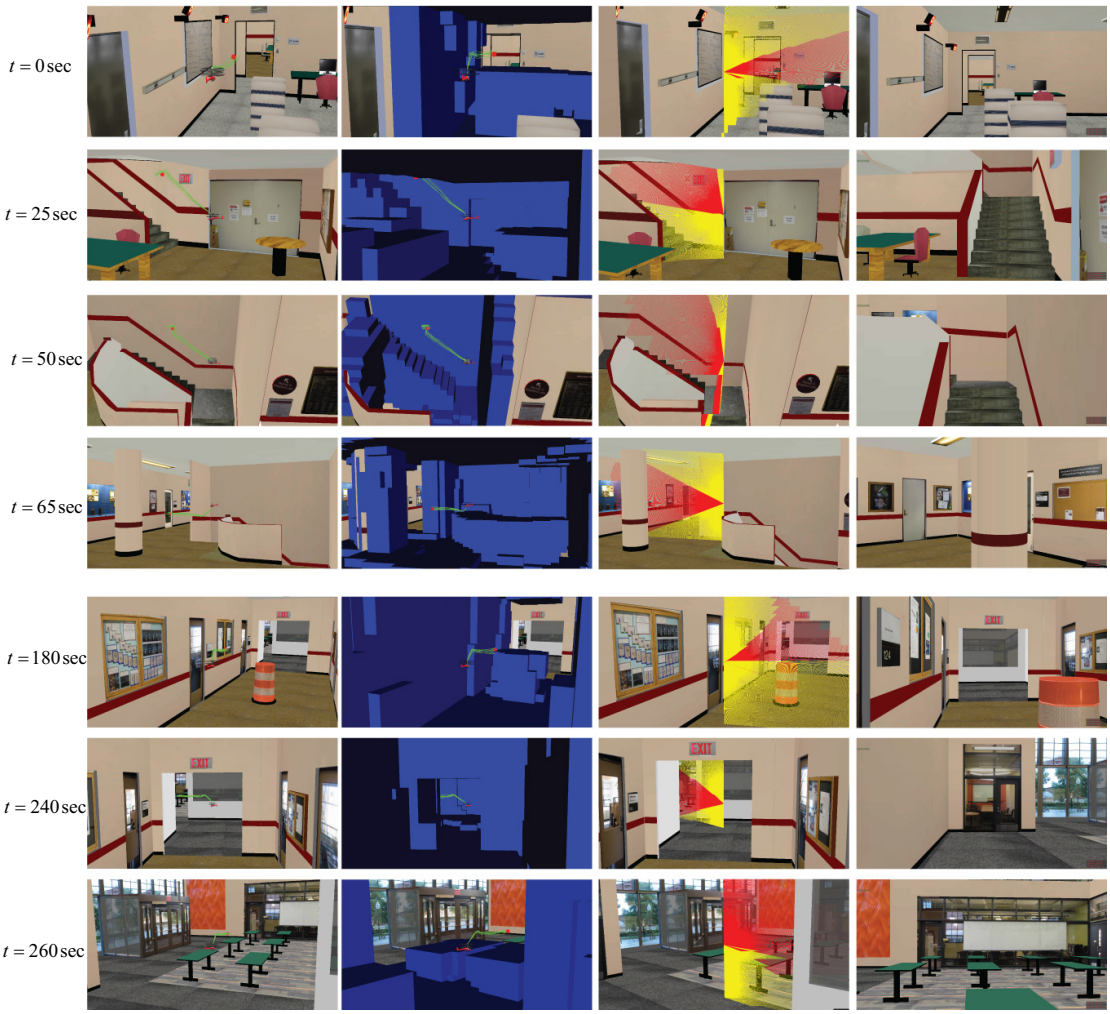


Figure 6.14: Snapshots of the simulation scenario where the helicopter takes off from the basement of Akerman Hall and fly toward the hangar area in the first floor. All the dimensions in the environment model are sub-meter accurate. First column: outside view, second column: mapping process, third column: laser scanning, forth column: on-board view.

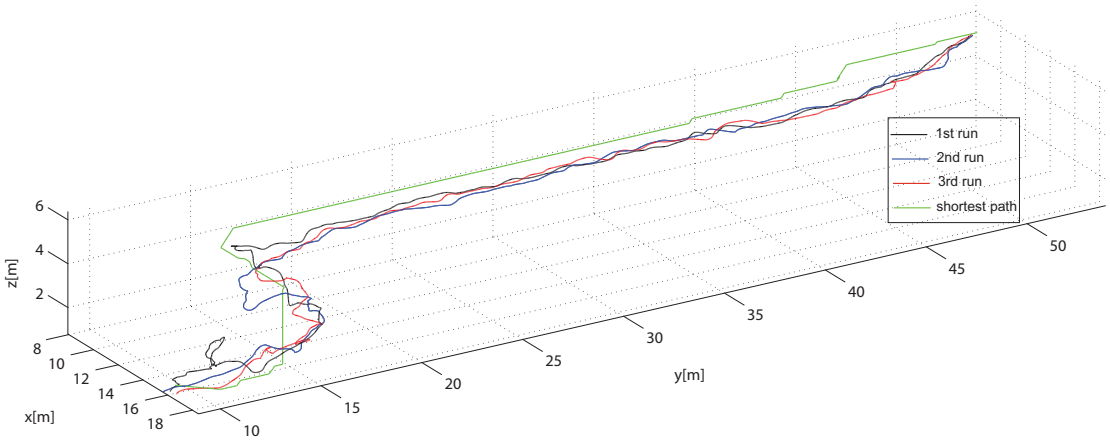
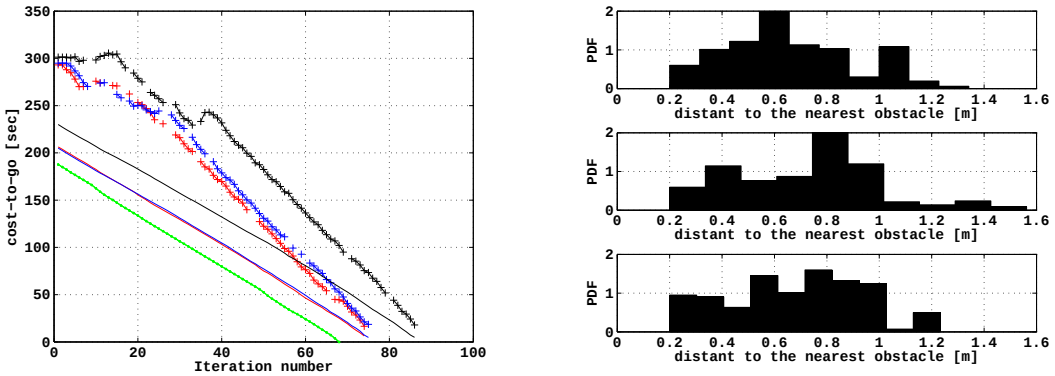


Figure 6.15: Trajectories obtained from several simulations of the same scenario. In the first simulation, no a-priori environmental knowledge is available. In each run the vehicle uses the latest environment knowledge and cost-to-go from the previous simulations. The shortest path from the starting point to the goal which was obtained by propagating wave fronts in four quadrants.

