Robust Post-Stall Perching with a Fixed-Wing UAV

by

Joseph L. Moore

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2014

Author . . . . . .
Joseph L. Moore
Ph.D. Candidate, Department of Mechanical Engineering
August 8, 2014

Certified by . . . . . .
Russ Tedrake
X Consortium Associate Professor of EECS - Thesis Supervisor

Certified by . . . . . .
H. Harry Asada
Ford Professor of Engineering - Thesis Committee Chair

Accepted by . . . . . .
David E. Hardt
Graduate Officer

Robust Post-Stall Perching with a Fixed-Wing UAV

by

Joseph L. Moore

Submitted to the Department of Mechanical Engineering
on August 8, 2014, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Mechanical Engineering

Abstract

Consider a bird perching on a branch. In the presence of environmental disturbances and complicated fluid flow, the animal exploits post-stall pressure drag to rapidly decelerate and land on a perch with a precision far beyond the capabilities of our best aircraft control systems.

In this thesis, I present a controller synthesis technique for achieving robust, post-stall perching with a fixed-wing Unmanned Aerial Vehicle (UAV). Using sum-of-squares (SOS) programming and building on a novel control synthesis approach known as LQR-Trees, I demonstrate the ability to improve the robustness of the post-stall perching manuever to variable initial conditions, modeling error, and external disturbances. I achieve this by developing methods for carrying out rigorous robust verification of the nonlinear aircraft model along a time-varying nominal perching trajectory in the presence of both dynamic and parametric uncertainty. I also present methods for carrying out stochastic verification in the presence of Gaussian acceleration uncertainty and adaptive control techniques for improving controller performance in the presence of parametric uncertainty.

Using the robust verification techniques for dynamic uncertainty, I proceed to generate a robust LQR-Tree controller and test that controller on real hardware using a small 24 inch wing span glider and a Vicon motion capture studio. The experiments show successful perching for 94 percent of the 147 flights launched between 6 and 8 m/s. I further demonstrate robustness to initial pitch variations by launching the glider by hand and showing repeatable successful perching results.

Following these experiments, I then build a magnetic field sensing system capable of estimating the position of the perching UAV using the magnetic field generated by a powerline. I demonstrate on hardware that the aircraft is still capable of successfully executing a closed-loop perching maneuver indoors using the lower fidelity state estimates. I then move the entire experiment outdoors and begin testing the UAV's perching performance in the presence of wind gusts. Very quickly, it becomes apparent that the glider can not reliably perch in wind. To address this, I describe a control approach for mitigating the effects of wind on aircraft performance and propose an experiment for testing this approach.

Thesis Supervisor: Russ Tedrake
Title: X Consortium Associate Professor of EECS

Acknowledgements

Over the past few years, there have been many people whose support has been crucial to the success of this thesis. I would first like to thank my thesis advisor, Russ Tedrake, for his insight, encouragement, and the many hours, both day and night, he dedicated to helping me hone my skills as a researcher. His vision for the future of robotics, his love for science, and his unwavering pursuit of excellence in all things have been a continual source of inspiration for me and the lab.

Secondly, I would like to thank the other members of the Robot Locomotion Group, who through many invaluable discussions have helped push this work through countless roadblocks. I would also like to thank the members of the Distributed Robotics Group and the Robust Robotics Group, who have provided a great deal of technical support and advice during my time in graduate school. In addition, I would like to thank my thesis committee: Professor Harry Asada, Professor Franz Hover, and Dr. Anuradha Annaswamy. They guided me steadily through the thesis process, provided me with an abundance of good advice, and helped me produce a much stronger and cohesive thesis as a result.

Perhaps, most importantly, I would like to thank my wife, Kayleen, who has been with me every step of the way, spurring me on by both word and deed, even when it meant helping to solder circuit components. She has more than earned an honorary engineering degree during my time in graduate school. I would also like to thank my parents, my sisters, and my grandparents. If it were not for their constant support, love, sacrifice, and the foundation they laid, I would not be where I am today. Finally, I would like to thank my Lord and Savior, Jesus Christ, the Author of all things, who has sustained me throughout graduate school and has been faithful even when I have been faithless.

Contents

List of Figures

Chapter 1

Introduction

(a)

(b)

Figure 1-1: (a) An Eagle lands on a branch in wind, compensating for the effects of the external disturbance (Estrada, 2010). (b) A Eurasian Eagle Owl spreads its wings and pitches up to a very high body angle (Turbary, 2010).

Birds routinely execute complicated flight maneuvers in the presence of aerodynamic uncertainty and do so with the utmost accuracy and agility. Consider a bird landing on a perch: As the bird approaches its target, it pitches up rapidly, spreads out its wings, and enters a post-stall flow regime, using pressure drag to slow itself down and land on the perch [6, 5, 35, 13]. What is even more astounding is that the bird can accomplish this maneuver in the face of complex, uncertain post-stall aerodynamics and external disturbances such as wind gusts. In contrast, few man-made aircraft even attempt to fly at such high angles of attack [83], and those that do usually make use of thrust vectoring and a large thrust-to-weight ratio [2]. These advanced aircraft, which can easily execute a post-stall maneuver in open air, greatly reduce their angle of attack whenever they carry out a landing maneuver.

Figure 1-2: An F-18 lands on an aircraft carrier (U.S. Navy, 2008). Notice that the fighter jet lands at a relatively low angle of attack (about 8.1 degrees) in order to avoid stall. The aircraft must be slowed down by a hook and tow cable as it lands.

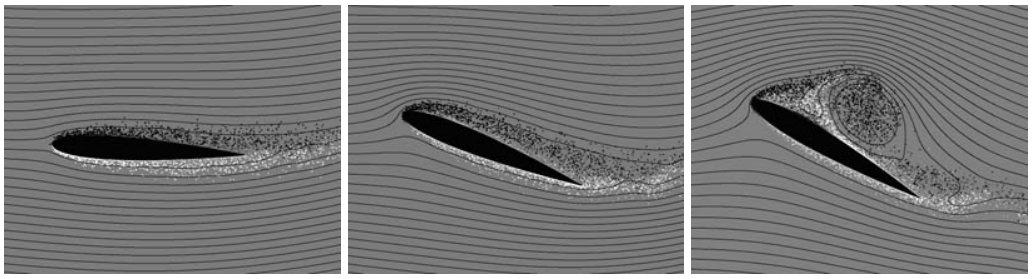


Figure 1-3: Image shows the flow around an airfoil as the airfoil transitions into a stall regime (van Dommelen, 2009). Notice how the flow in the third image is separated from the wing and exhibiting transient vortex shedding.

For instance, when landing on an aircraft carrier, the automatic control system used on an F-18 fighter jet maintains a 8.1 degree angle of attack [83]. Birds, in comparison, can reach up to 90 degrees angle of attack when landing on the branch of a tree [14].

One primary reason why many man-made aircraft avoid post-stall flight is because these conditions often correspond to a loss of control authority. When an airfoil transitions in to post-stall flight, the airflow begins to separate from the wing's leading edge, causing a sudden drop in lift. This is often problematic because, with such a drastic reduction in lift, many aircraft are unable to maintain their desired altitude. Moreover, this post-stall flow is both unsteady and turbulent, severely complicating the effort to model and control the aircraft. Birds, however, seem undaunted by the challenges posed by post-stall flight. By pitching their bodies up rapidly, they are able to exploit the increased pressure drag in this

Figure 1-4: A Navy X-47B Unmanned Combat Air System (U.S. Air Force, 2011). Currently, UAVs also must land at low angles of attack to avoid stall. Such unmanned systems could benefit greatly from the development of robust fixed-wing perching maneuvers.

complicated flow regime to land quickly and accurately [14, 13].

In this thesis, I first describe an approach for achieving robust post-stall perching with a fixed-wing UAV from a wide range of initial conditions in the presence of model uncertainty. This approach involves applying a feedback motion planning algorithm known as a *robust* LQR-Tree to account for uncertainty in the aircraft aerodynamics. Following in the footsteps of the method outlined in [99], this algorithm generates a library of local time-varying linear quadratic regulators (TVLQR) which can be combined together in to a tree-like structure to cover a larger region of state-space than what would be otherwise achievable by a single controller alone. However, unlike the algorithm in [99], the algorithm in this thesis is able to include robustness to parametric and dynamic uncertainty in the verification step, allowing the generation of a *robust* LQR-Tree. To validate my approach, I demonstrate this method on hardware, first by showing improved perching performance at higher initial launch speeds, and then by showing improved performance from a wide range of initial pitch angles.

In the second part of this thesis, I address the problem of fixed-wing perching in the presence of external disturbances. To do solve this problem, I develop a magnetic field sensing system capable of localizing the aircraft in outdoor environments. This is accomplished by combining the measurements from a magnetoresistive magnetometer and an

inertial measurement unit and by using an extended Kalman filter (EKF) to estimate the state of the aircraft. These magnetic field based estimates are then used in place of the state estimates from Vicon motion capture to execute the TVLQR feedback law. After showing that it is possible to successfully achieve closed-loop perching using magnetic field measurements indoors, I move the magnetic sensing system outdoors to test the performance of the system in wind gusts.

To mitigate the effect of windgusts on the perching maneuver, I incorporate the planar wind velocities into the glider flight dynamics and model the wind in the x and z dimension as white-noise passed through a first-order filter. I then proceed to augment the glider's seven states with two additional wind states to create a nine dimensional system, which includes the wind dynamics. By modeling the wind in this manner, I reduce the system uncertainty to white Gaussian noise on the system states. Moreover, since this is exactly the type input disturbance LQR attempts to minimize, I continue to apply TVLQR as the local feedback law.

To verify this system controller, I explore techniques for stochastic verification, where I seek to verify the probability that the system will reach the goal state, and methods for robust verification, where instead of unbounded white input noise, I consider bounded input noise. Once I am able to verify these local controllers, I then construct an LQR-Tree over a reasonable set of nominal wind speeds to achieve robust perching in the presence of external disturbances. I successfully demonstrate this approach in simulation and use a three-dimensional ultrasonic anemometer along with the powerline magnetic sensing system to test this approach on real hardware in an outdoor environment.

Chapter 2

The Post-Stall Perching Problem

The task of accurately landing an Unmanned Aerial Vehicle (UAV) with sufficiently low speed to perch on a branch or ledge has long been a problem explored by controls and aerospace engineers alike. In this chapter, I discuss my formulation of the perching problem, which is identical to the formulation presented in [22], where post-stall pressure drag is used to slow the vehicle down and land. However, I first describe the perching aircraft that already exist, in order provide a suitable context for understanding the chosen approach.

2.1 Perching UAVs

Perching UAVs have the potential to significantly expand the mission capabilities of autonomous aerial vehicles. By being able to execute a controlled landing maneuver in a short distance, these aircraft have the ability to land on a branch, ledge or powerline to conduct perch and stare surveillance and dramatically increase mission duration by eliminating the need for continuous flight. For this reason, the design and control of perching aircraft has received considerable attention from the controls and aerospace communities. In recent years, there have been many solutions proposed to solve the perching problem. Overall, these approaches can be divided into two different categories: those that rely primarily on novel vehicle design to achieve perching and those that use standard vehicle designs and rely primarily on control system design.

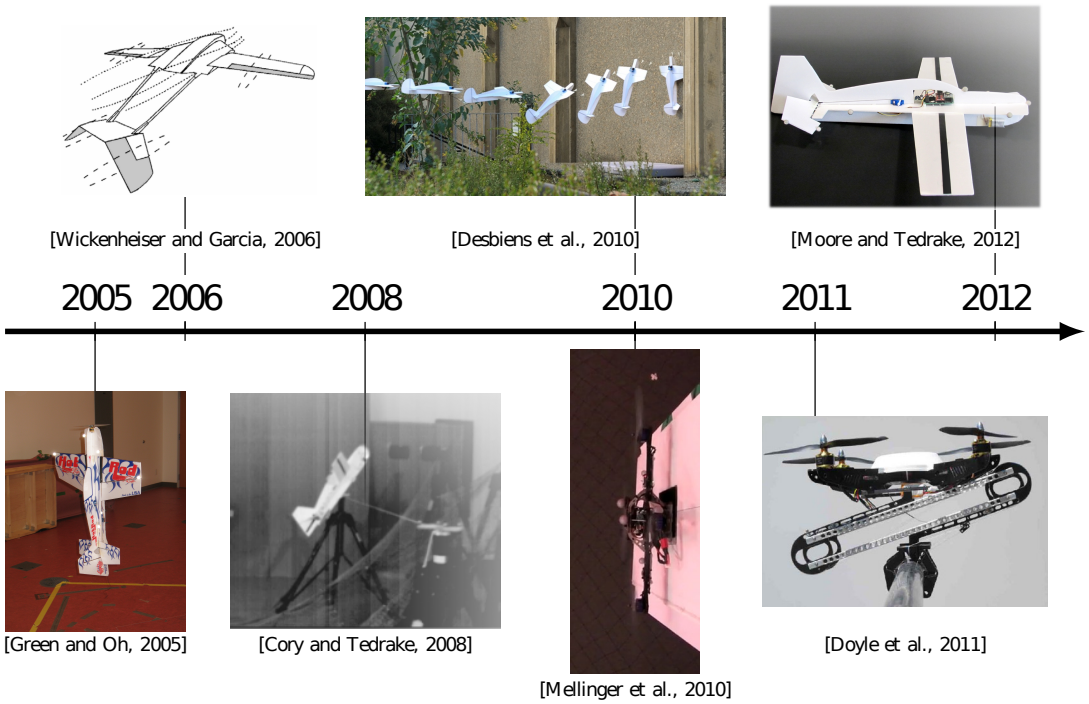


Figure 2-1: A timeline showing various micro UAV perching strategies. In many cases, these strategies can be divided in to those that focus primarily on vehicle design and those that focus primarily on control system design.

2.1.1 Vehicle Design Solutions

The morphing-wing approach to the perching problem investigated in [109, 110] is one of the earliest examples of using novel vehicle design to achieve UAV perching. While this method does employ an optimal control strategy for trajectory generation, it relies heavily on its morphing airframe to rotate its fuselage upwards while keeping its wings at a low angle of attack. In this way, the vehicle is able to exploit the post-stall pressure drag of the fuselage to reduce speed while still making use of attached flow on the wings to apply more conventional control strategies. A similar morphing wing approach was explored in [58, 84], where the authors investigated the perching capabilities of a morphing wing with multiple sections. Here, in a manner similar to [109, 110], the authors explored stalling the inner sections of the wing while keeping the outermost sections at a low angle of attack.

A different vehicle design approach was explored in [25, 26], where a fixed-wing aircraft was equipped with arrays of microspines capable of attaching to a wall. The authors launched their aircraft by hand at a wall and used an ultrasonic range finder to measure the

distance to the vertical surface. At a predetermined distance from the wall, they executed an open loop trajectory to pitch the vehicle upwards. When the vehicle came in to contact with the wall, the authors demonstrated that their micro-spine technology was robust enough to repeatedly enable the aircraft to grip on to the vertical surface.

Like [25, 26], other researchers have also sought to solve the perching problem through novel landing gear design. In [29], the authors developed an underactuated, avian-inspired grasping mechanism for rotorcrafts. This work examined the bone and muscle structure of bird legs to develop an underactuated gripper which could passively and robustly grasp to a cylindrical object. In [44], research was carried out to develop a light weight landing gear for attaching and detaching a fixed-wing UAV from walls using a dart-like mechanism.

2.1.2 Control Design Solutions

The second major class of perching UAVs encompasses those that use standard vehicle platforms and rely primarily on control design strategies to accomplish successful perching. Some of the earliest work in this area includes the development of perching aircraft using vertical take-off and landing (VTOL) control strategies with standard aerobatic hobby aircraft. Because these aerobatic aircraft have a very high thrust-to-weight ratio, it is possible to use a vehicle's powerful motor to hover it in place. While in this "prop-hang" configuration, so long as adequate sensing is available, a simple hand-tuned proportional-derivative (PD) controller can regulate the propeller speed and exploit the backwash of the propeller over the control surfaces to provide the vehicle with a large region of stability [21]. By adjusting the desired position set-points, this control strategy can then be used to achieve a *slow* and *energetically costly* VTOL perching maneuver [32, 36, 8, 20].

Another control-based perching approach which exploits high thrust-to-weight ratios is the recent work done on quadrotor perching in [67]. Building on their work in developing techniques for allowing quadrotors to carry out aggressive maneuvers using minimum snap trajectories [65, 64, 66], these authors have demonstrated experimentally robust approaches for landing and perching with quadrotors using a Vicon motion capture studio. And while these authors do use micro spines to enable quadrotor perching on vertical surfaces, the

control design methods themselves are the central contribution of this work.

Motivated primarily by studies carried out on bird flight kinematics during perching, such as [35], [5], and [6], which indicate that birds are exploiting high angles of attack in ways conventional aircraft never have, the authors in [22] placed a large emphasis on using control design to solve the perching problem. In [22, 38], the authors fully embraced the complicated, post-stall flow regime associated with bird-like perching maneuvers and built a nonlinear, post-stall model for a flat plat, unpowered glider. The authors then designed an optimal open-loop trajectory for that glider, applied time-varying LQR about this trajectory and successfully demonstrated their approach experimentally [20]. In [86], the authors applied a local time-varying linear controller to the glider in [22] and evaluated the controllability of the aircraft during a perching maneuver. This work effectively exposed many of the fundamental challenges which make the perching problem hard, even though the interpretation of the results was limited by the linearization-based analysis.

2.2 Post-Stall Perching

In the perching UAV approaches described above, few of the vehicles seek to make full use of pressure drag, as birds do, to slow down and land on their target. As mentioned previously, one reason for this is that the high angle of attack required for such maneuvers is associated with a decrease in lift and a loss of control authority. As shown in Figure 1-3, as the angle of attack of an airfoil increases, the air flow behind the wing begins to separate and eventually stall. Not only does this post-stall flow lead to reduced lift, but it also is extremely challenging to model and control, since the aerodynamics are both turbulent and unsteady.

Of the approaches described above, only [22] fully exploits post-stall pressure drag to land on a perch. To achieve the highest possible perching performance for a fixed-wing UAV, I decided to build upon the work in [22, 20, 86], with the intent of improving the robustness of the maneuver. Following the work described in [22], I constructed a fixed-wing glider and used the post-stall pressure drag of both the body and wings to accomplish the perching maneuver.

In the following sections, I introduce the perching problem as first outlined in [22]. Though much of this experimental set-up was described in [22], I revisit it here for the benefit of the reader, since all of the work that follows builds upon the initial efforts in [22].

2.3 Problem Formulation

Figure 2-2: A pigeon (Columbia livia) in flight (Wilson, 2006). The perching maneuver described here has similar flow characteristics to that of a pigeon landing on a perch.