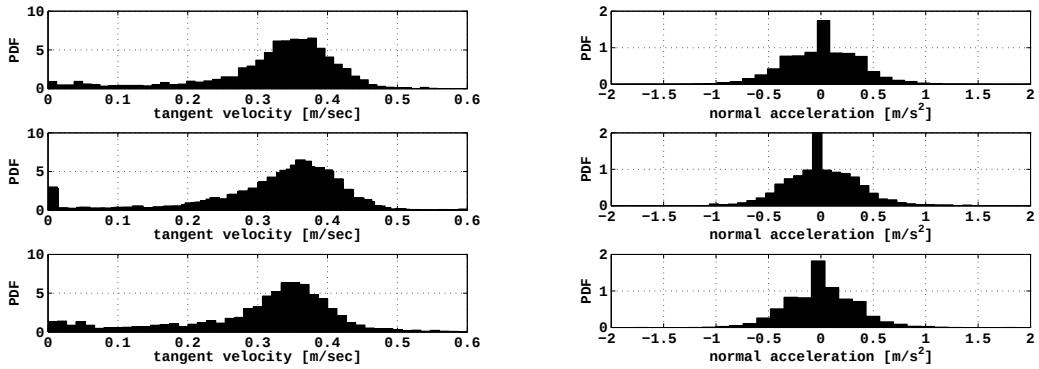


(a) Time-to-go for trajectory points and the corresponding upper-bounds and lower-bound.

(b) Histogram of the distance of the trajectory points to the nearest obstacle in each simulation.

Figure 6.16: Performance and safety analysis of the guidance system: (a) the time-to-go as a measure of performance (b) distance to the nearest obstacle as a measure of safety.

In addition, Figure 6.17 shows the histogram of the tangential velocity and normal acceleration of the trajectories in each simulation. As it can be seen, the tangential velocity and normal acceleration have almost the same distribution. This is due to the fact the the RH optimization parameters for the maximum velocity and maximum acceleration are held constant for all the simulations.



(a) Tangential velocity.

(b) Normal acceleration.

Figure 6.17: Histogram of tangential velocity and normal acceleration in each simulation.

Chapter 7

Conclusions

Through the course of this dissertation we have presented the underlying technologies necessary for integrating an autonomous guidance system for an RUAV to be able to plan a safe and effective path to the destination in indoor and confined environments. The capabilities necessary to perform this system integration include vehicle dynamic modeling, inner-loop control design and performance evaluation, environment mapping, dynamically updating the CTG function as well as reactive obstacle avoidance. Most material in this thesis is around the Blade-CX2 rotorcraft, however, due to the similarity of the dynamics characteristic of the miniature coaxial helicopters to the full-size helicopters, the general approach is relevant and should be valid for most RUAV platforms.

The model of the rotorcraft was identified using the frequency domain identification tool (CIFER) from input-output data that was collected in our indoor flight test facility built around a visual motion tracking system. The parametric model explicitly accounts for the lower rotor dynamics and also accounts for the upper-rotor and the stabilizer-bar dynamics using a lumped, equivalent system. We verified the fidelity of the coupled rotor-fuselage model by comparing the model predicted response with the responses collected during the experiments, both in the time and frequency domain. We also provided an analysis and verification of the identified derivatives based on first principles to demonstrate that the model structure is physically meaningful. We used image sequences from a high frame-rate camera to verify the rotor and stabilizer-bar time-constants. The helicopters low frequency Phugoid modes were verified via piloted experiments. These modes are dominant characteristics in the translational dynamics and can be challenging for velocity control system design. We expect that the model structure will be appropriate for the identification of other small-scale coaxial helicopters.

The control augmentation for Blade-CX2 helicopter was based on a nested attitude-velocity loop control architecture and was designed following classical loop-shaping and dynamic inversion techniques. A path following layer wrapped around the velocity control system which enabled the rotorcraft to follow reference trajectories specified by a sequence of waypoints and velocity vectors. Such path references are common in autonomous guidance algorithms. In the first evaluation step we analyzed the robust performance and robust stability of the closed-loop system using experimental frequency responses data together with parametric modeling uncertainties. These were extracted from the frequency-domain system identification procedure. In a second evaluation step, we described flight-test results based on a novel set of tracking performance criteria that were conceived to capture the type of performance requirements called for by autonomous guidance applications. Tracking error statistics were characterized over the entire flight envelope. These results are used to determine a tracking error model that can be used to predict the tracking error-bound for a pre-specified path.

In the last part of the dissertation we described a complete guidance framework for autonomous operation in a partially known or unknown cluttered environments. The approach assumes an on-board depth sensor is available for real-time environment update. The hierarchical architecture of the sensory-predictive guidance system is motivated by optimal control principles to ensure tight and consistent integration between the planning, control and sensing processes. The system is developed as an extension of the receding horizon scheme using an approximation of the spatial value function as cost-to-go function. The cost-to-go (CTG) function is updating as the new information about the environment are detected. By combining the CTG and occupancy probability map, we presented a novel approach for guiding the vehicle to the goal in cluttered unknown environment while maintaining a trade-off between safety and performance. Finally, to ensure safe path tracking, a path tracking error model was used in real-time to predict the tracking error bound based on a pre-specified path generated by a RH path planner in each planning cycle.

7.1 Future Directions

There are several future directions for this research. In this section we present some ideas for extension of this work to other problems:

- In modeling and control design, it would be interesting to compare the dynamics characteristics of the coaxial helicopter with other miniature indoor rotorcraft in

terms of dynamic agility, payload limitation, cost, and closed-loop performance.

- In this research we assumed that the states of the helicopter as well as its pose are available. Another direction for future research would be to integrate state estimation and localization algorithms (such as SLAM [17]) with the presented guidance system to make a self-contained autonomous system.

- In this research we presented the high fidelity simulation results of the sensory predictive guidance in Akerman Hall at the University of Minnesota. Experimental evaluation of the developed guidance system using onboard exteroceptive sensors such as a laser scanner in Akerman Hall would be another direction for future work.

- Test and validate the developed guidance algorithm for outdoor RUAVs. Due to the similarity of the dynamic characteristics of the Blade-CX2 helicopter to the full-size helicopters, we expect that the general approach in this thesis is relevant and should be valid for most RUAV platforms.

- Improvements can be made to the way the CTG function and the occupancy map are stored in the guidance computer. Techniques such as connected node-graphs could greatly reduce the memory occupied by the planner which is important for operating in large outdoor environments.

- In the RH optimization in this research we did not implicitly consider the uncertainty in the closed-loop dynamics. Integrating the closed-loop uncertainty model in the RH optimization makes the robustness analysis of the guidance system more formal.

- In this research we considered deterministic motion primitives for finite-state approximation of the vehicle dynamics. Another direction for future research would be considering uncertainty in control actions for the motion primitives for the CTG function computation.

- One of the main challenges faced during the flight tests was the limitation of the space covered by the motion tracking cameras. For further experiments involving modeling small-scale RUAVs, control laws design and 3D guidance a larger lab space is desired. This also reduces the risk of collision with nearby objects during the development phase of the algorithms (which damages the helicopter and the on-board sensors/electronics).

References

[1] K. S. Pratt. Analysis of VTOL MAV Use During Rescue and Recovery Operations Following Hurricane Katrina. Master's thesis, Department of Computer Science and Engineering, University of South Florida, USA, 2007.

[2] J. Andersh and B. Mettler. System integration of a miniature rotorcraft for aerial tele-operation research. *Mechatronics*, 21(5):776–788, 2011.

[3] D. H. Halley. ABC helicopter stability, control and vibration evaluation on the princeton dynamic model track. Washington, D.C., May 1973. Proceedings of the 29th American Helicopter Society Forum.

[4] *Unmanned Aircraft Systems Roadmap 2010-2035*. U.S. Army UAS Center of Excellence, 2010.

[5] E. N. Johnson and D. P. Schrage. The Georgia Tech Unmanned Aerial Research Vehicle: GTMax. Proceedings of the AIAA Guidance, Navigation, and Control Conference, 2003.

[6] O. Amidi, T. Kanade, and J. R. Miller. Autonomous Helicopter Research at Carnegie Mellon Robotics Institute. Proceedings of Heli Japan '98, 1998.

[7] E-flite Advancing Electric Flight. `http://www.e-fliterc.com`.

[8] Esky Helicopters. `http://www.esky-heli.com`.

[9] AR Drone Quadrotor. `http://ar-drone.org`.

[10] Tech Model Products. `http://www.tech-mp.com`.

[11] K. Celik, S.-J. Chung, M. Clausman, and A. K. Somani. Monocular Vision SLAM for Indoor Aerial Vehicles. St. Louis, MO, October 2009. IEEE Conference on Intelligent Robots and Systems (IROS).

[12] A. R. S. Bramwell, G. Done, and D. Balmford. *Helicopter Dynamics*. AIAA and Butterworth-Heinemann, 2001.

[13] G. D. Padfield. *Helicopter Flight Dynamics: The Theory and and Application of Flying Qualities and Simulation Modeling*. AIAA Education Series, 1996.

[14] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin. Quadrotor Helicopter Flight Dynamics and Control: Theory and Experiment. Hilton Head, South Carolina, 2007. AIAA Guidance, Navigation and Control Conference and Exhibit.

[15] M. Valenti, B. Bethke, D. Dale, A. Frank, J. McGrew, S. Ahrens, J. P. How, and J. Vian. The MIT Indoor Multi-Vehicle Flight Testbed. Roma, Italy, 2007. IEEE International Conference on Robotics and Automation.

[16] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar. The GRASP Multiple Micro-UAV Testbed. *IEEE Robotics and Automation Magazine*, 17(3):56–65, 2010.

[17] A. Bachrach, R. He, and N. Roy. Autonomous Flight in Unknown Indoor Environments. *International Journal of Micro Air Vehicles*, 1(4):217–228, 2009.

[18] Kamov coaxial helicopter. `http://en.wikipedia.org/wiki/Ka-50`.

[19] Gyrodyne coaxial helicopter. `http://www.gyrodynehelicopters.com`.

[20] R. V. Jategaonkar. *Flight Vehicle System Identification, A time domain methodology*. American Institute of Aeronautics and Astronautics, 2006.

[21] M. B. Tischler and R. K. Remple. *Aircraft and Rotorcraft System Identification, Engineering Methods with Flight Test Examples*. AIAA Education Series, 2006.

[22] V. Gavrilets, B. Mettler, and E. Feron. Nonlinear Model for a Small-Size Acrobatic Helicopter. Number AIAA 2001-4333, Montreal, Canada, August 2001. Proceedings of the American Institute of Aeronautics Guidance, Navigation, and Control Conference.

[23] B. Mettler, M.B. Tischler, and T. Kanade. System Identification Modeling of a Small-Scale Unmanned Rotorcraft for Control Design. *Journal of the American Helicopter Society*, 47(1):50–63, January 2002.

[24] V. Klein and E. A. Morelli. *Aircraft System Identification, Theory and Practice*. AIAA Education Series, 2006.

[25] M. La Civita, G. Papageorgiou, W. C. Messner, and T. Kanade. Design and Flight Testing of an H_∞ Controller for a Robotic Helicopter. *Journal of Guidance, Control, and Dynamics*, 29(2), March 2006.

[26] N. Dadkhah, V. R. Korukanti, Z. Kong, and B. Mettler. Experimental Demonstration of an Online Trajectory Optimization Scheme using Approximate Spatial Value Functions. Shanghai, China, 2009. 48th IEEE Conference on Decision and Control.

[27] J. Andersh, B. Mettler, and N. Papanikolopoulos. Miniature Embedded Rotorcraft Platform for Aerial Teleoperation Experiments. Thessaloniki, Greece, 2009. 17th Mediterranean Conference on Control and Automation.