I consider the problem of landing on a string with a 62 cm wingspan glider (no onboard propulsion) with a 9.8 cm cord. The size of the aircraft is comparable to that of common rock pigeon (Columbia livia), which has a wingspan ranging between 60 and 71 cm [79]. As shown in Figure 2-3 and described in [22], the glider is launched at a distance of 3.5 m from the perch with an initial speed of approximately 7 m/s (Re $\approx$ 50,000). It then must decelerate to almost zero velocity and come to rest on the string in a fraction of a second. This too matches well with the initial flow conditions observed in [6], where pigeons (a chord of 22.3 cm) are recorded approaching a perch from 2 m away with initial speeds around 4 m/s (Re $\approx$ 60,000) and landing with speeds below 1 m/s.

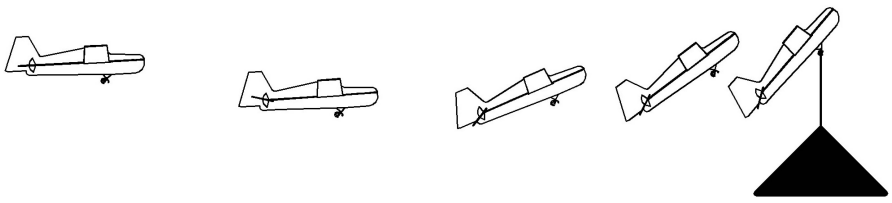


Figure 2-3: The fixed wing perching maneuver as defined in [22]. The aircraft starts 3.5m from the perch at 7m/s, pitches up and lands with low speed in a small capture region.

I claim that, with these specifications, the glider must execute a high angle of attack maneuver, exploiting both viscous and pressure drag to achieve the required rapid deceleration. Despite the obvious nonlinearity and complexity of this dynamic regime and the short-term loss of control authority on the stalled control surface, the glider's center of mass must land within 6.5 cm of the string, which is the approximate capture region for the simple hook landing gear. This post-stall perching strategy is similar to what is employed by the pigeon, which as reported in [6, 14], pitches up to body angles beyond 90 degrees when landing.

2.4 Vehicle Design

Figure 2-4: The flat plate glider is constructed from foam with carbon reinforced wings and a single actuator at the tail.

The experimental aircraft, whose design is derived from commercially available aerobatic aircraft, is an unpowered glider constructed from foam with 12-inch, carbon fiber reinforced wings. This carbon fiber reinforcement imparts substantial stiffness to the wings and eliminates all observable wing flexing during flight. To minimize complexity, I used a single control surface at the tail (i.e., the elevator) to control the aircraft's longitudinal dynamics but forwent all other control surfaces and relied on passive stability (and short flight durations) in roll and yaw. By simplifying the glider design and keeping the aircraft

in a two-dimensional plane, I tried to experimentally isolate the aspect of the maneuver which was of interest (i.e., post-stall perching). To actuate this control surface, I used a HS-50 Hitec hobby servo motor.

The aircraft's wings, tail and fuselage are simple flat plates, without any camber. The wings are slightly tapered with a 98 mm mean chord and, together, the wings, elevator and fuselage have a total surface area of 0.102 m^2. Six 11 mm reflective markers were placed along the fuselage and four on the elevator control surface to facilitate motion capture tracking. Just below the center of mass, the vehicle has a small slider which moves along a track on the cross-bow launching system. The slider stand-off doubles as a landing-gear for the vehicle, which makes contact with the perch when the vehicle's center of mass is within a 6.5 cm capture region. A GWS four channel micro receiver receives commands from a 72 MHz transmitter and the whole system is powered by a Full River 250 mAh 2-cell Lithium polymer battery for a total weight of 85 grams.

2.5 Experimental Setup

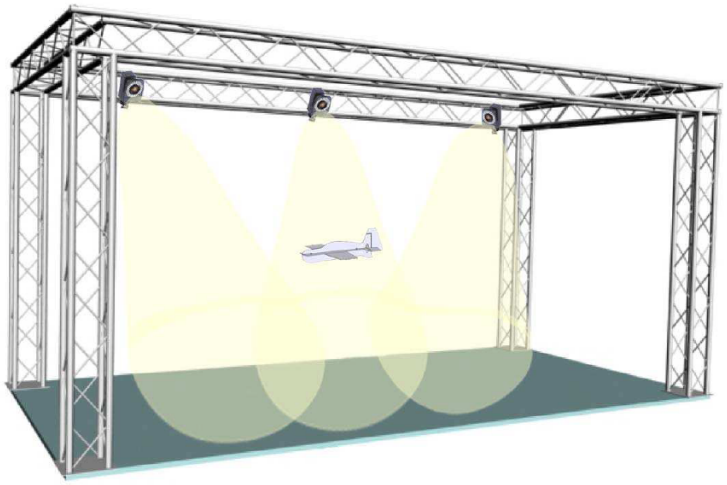


Figure 2-5: The Vicon motion capture arena shown above is equipped with 16 cameras to capture the position of the aircraft in flight. I used a crossbow with a carbon fiber bow to launch the aircraft.

To separate the control problem from the sensing problem and to improve experimental repeatability, I used an indoor Vicon motion capture arena. This arena consists of 16

infrared cameras with a total capture region of $27m^3$ and the capability of tracking the positions of reflective markers to sub-millimeter accuracy. To prevent damage to the glider during flight, I hung a net above the floor and behind the perch. For the perch, I used a small string suspended at one end of the capture region.

I applied the reflective markers to the glider's fuselage and elevator as well as the perch's position and used the Vicon motion capture system's software to reconstruct the positions and orientations of the aircraft in real-time. These raw motion capture measurements were then passed in to a state estimator to produce the full-state required by the perching control algorithm.

To launch the glider between 6 and 8 m/s, I used a crossbow launching mechanism constructed with a carbon fiber bow. As soon the the glider was launched, I used the Vicon motion capture system to detect the aircraft and send the aircraft's positions and orientations to a Linux base station at 90 Hz. After computing the state estimates, I used the controller running on the base station to evaluate the servo command required to achieve fixed-wing perching. This servo command was then sent to the RC Transmitter via the base station's serial port at 45 Hz and transmitted wirelessly to the glider's onboard receiver.

Chapter 3

System Identification and Modeling

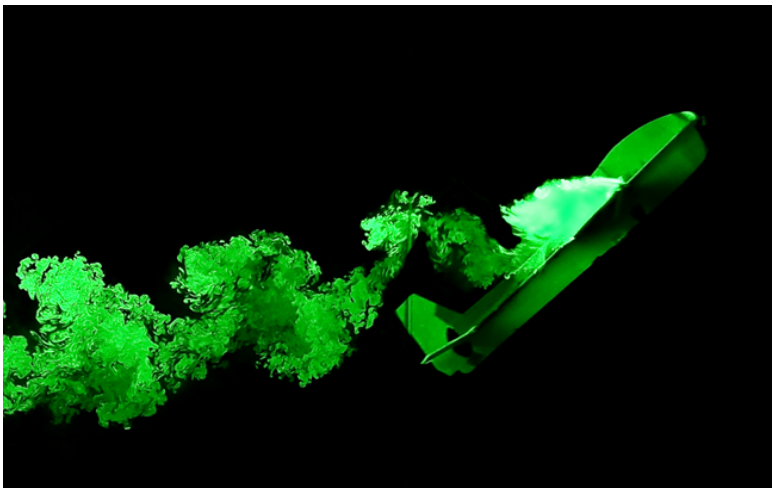


Figure 3-1: Flow visualization of the perching glider carried out by Cory in [20]. Even though the post-stall aerodynamics are turbulent and unsteady, they can be well approximated by average lift and drag coefficients.

For feedback control design and analysis, the aircraft was modeled as a rigid body whose dynamics are governed by gravitational and aerodynamic forces. Initially, these aerodynamic forces were derived by approximating all lifting surfaces as flat plates. However, to obtain high performance perching, I improved this model by using a more descriptive functional approximation of the aircraft's aerodynamic coefficients. This was achieved by using the flat plate glider model as a baseline and augmenting the lift and drag coefficients using radial basis functions. Here I describe both the flat plate glider model and its modifications.

3.1 Flat Plate Glider Model

Following [22], the flat plate glider model is based on the aerodynamic coefficients put forth in [95], which are given as

$$C_L(\alpha) = 2\sin(\alpha)\cos(\alpha) \tag{3.1}$$

$$C_D(\alpha) = 2\sin^2(\alpha), \tag{3.2}$$

where α is the angle of attack of the lifting surface. As in [22], the model restricts the aircraft dynamics to two dimensions, ignoring contributions from yaw, roll, and three dimensional aerodynamics. This leads to a seven dimensional control differential equation,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}),$$

where the states are x-position (m), z-position (m), pitch (rad), elevator angle (rad), x-velocity (m/s), z-velocity (m/s), and pitch rate (rad/s), as illustrated in Fig. 3-2. These states are represented by $\mathbf{x} = [x, z, \theta, \phi, \dot{x}, \dot{z}, \dot{\theta}]$ respectively. It is assumed that direct control over the angular rate of the elevator, $\mathbf{u} = \dot{\phi}$, is possible.

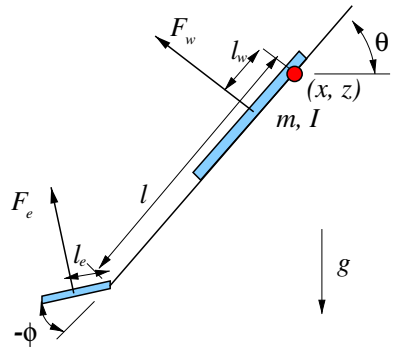
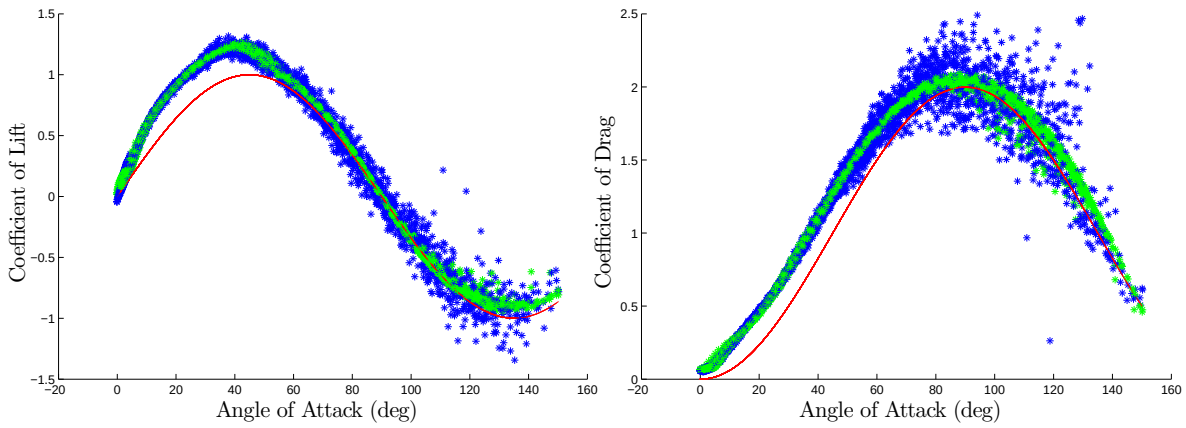


Figure 3-2: Aircraft Model. x and z denote the positions of the center of mass, θ denotes the pitch angle, and ϕ denotes the elevator angle.

As illustrated in Figure 3-2, the unit vectors are defined normal to the control surfaces



(a) (b)

Figure 3-3: **(a-b)** Plot of Lift and Drag Coefficients: Blue represents the flight data, red represents the flat plate model, and green is the flat plate model augmented with radial basis functions

in the directions of the force vectors as

$$\mathbf{n}_w = \begin{bmatrix} -s_\theta \\ c_\theta \end{bmatrix}, \quad \mathbf{n}_e = \begin{bmatrix} -s_{\theta+\phi} \\ c_{\theta+\phi} \end{bmatrix},$$

where $s_\gamma = \sin(\gamma)$ and $c_\gamma = \cos(\gamma)$. This allows solving for the kinematics of the geometric centroid of the aerodynamic surfaces, which, for the flat plate model, is equivalent to the mean aerodynamic chord. This yields

$$\mathbf{x}_w = \begin{bmatrix} x - l_w c_\theta \\ z - l_w s_\theta \end{bmatrix}, \quad \mathbf{x}_e = \begin{bmatrix} x - l c_\theta - l_e c_{\theta+\phi} \\ z - l s_\theta - l_e s_{\theta+\phi} \end{bmatrix}, \tag{3.3}$$

$$\dot{\mathbf{x}}_w = \begin{bmatrix} \dot{x} + l_w \dot{\theta} s_\theta \\ \dot{z} - l_w \dot{\theta} c_\theta \end{bmatrix}, \quad \dot{\mathbf{x}}_e = \begin{bmatrix} \dot{x} + l \dot{\theta} s_\theta + l_e(\dot{\theta}+\dot{\phi}) s_{\theta+\phi} \\ \dot{z} - l \dot{\theta} c_\theta - l_e(\dot{\theta}+\dot{\phi}) c_{\theta+\phi} \end{bmatrix}. \tag{3.4}$$

Using flat plate theory, the resulting aerodynamic forces on the vehicle can be approximated

by

$$\alpha_w = \theta - \tan^{-1}(\dot{z}_w, \dot{x}_w), \quad \alpha_e = \theta + \phi - \tan^{-1}(\dot{z}_e, \dot{x}_e) \tag{3.5}$$

$$\mathbf{F}_w = \frac{1}{2}\rho |\dot{\mathbf{x}}_w|^2 S_w \big(C_L(\alpha_w) + C_D(\alpha_w)\big)\mathbf{n_w} \tag{3.6}$$

$$\mathbf{F}_e = \frac{1}{2}\rho |\dot{\mathbf{x}}_e|^2 S_e \big(C_L(\alpha_e) + C_D(\alpha_e)\big)\mathbf{n_e}, \tag{3.7}$$

where α_w and α_e are the angles of attack of the wing and elevator, respectively, ρ is the density of air, and S_w and S_e are the surface areas of the wing and tail control surfaces, respectively. Finally, the dynamics are given by

$$m \begin{bmatrix} \ddot{x} \\ \ddot{z} \end{bmatrix} = \mathbf{F}_w + \mathbf{F}_e - \begin{bmatrix} 0 \\ mg \end{bmatrix} \tag{3.8}$$

$$I\ddot{\theta} = \begin{bmatrix} l_w \\ 0 \end{bmatrix} \times \mathbf{F}_w + \begin{bmatrix} -l - l_e c_\theta \\ -l + l_e s_\theta \end{bmatrix} \times \mathbf{F}_e. \tag{3.9}$$

3.2 Radial Basis Function Augmentation

To improve the aircraft model, an effort was made to augment the aircraft lift and drag coefficients with Guassian radial basis functions, using the aforementioned flat plate glider as a baseline. To achieve this, data were collected by launching the aircraft approximately 50 times over a range of initial velocities from 6 to 8 m/s using a set of optimal elevator trajectories computed for the flat plate model. To obtain the aircraft accelerations, the position measurements produced by the Vicon motion capture system were differentiated twice and filtered acausally.

Once all the data were collected, the accelerations predicted by flat plate theory were subtracted from the aircraft's accelerations. These residual accelerations were then assumed to contribute to residual lift, drag, and moment coefficients of the entire aircraft. In other words, residual aerodynamic coefficients $C_{l,r}$, $C_{d,r}$ and $C_{m,r}$ were modeled as functions of both wing angle of attack and elevator position, using Gaussian radial basis func-

tions specified on a two dimensional grid over a range of wing angle of attacks from $\mu_{\alpha_w,k} \in [0, \pi]$, elevator angles from $\mu_{\phi,k} \in [-\frac{\pi}{4}, \frac{\pi}{4}]$, and a fixed covariance $\Sigma = \text{diag}\begin{bmatrix}0.1 & 0.1\end{bmatrix}$, such that:

$$C_{i,r} = \sum_{k=1}^{N} \psi_k e^{-\frac{1}{2}\left\|\begin{bmatrix}\alpha_w - \mu_{\alpha_w,k} \\ \phi - \mu_{\phi,k}\end{bmatrix}\right\|_{\Sigma}^{2}}. \tag{3.10}$$

Here ψ_k represent the radial basis function magnitudes, which can be fit using regularized least squares to build a more accurate aircraft model. Figure 3-3 compares the resulting model's aerodynamic coefficients with the data post-processed from Vicon.

Chapter 4

Local Linear Feedback

In this chapter, I describe the feedback method first proposed in [86] to stabilize the aircraft and I also describe some alternatives to controlling the aircraft at high angles of attack which are insufficient for achieving fixed-wing perching. In addition, I highlight some of the shortcomings of the approach presented in [86], notably local nature of the control design.

4.1 Post-Stall Control Approaches

In the aerospace controls community, substantial work has been done to design control systems for high performance aircraft in post-stall flight. Initially, much of the work done in this area involved gain scheduling, where linear controllers were generated for multiple operating conditions and the control gains were changed based on the state of the gain scheduled variable [90]. In an effort to improve on this design, dynamic inversion or feedback linearization was applied to transform the nonlinear aircraft dynamics into a stable linear system [41, 91, 11]. In [1, 10], the authors applied robust controller synthesis to account for dynamic uncertainties and, in [19], the authors augmented dynamic inversion with adaptive control to improve the high angle attack performance when the dynamic model changes. Sliding mode methods were also explored for high-angle of attack maneuvers, especially for aggressive missile maneuvers such as in [100].

While these high angle-of-attack approaches are useful for maintaining stability when

the aircraft enters the post-stall regime, they are not necessarily useful for achieving accurate trajectory tracking over all states; these methods can control both the pitch and speed of the aircraft, but are unable to regulate aircraft position to achieve an accurate post-stall landing without substantial thrust vectoring. As mentioned previously, the aircraft investigated in this work has neither thrust or thrust-vectoring capabilities. This is by design in order to ensure that any control method chosen is required to reason intelligently about the aircraft's post-stall aerodynamics and exploit as much post-stall pressure drag as possible (If a propeller is added, it should only be to improve robustness, not to enable the maneuver.). For this reason, it is extremely difficult, if not impossible, to conduct feedback linearization or dynamic inversion. This will be shown in the next section.

4.2 Partial Feedback Linearization vs. Model Predictive Methods

Because the aircraft dynamics are highly nonlinear, and because the perching maneuver involves a large operating range, nonlinear control is essential for achieving successful perching. Two clear nonlinear control methods for addressing this problem are feedback linearization (which has been studied extensively) and model predictive control. In this section, I try to show why, of the two strategies, the model predictive methods are better suited to fixed-wing perching.

To explore the use of feedback linearization on the aircraft, I used the flat-plate baseline model described in Section 3.1 and carried out partial feedback linearization to control the pitch of the vehicle. The flat-plate model is in fact feedback linearizable in this dimension and requires the solving of a fourth order polynomial equation. Figure 4-1 shows the results of applying this control. Notice that although excellent pitch tracking is achieved, as expected, there is no convergence in any of the other states. Moreover, there is a great deal of chatter present in the control input, due mostly to the fact that the controller is limited to a 90 Hz update rate. Although this update rate is twice what is available on the hardware platform, it is not fast enough to prevent large errors in the feedback linearization step.