[28] D. J. Halaas, S. R. Bieniawski, P. Pigg, and J. Vian. Control and Management of an Indoor, Health Enabled, Heterogeneous Fleet. Washington, Seattle, 2009. AIAA Infotech Aerospace Conference and AIAA Unmanned Unlimited Conference.

[29] J. A. Grauer, J. K. Conroy, J. E. Hubbard, and D. J. Pines. System Identification of a Miniature Helicopter. Honolulu, Hawaii, 2008. AIAA Atmospheric Flight Mechanics Conference and Exhibit.

[30] D. Schafroth, C. Bermes, S. Bouabdallah, and R. Siegwart. Modeling, System Identification and Robust Control of a Coaxial Micro Helicopter. *Control Engineering Practice*, 18(7):700–711, 2010.

[31] B. Michini and J. P. How. Adaptive Control for Indoor Autonomous Vehicles: Design Process and Flight Testing . Chicago, Illinois, 2009. AIAA Guidance, Navigation and Control Conference and Exhibit.

[32] M. D. Takahashi, G. Schulein, and M. Walley. Flight Control Law Design and Development for an Autonomous Rotorcraft. Proceedings of the 64th Forum of American Helicopter Society, May 2008.

[33] E.N. Johnson and S.K. Kannan. Adaptive Flight Control for an Autonomous Unmanned Helicopter. In *AIAA Guidance, Navigation and Control Conference*, Monterey, California, August 2002.

[34] H.J. Kim and D.H. Shim. A Flight Control System for Aerial Robots: Algorithms and Experiments. *Control engineering practice*, 11(12):1389–1400, 2003.

[35] B. Mettler, M.B. Tischler, T. Kanade, and W. Messner. Attitude Control Optimization for a Small-Scale Unmanned Helicopter. Denver, CO, August 2000. AIAA Guidance, Navigation and Control Conference and Exhibit.

[36] M.B. Tischler, J.D. Colbourne, M.R. Morel, D.J. Biezad, K.K Cheung, W.S. Levine, and V. Moldoveanu. A Multidisciplinary Flight Control Development Environment and Its Application to a Helicopter. *IEEE Control Systems Magazine*, 19(4):22–33, August 1999.

[37] V. Gavrilets, I. Martinos, B. Mettler, and E. Feron. Control Logic for Automated Aerobatic Flight of Miniature Helicopter. Monterey, CA, Aug. 2002. Proceedings of the AIAA Guidance, Navigation and Control Conference.

[38] D. Enns and T. Keviczky. Dynamic Inversion Based Flight Control for Autonomous RMAX Helicopter. Minneapolis, Minnesota, June 2006. American Control Conference.

[39] António Pedro Aguiar and João Pedro Hespanha. Trajectory-Tracking and Path-Following of Underactuated Autonomous Vehicles with Parametric Modeling Uncertainty. *IEEE Trans. on Automat. Control*, 52(8):1362–1379, Aug. 2007.

[40] P. Encarnaçao and A. Pascoal. Combined Trajectory Tracking and Path Following: An Application to the Coordinated Control of Autonomous Marine Craft. In *IEEE Conference on Decision and Control*, volume 1, pages 964–969, 2001.

[41] António P. Aguiar, Dragan B. Dačić, João Pedro Hespanha, and Petar Kokotović. Path-Following or Reference-Tracking? An Answer Based on Limits of Performance. In *Proc. of the 5th IFAC Symp. on Intelligent Autonomous Vehicles*, July 2004.

[42] António Pedro Aguiar, João Pedro Hespanha, and Petar V. Kokotović. Path-Following for Non-Minimum Phase Systems Removes Performance Limitations. *IEEE Trans. on Automatic Control*, 50(2):234–239, Feb. 2005.

[43] H.J. Kim, D.H. Shim, and S. Sastry. Nonlinear Model Predictive Tracking Control for Rotorcraft-based Unmanned Aerial Vehicles. In *American Control Conference*, volume 5, pages 3576–3581, 2002.

[44] I. Kaminer, A. Pascoal, E. Hallberg, and C. Silvestre. Trajectory Tracking for Autonomous Vehicles: An Integrated Approach to Guidance and Control. *Journal of Guidance, Control, and Dynamics*, 21(1):29–38, 1998.

[45] I. A. Raptis and K. P. Valavanis. *Linear and Nonlinear Control of Small-Scale Unmanned Helicopters*. International Series on Intelligent Systems, Control and Automation: Science and Engineering, Volume 45, Spriner, 2011.

[46] I.A. Raptis, K.P. Valavanis, and W.A. Moreno. A Novel Nonlinear Backstepping Controller Design for Helicopters Using the Rotation Matrix. *Control Systems Technology, IEEE Transactions on*, 19(2):465–473, March 2011.

[47] E. Frazzoli, M.A. Dahleh, and E. Feron. Trajectory Tracking Control Design for Autonomous Helicopters Using a Backstepping Algorithm. In *American Control Conference*, volume 6, pages 4102–4107, 2000.

[48] G.M. Hoffmann, S.L. Waslander, and C.J. Tomlin. Quadrotor Helicopter Trajectory Tracking Control. In *AIAA Guidance, Navigation, and Control Conference*, Honolulu, Hawaii, August 2008.

[49] G. Conte, S. Duranti, and T. Merz. Dynamic 3D Path Following for an Autonomous Helicopter. In *Proc. of the IFAC Symp. on Intelligent Autonomous Vehicles*, 2004.

[50] R. Cunha, D. Antunes, P. Gomes, and C. Silvestre. A Path-Following Preview Controller for Autonomous Air Vehicles. In *AIAA Guidance, Navigation, and Control Conference*, Keystone, Colorado, August 2006.

[51] C. Boutilier, T. Dean, and S. Hanks. Decision-theoretic Planning: Structural Assumptions and Computational Leverage. *Journal of Artificial Intelligence Research*, 11:1–64, 1999.

[52] J. Blythe. Decision-theoretic Planning. *AI Magazine*, 20(2):37–54, 1999.

[53] N. Dadkhah and B. Mettler. Survey of Motion Planning Literature in the Presence of Uncertainty: Considerations for UAV Guidance. *Journal of Intelligent and Robotic Systems*, 65:233–246, 2012.

[54] B. Weiß, M. Naderhirn, and L. del Re. Global Real-time Path Planning for UAVs in Uncertain Environment. pages 2725–2730, Munich, Germany, Oct 2006. IEEE

International Symposium on Intelligent Control and International Conference on Control Applications.