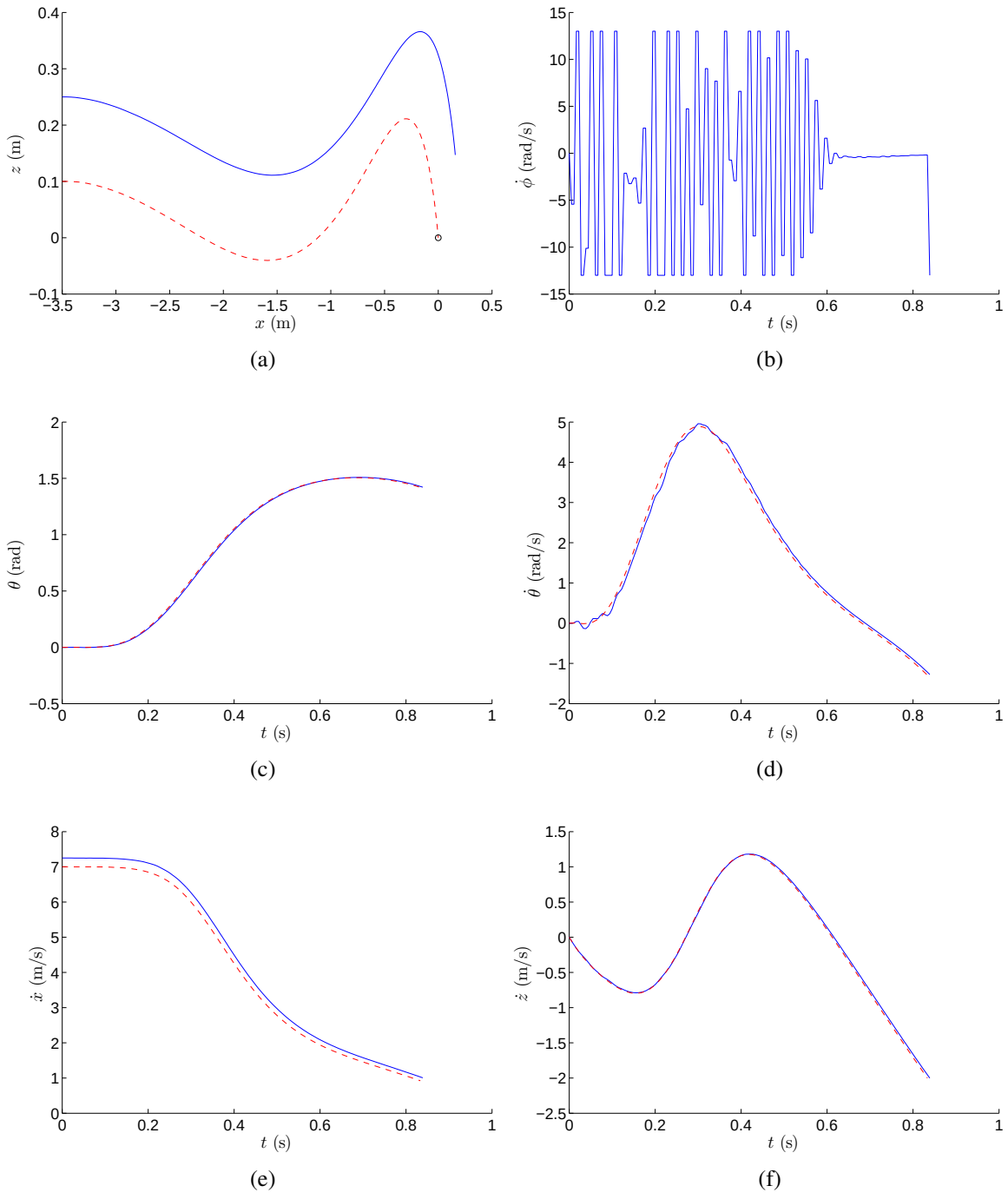


Figure 4-1: Partial feedback linearization to track a pitch trajectory (see **(c)** and **(d)**). As seen in **(e)**, the approach suffers greatly from chatter

To explore dynamic inversion further, I added thrust along the longitudinal axis of the simulated aircraft. With this thrust, I could then apply the contraction method described in [56] to try to control additional states. As can be seen in 4-2, there was an improvement in velocity tracking due to the contraction phenomenon which is enabled by the hierarchical nature of the system and the dissipative nature of the aerodynamic forces. However, as expected, the aircraft was still not able to successfully track position since integration from velocity to position is only marginally stable. Moreover, the control input chatter still existed.

Nonlinear Model Predictive Control (NMPC) or real-time trajectory optimization is an alternative to feedback linearization which has shown promise for controlling highly underactuated systems [55, 42]. However, these methods can be challenging to apply on real systems, especially at real-time rates for systems with large state spaces. And while there has been some effort to overcome this difficulty through fast trajectory generation methods which exploit differential flatness, a differentially flat system is required to apply them [74].

One viable alternative to NMPC is planning an optimal nominal trajectory offline and stabilizing that trajectory with a local linear controller, such as a time-varying linear quadratic regulator (TVLQR). This is the method proposed in [86], the results of which can be seen in Figure 4-3. Note that, while the aircraft does not achieve perfect trajectory tracking, it does in fact reach the desired goal state, due to a high cost on the final state. In the following sections, I describe my method for generating this local controller and compare its results against NMPC.

4.3 Optimal Trajectory Design

The first step for applying both NMPC and TVLQR involves finding a nominal trajectory for the perching maneuver. Despite the relative complexity of the aircraft dynamics, standard tools for trajectory optimization work well for designing a nominal, locally optimal trajectory. To find an optimal trajectory for the glider, I used a direct-collocation method [107] implemented using SNOPT [34]. I prefer this direct method for trajectory

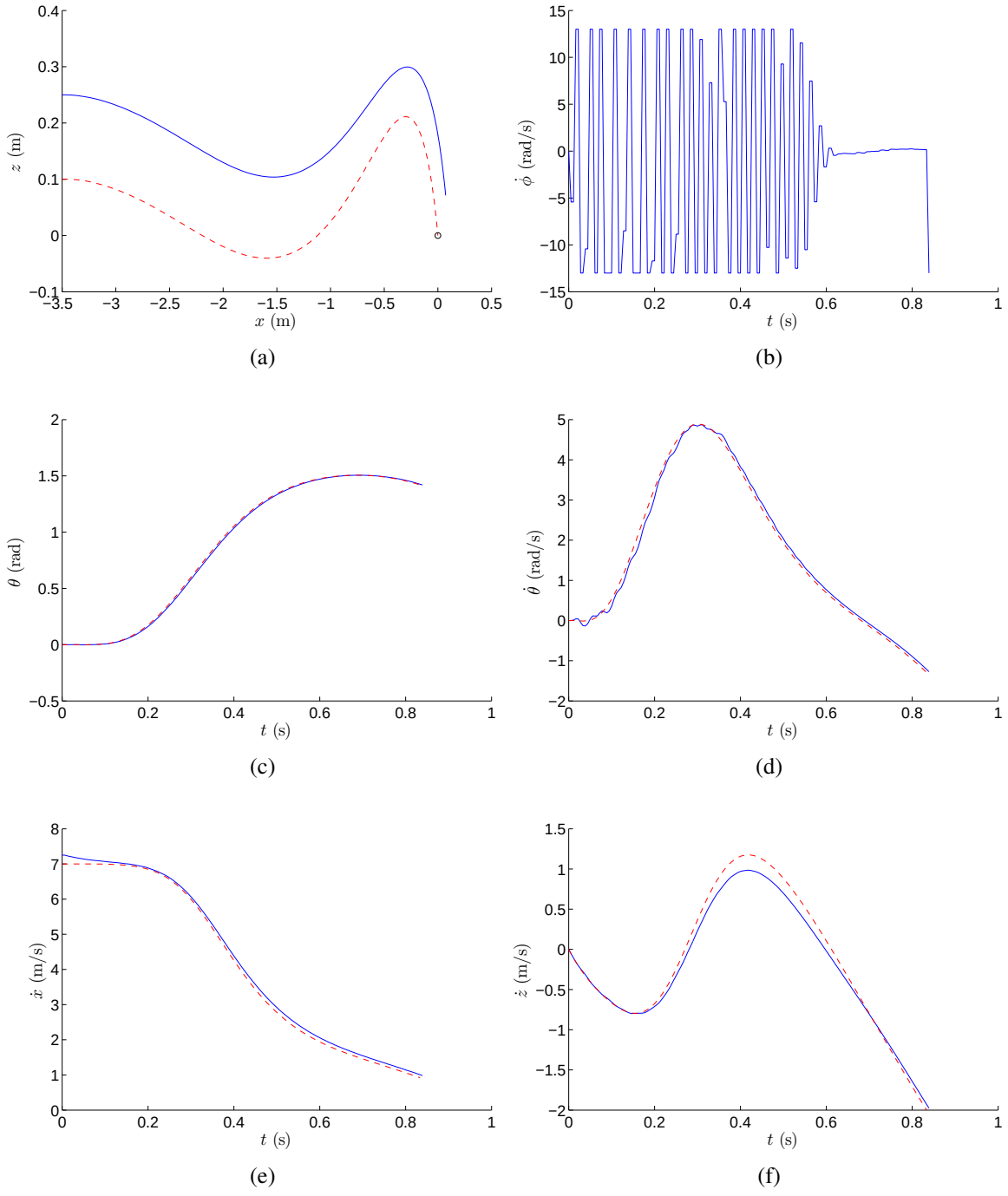


Figure 4-2: Partial feedback linearization with thrust. This approach enables tracking in velocity due to contraction (see **(e)** and **(f)**), though the input still suffers from chatter (see **(b)**)

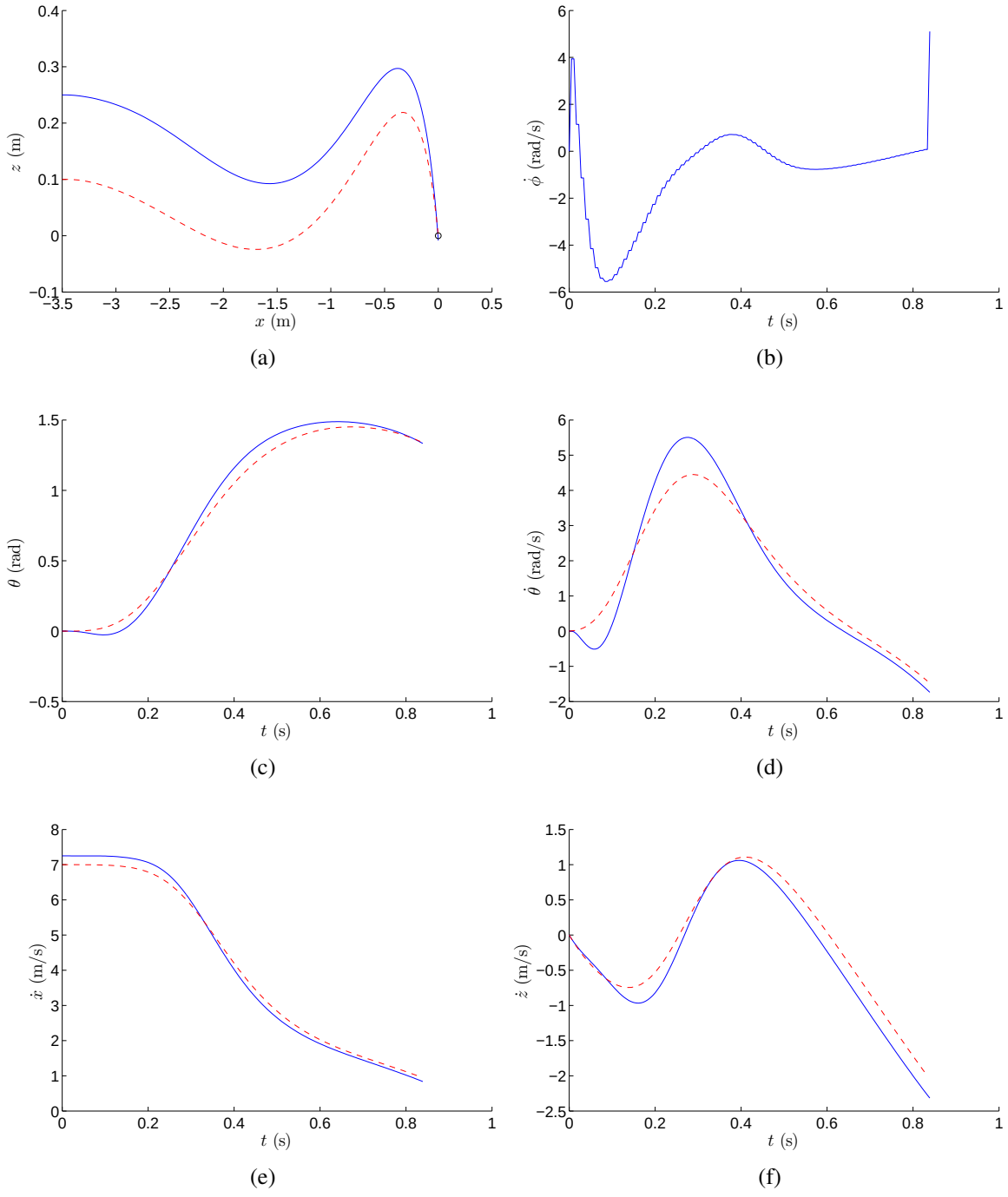


Figure 4-3: Perching results using Time-Varying LQR. Notice that, compared to the partial feedback linearization approach, the TVLQR approach yields better perching results (see **(a)**) as well as a more reasonable control input (see **(b)**).

optimization over the shooting methods based on the adjoint equations [7] since, by treating both the control inputs $\{\mathbf{u}_0, \mathbf{u}_1, ..., \mathbf{u}_{N-1}\}$ and the system states $\{\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_{N-1}\}$ as variables in the optimization routine, it naturally allows for constraints to be placed on the states and the inputs.

To find an optimal trajectory for the perching glider, I solved the following problem:

$$\min_{\mathbf{x}_n, \mathbf{u}_n, h} J = h \sum_{n=0}^{N-1} \mathbf{u}_n^T \mathbf{R} \mathbf{u}_n + \mathbf{x}_n^T \mathbf{Q} \mathbf{x}_n \tag{4.1}$$

$$\text{s.t. } \mathbf{x}_{n+1} = \mathbf{x}_n + h\mathbf{f}(\mathbf{x}_n, \mathbf{u}_n). \tag{4.2}$$

In the equation above, J is a cost function which places a quadratic cost on the state and action of the nominal trajectory using weights $\mathbf{Q}$ and $\mathbf{R}$ respectively. Here, h is the size of the time step, $1/90$ s. By using SNOPT to minimize this quadratic cost function, I was able to find a nominal trajectory $\{\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_{N-1}\}$ and control input $\{\mathbf{u}_0, \mathbf{u}_1, ..., \mathbf{u}_{N-1}\}$ which satisfied the dynamics in 3.1 (represented here by $\mathbf{f}(\mathbf{x}_n, \mathbf{u}_n)$) as well as the initial and final state constraints of the vehicle which are used to shape the maneuver (low final speed, final position at perch). To ensure that the trajectory was sufficiently smooth, I set $\mathbf{Q} = \text{diag}[10, 10, 10, 10, 10, 10, 10]$, $\mathbf{R} = 100$. To shape the perching maneuver, I set the initial state vector to $\mathbf{x}(t_0) = [-3.5, 0.1, 0, 0, 7, 0, 0]$ and the final state constraints to $\mathbf{x}_u(t_f) = [0, 0, \frac{\pi}{2}, \frac{\pi}{8}, 2, 0, \infty]$ and $\mathbf{x}_l(t_f) = [0, 0, \frac{\pi}{8}, -\frac{\pi}{3}, 0, -2, -\infty]$, where $\mathbf{x}_u(t_f)$ and $\mathbf{x}_l(t_f)$ represent upper and lower bounds on the final state, $\mathbf{x}(t_f)$. Lastly, to capture the limited range and speed of the elevator, I placed constraints on ϕ as $\phi \in [-\frac{\pi}{3}, \frac{\pi}{8}]$ and $\dot{\phi}$ as $\dot{\phi} \in [-13, 13]$. The resulting nominal trajectory is shown in Figure 4-4.

4.4 Time-Varying Linear Quadratic Regulator

The NMPC approach mentioned in 4.2 involves running the nonlinear optimization routine described in 4.3 with a receding time horizon at every time step, which can be very computationally demanding. One way to avoid the computational burden of NMPC is to linearize the nonlinear system dynamics about the nominal trajectory described in Section 4.3 and use optimal control to derive a closed-form approximation to the NMPC problem

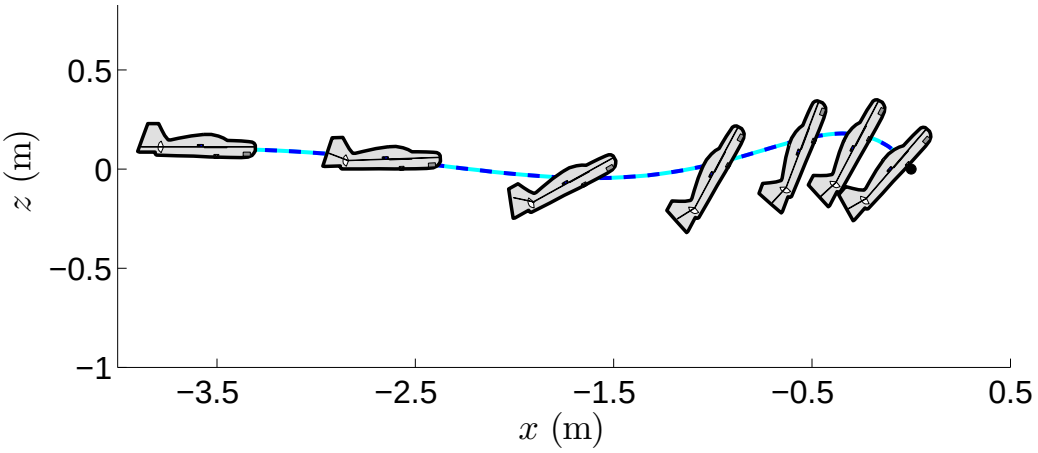


Figure 4-4: Optimized nominal perching trajectory.

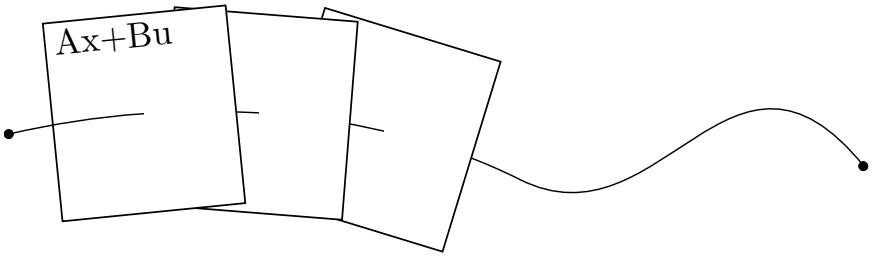


Figure 4-5: A graphical depiction of linearizing about a trajectory in time (Barry 2014).

which is valid in a local neighborhood around the trajectory. For the quadratic cost function used here, this approximation is given by the solution to a time-varying linear quadratic regulator (TVLQR) problem, the basic building block of the LQR-Tree controller.