[55] E. Frazzoli. Robust Hybrid Control for Autonomous Vehicle Motion Planning, PhD Thesis, MIT, June 2001.

[56] L. Kavraki, P. Svestka, J. Latombe, and M. Overmars. Probabilistic Roadmaps for Path Planning in High Dimensional Configuration Spaces. *IEEE Transaction on Robotics and Automation*, 12(4):566–580, 1996.

[57] S.M. LaValle and J.J. Kuffner Jr. Randomized Kinodynamic Planning. *International Journal of Robotics Research*, 20(5):378, 2001.

[58] P.O. Pettersson and P. Doherty. Probabilistic Roadmap Based Path Planning for an Autonomous Unmanned Aerial Vehicle. *Journal of Intelligent and Fuzzy Systems*, 17:395–405, 2006.

[59] P. Missiuro and N. Roy. Adapting Probabilistic Roadmaps to Handle Uncertain Maps. In *IEEE Conference on Robotics and Automation (ICRA)*, pages 1261–1267, Orlando, FL, May 2006.

[60] D. Frigioni, A. Marchetti-Spaccamela, and U. Nanni. Fully Dynamic Algorithms for Maintaining Shortest Paths Trees. *Journal of Algorithms*, 34(2):251 – 281, 2000.

[61] G. Ramalingam and T. Reps. An Incremental Algorithm for a Generalization of the Shortest-path Problem. *Journal of Algorithms*, 21(2):267 – 305, Sept. 1996.

[62] A. Stentz. Optimal and Efficient Path Planning for Partially-known Environments. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3310 –3317, May 1994.

[63] A. Stentz. The Focussed D* Algorithm for Real-Time Replanning. In *International Joint Conference on Artificial Intelligence*, August 1995.

[64] S. Koenig and M. Likhachev. D* Lite. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI)*, pages 476–483, 2002.

[65] S. Zilberstein and S. Russell. Approximate Reasoning Using Anytime Algorithms. In Swaminathan Natarajan, editor, *Imprecise and Approximate Computation*, volume 318 of *The International Series in Engineering and Computer Science*, pages 43–62. Springer US, 1995.

[66] Maxim Likhachev, Dave Ferguson, Geoff Gordon, Anthony Stentz, and Sebastian Thrun. Anytime Search in Dynamic Graphs. *Artificial Intelligence*, 172:1613–1643, September 2008.

[67] S. Thrun. Robotic Mapping: A Survey. *Exploring Artificial Intelligence in the New Millennium*, pages 1–35, 2002.

[68] M. Csorba. *Simultaneous Localization and Map Building*. PhD thesis, University of Oxford, 1997.

[69] J.A. Castellanos and J.D. Tardos. *Mobile Robot Localization and Map building: A Multisensor Fusion Approach*. Kluwer Academic Pub, 1999.

[70] G.J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley New York, 1997.

[71] A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice*. Springer Verlag, 2001.

[72] H. Moravec and A. Elfes. High Resolution Maps from Wide Angle Sonar. In *IEEE Conference on Robotics and Automation (ICRA)*, volume 2, 1985.

[73] A. Elfes. Using Occupancy Grids for Mobile Robot Perception and Navigation. *Computer*, 22(6):46–57, 1989.

[74] K. Konolige. Improved Occupancy Grids for Map Building. *Autonomous Robots*, 4(4):351–367, 1997.

[75] K. Pathak, A. Birk, J. Poppinga, and S. Schwertfeger. 3D Forward Sensor Modeling and Application to Occupancy Grid Based Sensor Fusion. In *IEEE Conference on Intelligent Robots and Systems (IROS)*, San Diego, CA, 2007.

[76] S. Hrabar. 3D Path Planning and Stereo-based Obstacle Avoidance for Rotorcraft UAVs. In *IEEE Conference on Intelligent Robots and Systems (IROS)*, pages 807–814, Nice, France, 2008.

[77] S. Scherer, S. Singh, L. Chamberlain, and S. Saripalli. Flying Fast and Low Among Obstacles. In *IEEE Conference on Robotics and Automation (ICRA)*, pages 2023–2029, Roma, Italy, 2007.

[78] Franz Andert and F. Adolf. Online World Modeling and Path Planning for an Unmanned Helicopter. *Autonomous Robots*, 27(3):147–164, 2009.

[79] J.D. Davis and S. Chakravorty. Motion Planning Under Uncertainty: Application to an Unmanned Helicopter. *Journal of Guidance Control and Dynamics*, 30(5):1268–1276, 2007.

[80] P. R. Kumar and P. Varaiya. *Stochastic Systems: Estimation, Identification and Adaptive Control*. Prentice-Hall, New Jersey, USA, 1986.

[81] D.H. Shim, H. Chung, and S.S. Sastry. Conflict-free Navigation in Unknown Urban Environments. *IEEE Robotics and Automation Magazine*, 13(3):27–33, 2006.

[82] C. Goerzen and M. Whalley. Minimal Risk Motion Planning: a New Planner for Autonomous UAVs in Uncertain Environment. In *AHS International Specialists' Meeting on Unmmaned Rotorcraft*, Tempe, Arizona, 2011.

[83] J. Redding, J. N. Amin, J. D Boskovic, Y. Kang, K. Hedrick, J. Howlett, and S. Poll. A Real-time Obstacle Detection and Reactive Path Planning System for Autonomous Small-Scale Helicopters. In *AIAA Navigation, Guidance and Control Conference*, Hilton Head, South Carolina, 2007.

[84] C. Goerzen, Z. Kong, and B. Mettler. A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance. *Journal of Intelligent and Robotic Systems*, 57(1):65–100, 2010.

[85] U.S. Army Aviation and Missile Command. *Handling Qualities Requirements for Military Rotorcraft, ADS-33E-PRF*. Aviation Engineering Directorate, March 2000.

[86] M. J. van Nieuwstadt. Trajectory Generation for Nonlinear Control Systems, PhD Thesis, California Institude of Technology 1996.

[87] M.B. Milam. *Real-time Optimal Trajectory Generation for Constrained Dynamical Systems*. PhD thesis, California Institute of Technology, 2003.

[88] A. Jadbabaie, J. Yu, and J. Hauser. Unconstrained Receding-Horizon Control of Nonlinear Systems. *IEEE Transactions on Automatic Control*, pages 776–783, May 2001.