To compute the TVLQR control law, I write the dynamics of the linearized system as

$$\dot{\bar{\mathbf{x}}}(t) = \mathbf{A}(t)\bar{\mathbf{x}}(t) + \mathbf{B}(t)\bar{\mathbf{u}}(t), \tag{4.3}$$

where $\bar{\mathbf{x}}(t) = \mathbf{x}(t) - \mathbf{x}_0(t)$ and $\bar{\mathbf{u}}(t) = \mathbf{u}(t) - \mathbf{u}_0(t)$. Although it is possible to solve explicitly for a final state constraint [98], in practice I relax the final state constraint with a final state cost, resulting in a cost function of the form:

$$J = \bar{\mathbf{x}}(t_f)^T \mathbf{Q}_f \bar{\mathbf{x}}(t_f) + \int_0^T \left[\bar{\mathbf{x}}(t)^T \mathbf{Q} \bar{\mathbf{x}}(t) + \bar{\mathbf{u}}(t)^T \mathbf{R} \bar{\mathbf{u}}(t)\right] dt. \tag{4.4}$$

To fully define the cost function, I use $\mathbf{Q} = diag([10,10,10,1,1,1,1])$, $\mathbf{R} = .1$, and $\mathbf{Q}_f = diag([400,400,\frac{1}{9},\frac{1}{9},1,1,\frac{1}{9}])$, where $\mathbf{Q}$ was chosen to encourage the vehicle to stay close to

the nominal trajectory (where the linearization is valid) and the cost on $\mathbf{R}$ was chosen so that over a reasonable range of initial conditions, control saturations were limited. The entries of $\mathbf{Q}_f$ were selected by approximating the desired goal region as an ellipsoid, $G = \bar{\mathbf{x}}^T \mathbf{Q}_f \bar{\mathbf{x}}$, so that the maximum distances are defined by $\bar{\mathbf{x}}_{f,max} = [0.05, 0.05, 3, 3, 1, 1, 3]$ where $\mathbf{Q}_f = diag(\bar{\mathbf{x}}_{f,max})^{-2}$. Furthermore, the entries for $\mathbf{Q}$ were also chosen with respect to $\mathbf{Q}_f$ so that the solutions to the Riccati equation remained well conditioned.

Finally, I numerically solve the differential Riccati equation

$$-\dot{\mathbf{S}}(t) = \mathbf{Q} - \mathbf{S}(t)\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}(t) + \mathbf{S}(t)\mathbf{A} + \mathbf{A}^T\mathbf{S}(t) \tag{4.5}$$

backwards in time with

$$\mathbf{S}(t_f) = \mathbf{Q}_f. \tag{4.6}$$

This yields the TVLQR control law

$$\bar{\mathbf{u}} = -\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}(t)\bar{\mathbf{x}}(t), \tag{4.7}$$

and the cost-to-go

$$J = \bar{\mathbf{x}}^T\mathbf{S}(t)\bar{\mathbf{x}}. \tag{4.8}$$

4.5 NMPC vs. TVLQR

The local linear approximation of the dynamics and the locally quadratic approximation of the objective provides a feedback control law which is trivial to compute at run time (evaluating one piecewise polynomial spline which provides the feedback gains, then one matrix multiplication), but one cannot necessarily expect this controller to perform well for initial conditions far away from the nominal trajectory. To quantify this, I conducted exhaustive numerical simulations of the closed-loop systems from a range of initial conditions. Since the state space is ≥ 7 dimensions, this exhaustive evaluation is only possible in small subspaces of initial conditions. Given a particular initial condition, the closed-

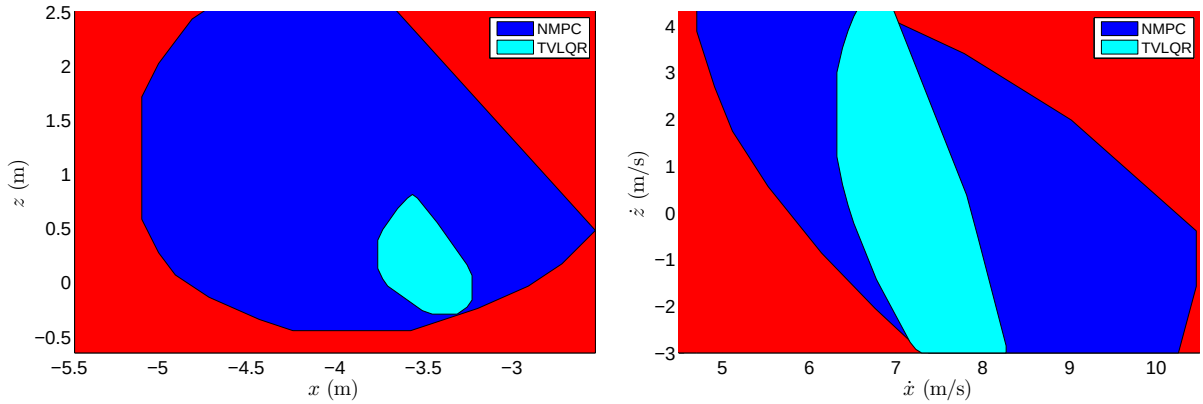


Figure 4-6: Comparison of NMPC with TVLQR via exhaustive simulation from a subspace of initial conditions. Cyan indicates the initial conditions for which the TVLQR controller navigates the aircraft through the goal region, blue indicates the same for NMPC. (a) Initial conditions were densely sampled from $\mathbf{x}(t_0) = [x, z, 0, 0, 7, 0, 0]$, (b) initial conditions were densely sampled from $\dot{\mathbf{x}}(t_0) = [-3.5, 0.1, 0, 0, \dot{x}, \dot{z}, 0]$.

loop system was considered successful if the state of the aircraft during forward simulation entered the goal region around the perch at *any* time in the trajectory (a more generous evaluation than only testing at the specific final time of the nominal trajectory). Figure 4-6 shows the results, which confirms that the nonlinear controller from NMPC significantly outperforms the optimal time-varying linear controller. Unfortunately, this quantitative comparison is limited to the simulation model, as I am unable to evaluate the NMPC controller at real-time rates in the physical experiments due to the processing limitations of the onboard computer.

However, a relatively simple solution presents itself. In order to reproduce the performance of NMPC with a controller that can be evaluated efficiently at run-time, I will compute a small library of TVLQR controllers, each constructed around a different nominal trajectory from NMPC. Each controller will only be valid locally, but even in 7+ dimensions, I hope to fill the (closed and bounded) subspace of initial conditions of interest since each TVLQR controller covers a non-negligible region. The run-time cost then reduces to a one-time evaluation of the initial conditions to determine which controller to run, followed by the cost of evaluating that TVLQR controller. Minimizing the number of controllers in the library will be essential to mitigate this one-time cost, as well as the memory and design-time creation cost of the library. This control library idea, along with

a recipe for constructing an efficient library coverage with minimal controllers, is called "LQR-Trees"[99].

In order to achieve coverage, the LQR-Trees algorithm needs to be able to efficiently *verify* the local controllers, as in Figure 4-6. But this verification must work in the full state-space of the model, not just a subspace of initial conditions; and to be practical it must be much more efficient than the exhaustive simulations. To achieve this, I examine algorithms for producing efficient inner-approximations of these verified regions.

Chapter 5

Trajectory Verification

5.1 Verification Methods

The controls community has produced a number of approaches for verifying the behavior of complex dynamical systems. In [69], the authors made use of game theory and solved a Hamilton-Jacobi-Isaacs partial differential equation by discretizing on a grid to compute the set of reachable states for a continuous dynamical system. In [103], this approach was extended to handle hybrid systems. Besides grid-based methods, some researchers have explored methods of approximating reachable sets through combining numerical simulation with sensitivity analysis [28]. In [23], the authors merged this technique with rapidly exploring random trees to explore the state space of a dynamical system.

Here, I explore an alternative approach to verification which uses sum-of-squares (SOS) optimization [77] to compute an inner approximation of the regions displayed in Figure 4-6, more formally defined as the "backwards reachable set to the goal". This approach has some merits compared to the techniques mentioned above. First of all, this method reasons algebraically about the governing equations, eliminating the need to discretize the system's state space. This requires that the closed-loop system of interest be described or approximated in closed-form using polynomials. Moreover, because computationally costly PDE solvers are replaced with semi-definite program solvers, the SOS optimization routines scale much more reasonably with state than other methods. Since the system of interest has a large state space but is described with a smooth closed-form expression, I

focus on the sum-of-squares approach here.