[89] A. Jadbabaie and J. Hauser. On the Stability of Receding Horizon Control with a General Terminal Cost. *IEEE Transactions on Automatic Control*, 50(5):674–678, 2005.

[90] B. Mettler, N. Dadkhah, and Z. Kong. Agile Autonomous Guidance Using Spatial Value Functions. *Control Engineering Practice*, 18(7):773 – 788, 2010. Special Issue on Aerial Robotics.

[91] Real-time Interactive Prototype Technology Integration/Development Environment (RIPTIDE). `http://uarc.ucsc.edu/flight-control/riptide/`.

[92] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng. ROS: an Open-source Robot Operating System. In *Proc. Open-Source Software Workshop Int. Conf. Robotics and Automation (ICRA)*, Kobe, Japan, 2009.

[93] B. Mettler, N. Dadkhah, Z. Kong, and J. Andersh. Infrastructure and Research Framework for Human and Autonomous Interactive Guidance. Philadelphia, PA, June 2012. International Conference on Unmanned Aircraft Systems (ICUAS).

[94] N. Dadkhah and B. Mettler. System Identification Modelling and Flight Characteristics Analysis of Miniature Co-Axial Helicopter. *Accepted for Publication in Journal of American Helicopter Society*, 2012.

[95] N. Dadkhah and B. Mettler. Design and Analysis of Path Following for Robust Autonomous Guidance. Philadelphia, PA, June 2012. International Conference on Unmanned Aircraft Systems (ICUAS).

[96] N. Dadkhah and B. Mettler. Sensory Predictive Guidance in Partially Known Environment. In *AIAA Guidance, Navigation, and Control Conference*, Portland, Oregon, August 2011.

[97] Vicon MX Systems, 2010. `http://www.vicon.com`.

[98] R. Simmons and D. James. Inter-process communication: A reference manual. *Robotics Institute, Carnegie Mellon University*, 2001. `http://www.cs.cmu.edu/~IPC`.

[99] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with Application to Tracking and Navigation*. John Wiley and Sons, 2001.

[100] J.L. Crassidis and J.L. Junkins. *Optimal Estimation of Dynamic Systems*. Chapman & Hall, 2004.

[101] F. S. Grassia. Practical Parameterization of Rotations Using the Exponential Map. *Journal of Graphics Tools*, 3(3):29–48, 1998.

[102] S. Shajiee, N. Kundak, and B. Mettler. Identification Modeling and Analysis of the Coupled Drive-Train/Inflow in Miniature Rotorcraft Axial Flight Dynamics. Montreal, Canada, May 2008. Proceedings of the American Helicopter Society 64th Annual Forum.

[103] S. Sunada, H. Sumino, A. Matsue, and H. Tokutake. Analysis of Flapping Motion of Rotors of Small Coaxial Helicopter. *Transaction of Japan Society for Aeronautical and Space Science*, 49(164):101–108, 2006.

[104] B. Mettler. *Identification Modeling and Characteristics of Miniature Rotorcraft*. Kluwer Academic Publishers, Boston, 2002.

[105] R. W. Prouty. *Helicopter Aerodynamics*. PJS Publication Inc., 1985.

[106] M. Tischler and M. Cauffman. Comprehensive Identification from Frequency Responses (CIFER): An Interactive Facility for System Identification and Verification. Technical report, NASA CP 10149, USAATCOM TR-94-A-017, September 1994.

[107] M.A. Lawler, C.M. Ivler, M. B. Tischler, and Y.B. Shtessel. System Identification of the Longitudinal/Heave Dynamics for a Tandem-Rotor Helicopter Including Higher-Order Dynamics. Keystone, Colorado, 2006. AIAA Atmospheric Flight Mechanics Conference and Exhibit.

[108] J.S. Bendat and A.G. Piersol. *Engineering Applications of Correlation and Spectral Analysis*. John Wiley & Sons, New York, NY, 1993.

[109] R. R. Burrows and G. A. McDaniel. A Method of Trajectory Analysis with Multi-mission Capability and Guidance Application. AIAA Guidance, Control and Flight Dynamics Conference, 1968.

[110] R. E. Maine and K. W. Iliff. Theory and Practice of Estimating the Accuracy of Dynamic Flight-determined Coefficient. Technical Report PR 1077, NASA, 1986.

[111] R. D. Milne. The Analysis of Weakly Coupled Dynamical Systems. *Int. Journal of Control*, 2(2), 1965.

[112] S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control Analysis and Design.* John Wiley and Sons, 2005.

[113] G. J. Balas, R. Chiang, A. Packard, and M. Safonov. Robust control toolbox. *For Use with Matlab. Users Guide, Version*, 3, 2005.

[114] Anon. Application of Multivariable Control Theory To Aircraft Flight Control Laws, Final Report: Multivariable Control Design Guidelines. Technical Report WL-TR-96 3099, Honeywell and Lockheed Martin Final Report to U.S. Air Force, May 1996.

[115] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C++: The Art of Scientific Computing.* Cambridge University Press, 2002.

[116] M. Spenko, Y. Kuroda, S. Dubowsky, and K. Iagnemma. Hazard Avoidance for High-Speed Mobile Robots in Rough Terrain. *Journal of Field Robotics*, 23(5):311–331, 2006.

[117] A. Pavlov, N. Wouw, and H. Nijmeijer. Frequency Response Functions for Nonlinear Convergent Systems. *IEEE Transaction on Automatic Control*, 52(6), June 2007.

[118] A. Pavlov, N. van de Wouw, A. Pogromsky, M.F. Heertjes, and H. Nijmeijer. Frequency Domain Performance Analysis of Nonlinearly Controlled Motion Systems. In *IEEE Conference on Decision and Control*, New Orleans, LA, 2007.

[119] T. Schouwenaars, B. Mettler, E. Feron, and J. How. Hybrid Model for Trajectory Planning of Agile Autonomous Vehicles. *Journal of Aerospace Computing, Information, and Communication*, 1, December 2004.

[120] Eduardo D. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems.* Springer, New York, NY, 1998.

[121] A. E. Bryson and Y. C Ho. *Applied Optimal Control.* Taylor and Francis, Bristol, PA, 1975.

[122] R. Bellman and S.E. Dreyfus. *Applied Dynamic Programming.* Princeton University Press, Princeton, New Jersey, 1962.

[123] E. Frazzoli, M. A. Dahleh, and E. Feron. A Hybrid Control Architecture for Aggressive Maneuvering of Autonomous Helicopter. Phoenix, AZ, December 1999. IEEE Conference on Decision and Control.

[124] E. Frazzoli, M.A. Dahleh, and E. Feron. Real-Time Motion Planning for Agile Autonomous Vehicles. *AIAA Journal of Guidance, Control, and Dynamics*, 25(1):116–129, 2002.

[125] Donald E. Kirk. *Optimal Control Theory*. Dover Publications, Inc., Mineola, NY, 1998.

[126] B. Mettler and Z. Kong. Receding Horizon Trajectory Optimization with a Finite-state Value Function Approximation. Seattle, Washington, June 2008. American Control Conference.

[127] Z. Kong, V. Korukanti, and B. Mettler. Mapping 3D Guidance Performance Using Approximate Optimal Cost-to-go Function. Chicago, Illinois, August 2009. AIAA Guidance, Navigation, and Control Conference.

[128] S. Chakravorty and J.L. Junkins. Motion Planning in Uncertain Environments with Vision-like Sensors. *Automatica*, 43(12):2104–2111, 2007.

[129] Hokuyo URG-04LX-UG01 Scanning Laser Rangefinder. `http://www.autonomoustuff.com/hokuyo-laser-scanners.html`.

[130] OpenSeneGraph: 3D Graphics Application Programming Interface. `www.openscenegraph.org`.

[131] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, 2005.

[132] Mario Micheli. A Probabilistic Approach to Three-dimensional Autonomous Navigation, PhD Thesis, University of Padova, Italy 1999.

[133] Sean Q. Marlow and Jack W. Langelaan. Local Terrain Mapping for Obstacle Avoidance Using Monocular Vision. *Journal of the American Helicopter Society*, 56(2), 2011.

[134] K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: A Probabilistic, Flexible, and Compact 3D Map Representation for Robotic Systems. In *Proc. of the ICRA 2010 Workshop on Best Practice in 3D*

Perception and Modeling for Mobile Manipulation, Anchorage, AK, USA, May 2010.

[135] J. Fischer and Á. del Río. A Fast Method for Applying Rigid Transformations to Volume Data. In *Proc. of WSCG*, Plzen, Czech Republic, 2004.

[136] J. Bellingham, A. Richards, and J. How. Receding Horizon Control of Autonomous Aerial Vehicles. pages 3741–3746. American Control Conference, May 2002.

[137] B. Mettler and O. Toupet. Receding Horizon Trajectory Planning with an Environment-Based Cost-to-go Function. Seville, Spain, Dec. 2005. Proceedings of the joint ECC-CDC Conference.

[138] R. Fourer, D.M. Gay, and B.W. Kernighan. *AMPL: A Mathematical Programming Language*. Brooks/Cole Publishing Company, 2002.

[139] IBM ILOG CPLEX. High-performance Software for Mathematical Programming and Optimization. 2011. `http://www.ilog.com/products/cplex`.

[140] A. E. Bryson. *Dynamic Optimization*. Addison Wesley Longman, 1999.

[141] B. Mettler and E. Bachelder. Combining On- and Offline Optimization Techniques for Efficient Autonomous Vehicle's Trajectory Planning. Number AIAA 2005-5861, San Francisco, CA, August 2005. AIAA Guidance, Navigation and Control Conference and Exhibit.

[142] J. Borenstein and Y. Koren. The Vector Field Histogram- Fast Obstacle Avoidance for Mobile Robots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288, 1991.

[143] H. Choset, K.L. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun. *Principles of Robot Motion; Theory, Algorithms, and Implementations*. MIT Press, Cambridge, MA, 2005.

[144] D. Soetanto, L. Lapierre, and A. Pascoal. Adaptive, Non-singular Path-following Control of Dynamic Wheeled Robots. In *IEEE Conference Decision and Control*, pages 1765–1770, 2003.

[145] H. K. Khalil. *Nonlinear Systems*. Prentice Hall, 2003.

[146] S. H. Wang and E. J. Davison. On the Stabilization of Decentralized Control Systems. *IEEE Transaction Automatic Control*, 18(5):473–478, 1973.

Appendix A

A.1 Nonlinear Analysis of Path Following Based on Point Mass Model

This section of the appendix addresses the problem of steering a point-mass model(as an approximation for the rotorcraft closed-loop dynamics) along a desired trajectory. The point mass model has the following dynamics

$$\begin{aligned} \dot{u} &= a_x \\ \dot{v} &= a_y \\ \dot{x} &= u \\ \dot{y} &= v \end{aligned} \tag{A.1}$$

The inputs to the system are the accelerations a_x and a_y. Consider the Serret-Frenet frame $\{F\}$ in Figure A.1 as described in [144] that moves along the path with one axis tangent to the path and another axis perpendicular to the path. The inertial frame is denoted by $\{I\}$. Let also p and q be the origin of the $\{F\}$ frame and vehicle, respectively. In addition, ψ is the path angle, s is the path arc length, κ is the path curvature and $\omega_c = \kappa\dot{s}$ is the angular velocity of the $\{F\}$ frame.

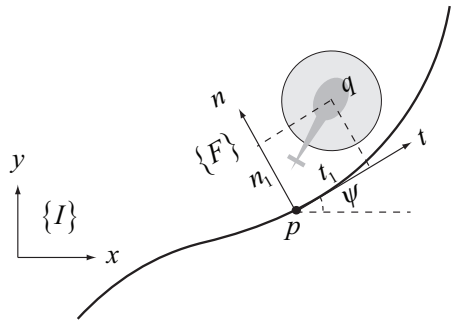


Figure A.1: Illustration of the helicopter in path following problem using a Serret-Frenet coordinate frame (t, n).

The rotation matrix from the $\{I\}$ frame to the $\{F\}$ frame is given by

$$C_I^F(\psi) = \begin{bmatrix} \cos\psi & \sin\psi \\ -\sin\psi & \cos\psi \end{bmatrix} \tag{A.2}$$

The inertial velocity of point q in $\{F\}$ can be expressed as

$${}^F\left(\frac{dr}{dt}\right) = C_I^F(\psi)\,{}^I\left(\frac{dq}{dt}\right) - {}^F\left(\frac{dp}{dt}\right) - {}^F\left(\omega_c \times r\right) \tag{A.3}$$

where

$$\begin{aligned} {}^F r &= [t_1,\, n_1]^T \\ {}^F\left(\frac{dp}{dt}\right) &= [\dot{s},\, 0]^T \\ {}^I\left(\frac{dq}{dt}\right) &= [\dot{x},\, \dot{y}]^T \end{aligned} \tag{A.4}$$

Combining and rearranging equations (A.2) to (A.4) yields

$$\begin{aligned} \dot{t}_1 &= \cos\psi\,\dot{x} + \sin\psi\,\dot{y} - \dot{s} + \kappa\,\dot{s}\,n_1 \\ \dot{n}_1 &= -\sin\psi\,\dot{x} + \cos\psi\,\dot{y} - \kappa\,\dot{s}\,t_1 \end{aligned} \tag{A.5}$$

Combining equation (A.1) with (A.5), the complete dynamical model for path following

is

$$\begin{aligned} \dot{t}_1 &= \cos\psi\, u + \sin\psi\, v - \dot{s} + \kappa\, \dot{s}\, n_1 \\ \dot{n}_1 &= -\sin\psi\, u + \cos\psi\, v - \kappa\, \dot{s}\, t_1 \\ \dot{u} &= a_x \\ \dot{v} &= a_y \end{aligned} \tag{A.6}$$

Equations (A.6) can be written in the form of

$$\dot{z} = f(z) + g(z)\zeta \tag{A.7}$$

$$\dot{\zeta} = a \tag{A.8}$$

where $z = [t_1,\, n_1]^T$, $\zeta = [u,\, v]^T$ and $a = [a_x\, a_y]^T$. Backstepping technique [145] can be used to find a control law that allows to regulate the tracking error t_1 and n_1. To this end, we first seek a control law of the form $\zeta = \Phi(z)$ to stabilize the system (A.7). Given a Control Lyapunov Function (CLF) of the form

$$V = \frac{1}{2}(t_1^2 + n_1^2) \tag{A.9}$$

after some manipulations, it can be shown that the following control law

$$\zeta = \Phi(z) = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \dot{s}\, \cos\psi + n_1\, \sin\psi - t_1\, \cos\psi \\ \dot{s}\, \sin\psi - n_1 \cos\psi - t_1\, \sin\psi \end{bmatrix}$$

implies that

$$\dot{V} = -n_1^2 - t_1^2 \leq 0 \tag{A.10}$$

Now in order to take into account the integral action in equation (A.8), let us now state the backstepping lemma [145]:

Lemma 1 ***(Backstepping)*** *Consider the system (A.7)-(A.8) and let $\Phi(z)$ be a stabilizing state feedback law for the system in equation (A.7). Let V(z) be a CLF such that*

$$\frac{\partial V}{\partial z}[f(z) + g(z)\Phi(z)] \leq -W(z) \tag{A.11}$$

for some positive definite $W(z)$. Then the state feedback law of the form

$$a = \frac{\partial \Phi}{\partial z}[f(z) + g(z)\Phi(z)] - \frac{\partial V}{\partial z}g(z) - k[\zeta - \Phi(z)] \tag{A.12}$$

stabilizes the origin of (A.7)-(A.8) with $V(z) + 0.5(\zeta - \Phi(z))^T(\zeta - \Phi(z))$ as a Lyapunov function.

Applying Lemma 1 to the path following problem, it can be show that the following state feedback control law regulates the path tracking error

$$\begin{bmatrix} a_x \\ a_y \end{bmatrix} = -C_F^I(\psi) \begin{bmatrix} t_1 \\ n_1 \end{bmatrix} - \begin{bmatrix} u \\ v \end{bmatrix} \tag{A.13}$$

The first term in the control law (A.13) transforms the tangential and normal tracking error to the vehicle reference frame (here it is inertial frame) and the second term adds the velocity feedback. In essence, the control law (A.13) is a decoupled PD compensator that acts on track and cross-track errors.

A.2 Linear Analysis of Path Following Around a Nominal Circular Path

This section of the appendix provides the analysis of the path following controller for the point-mass model. The purpose of this analysis is to understand how a linear controller behaves compared to its nonlinear counterpart (which was discussed in Appendix A.1) for the path following problem of the same simplified vehicle model. It is convenient to represent the motion on a circular path using radial-tangential coordinate system as shown in Figure A.2. The equations of motion in this frame are given by

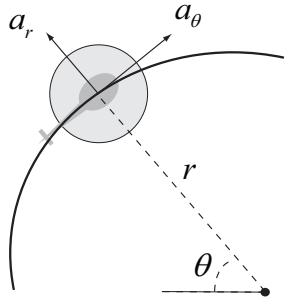


Figure A.2: Description of the helicopter motion on a circular path using radial coordinate system (r, θ).

$$\begin{aligned} \ddot{r} - r\,\dot{\theta}^2 &= a_r \\ r\,\ddot{\theta} + 2\,\dot{r}\,\dot{\theta} &= a_\theta \end{aligned} \tag{A.14}$$

Linearizing around a nominal circular path $r = r_o$ and $\dot{\theta} = \omega = \omega_o$, we obtain the following linear perturbation state-space model

$$\frac{d}{dt}\begin{bmatrix} \delta v \\ \delta r \\ \delta\omega \\ \delta\theta \end{bmatrix} = \begin{bmatrix} 0 & \omega_o^2 & 2\omega_o r_o & 0 \\ 1 & 0 & 0 & 0 \\ \frac{-\omega_o}{2r_o} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \delta v \\ \delta r \\ \delta\omega \\ \delta\theta \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & \frac{1}{r_o} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta a_r \\ \delta a_\theta \end{bmatrix} \tag{A.15}$$

where δv, δr, $\delta\omega$ and $\delta\theta$ are perturbations in velocity, position, angular rate and angle with respect to the nominal path.

The analysis based on [146] shows that the linear state-space model in equation(A.15) can be stabilized by a decentralized, decoupled state feedback controller of the form

$$\begin{bmatrix} a_r \\ a_\theta \end{bmatrix} = \begin{bmatrix} \times & \times & 0 & 0 \\ 0 & 0 & \times & \times \end{bmatrix} \begin{bmatrix} \delta v \\ \delta r \\ \delta\omega \\ \delta\theta \end{bmatrix} \tag{A.16}$$

Notice that this linear controller describes a decoupled PD controller in radial and tangential axis. The results of this linear analysis are in accordance with the nonlinear analysis in Appendix A.1.