5.2 Sum-of-Squares Verification

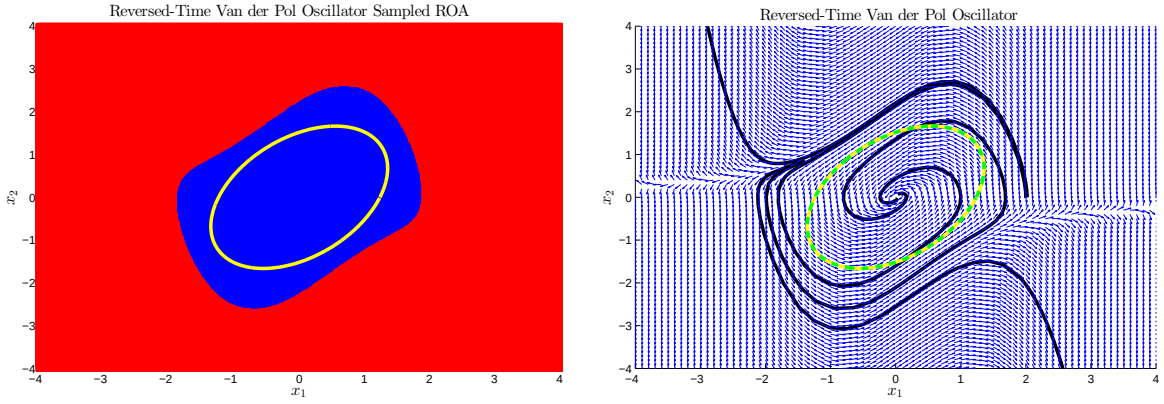


Figure 5-1: The reverse-time Van der Pol Oscillator is a system commonly used to test region of attraction analysis methods. (a) Shows the the true region of attraction for the reverse-time Van der Pol (blue), obtained through exhaustive simulation. The ellipse (yellow) depicts the bounds of the largest quadratic Lyapunov function found in this region, also found through sampling the quadratic form $\mathbf{x}^T\mathbf{P}\mathbf{x}$. (b) Figure shows the vector field for the reverse-time Van der Pol oscillator (blue) along with a few system trajectories (black) and the largest quadratic region of attraction found using the sum-of-squares (SOS) approach (green).

Verification using sum-of-squares optimization is accomplished by searching for a verification certificate in the form of a Lyapunov function [77], which I will denote $V(\mathbf{x})$. For dynamical systems with polynomial vector fields, SOS makes it possible to verify the Lyapunov conditions ($V > 0, \dot{V} < 0$) for a candidate Lyapunov function as a convex optimization [77]. It is also possible to search for a Lyapunov function which satisfies these conditions, to search for a Lyapunov function which proves stability in a bounded positively invariant region, and/or to simultaneously optimize a controller in order to maximize a verified positively invariant region by solving a series of convex optimizations [59, 80, 94, 111].

Consider the time-invariant case where $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$; in this case, computing the backwards reachable set to a fixed-point is equivalent to computing the region of attraction to that fixed point. I restrict my search to positive-definite Lyapunov functions, $V(\mathbf{x})$, and define

the region of interest, B, as a sub-level set of this positive function:

$$B = \{ \mathbf{x} \mid V(\mathbf{x}) < \rho \},$$

for some positive scalar ρ. If I can find a positive V such that $\forall \mathbf{x} \in B$ that:

$$\dot{V} = \frac{\partial V(\mathbf{x})}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}) < 0,$$

then this verifies that B is a positively invariant region of the closed-loop system, and that all initial conditions in B will eventually reach the origin. In other words, B is an inner-approximation of the region of attraction to the origin. If $\mathbf{f}(\mathbf{x})$ is polynomial and I restrict my search to polynomial Lyapunov functions, then the search for this certificate of regional stability is reduced to a search over positive polynomials.

In SOS optimization, the test for positivity is replaced with the condition that the polynomial is a sum-of-squares. To isolate the analysis to the region defined by the sub-level set of B, I apply a technique called the S-procedure [77] which is analogous to the use of Lagrange multipliers in constrained optimization. Using Σ to denote the sum-of-squares polynomials over $\mathbf{x}$, I write

$$V \in \Sigma \tag{5.1}$$

$$-\dot{V} + s_1(V - \rho) \in \Sigma \tag{5.2}$$

$$s_1 \in \Sigma, \tag{5.3}$$

where s_1 is an additional polynomial that serves as a multiplier for the S-procedure. I seek to maximize the size of B in order to find the largest inner-approximation of the true region of attraction; here I approximate volume by enforcing a scale on V and maximizing ρ. In order to optimize these polynomial constraints over the decision variables, V,ρ, s_1, I carry out two steps of bilinear alternations [89].

If I let $V = \bar{\mathbf{x}}^T \mathbf{S} \bar{\mathbf{x}}$, for some $\mathbf{S} = \mathbf{S}^T \succ 0$, then the constraint in equation (5.1) is satisfied trivially. I can now perform the optimization in the following steps:

STEP 1:

$$\begin{aligned} \underset{s_1,\gamma}{\text{maximize}} \quad & \gamma \\ \text{subject to} \quad & \gamma > 0, \\ & -\dot{V} + s_1(V-\rho) - \gamma \in \Sigma. \\ & s_1 \in \Sigma \end{aligned} \tag{5.4}$$

STEP 2:

$$\begin{aligned} \underset{V,\rho}{\text{maximize}} \quad & \rho \\ \text{subject to} \quad & \rho > 0, \\ & -\dot{V} + s_1(V-\rho) \in \Sigma. \\ & s_1 \in \Sigma \end{aligned} \tag{5.5}$$

These steps then repeat until convergence is observed in ρ.

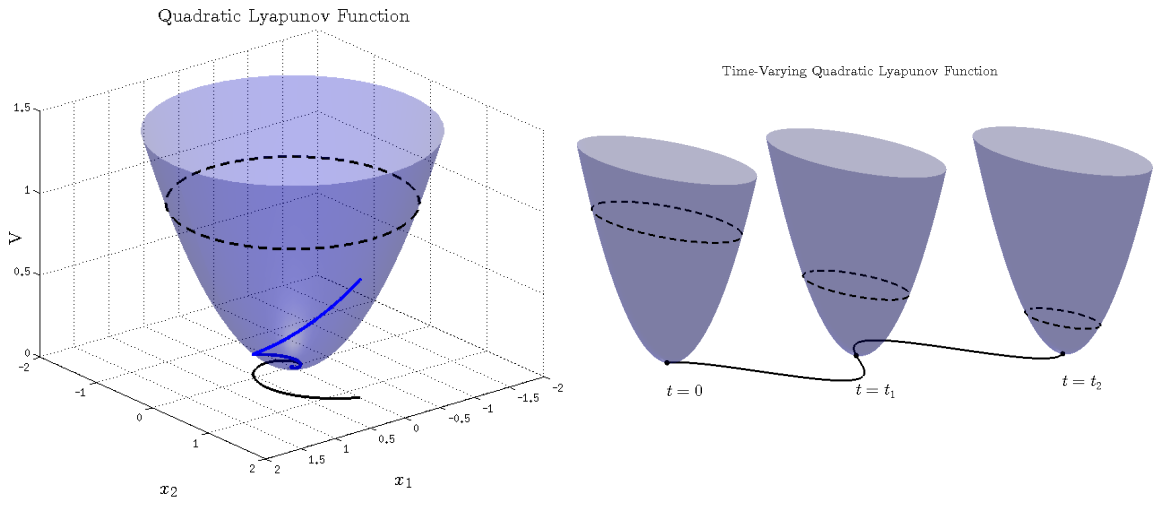
5.3 Verification along Trajectories

The approach presented in the preceding section can also be applied to computing the backwards reachable set to a goal region along finite-time trajectories. To do this, I build on the work done in [102], but instead of only searching over a single rescaling of the level set, I search over the entire Lyapunov function. My approach is as follows:

First, I generalize the Lyapunov function to be a positive function of time as well as state, $V(t,\mathbf{x})$. Then, rather than requiring stability to some fixed-point, I search for a bounded positively invariant region

$$B(t) = \{\mathbf{x} | V(t,\mathbf{x}) \leq \rho(t)\},$$

which can be verified by ensuring that $\dot{V} < \dot{\rho}$ for all states on the boundary of the region, $V = \rho$. Following [99, 102], I refer to these bounded positively invariant regions "funnels".



(a) (b)

Figure 5-2: (a) A depiction of a quadratic Lyapunov function at a single operating point for a two-dimensional system. The black dotted line represents the "One-Level Set", the black solid line represent the system trajectory in state space and the blue sold line represents the decreasing value of the Lyapunov function over time. (b) A depiction of a time-varying quadratic Lyapunov function along a trajectory. The black dotted lines represents the ρ level set, which can be optimized to find the largest invariant set.

I am then able to reformulate the constraints as:

$$V(t,\mathbf{x}) = \rho(t) \implies \dot{V}(t,\mathbf{x}) - \dot{\rho}(t) \leq 0, \tag{5.6}$$

where

$$\dot{V}(t,\mathbf{x}) = \frac{\partial V(t,\mathbf{x})}{\partial \mathbf{x}}\dot{\mathbf{x}} + \frac{\partial V(t,\mathbf{x})}{\partial t}. \tag{5.7}$$

Using the generalized S-Procedure, I can now write

$$-\dot{V}(t,\mathbf{x}) + \dot{\rho}(t) - \mu(t,\mathbf{x})(V(t,\mathbf{x}) - \rho(t)) \geq 0, \tag{5.8}$$

where $\mu(t,\mathbf{x})$ is a polynomial Lagrange multiplier. To further simplify the problem, I sample in time to eliminate t from the above expression, as recommended in [102]. This results

in N positivity constraints

$$-\dot{V}(t_i,\mathbf{x})+\dot{\rho}(t_i)-\mu_i(\mathbf{x})(V(t_i,\mathbf{x})-\rho(t_i))\geq 0, \tag{5.9}$$

where $i=1,2,3...N$. To evaluate $\rho(t)$ at $t=t_i$, I choose $\rho(t)$ to be a piecewise linear polynomial such that

$$\rho(t)=\rho_i+(t-t_i)\dot{\rho}_i, \quad i=\text{floor}\left(\frac{t-t_0}{\Delta t}\right) \tag{5.10}$$

$$\dot{\rho}_i=\frac{\rho_{i+1}-\rho_i}{\Delta t}. \tag{5.11}$$

I also let $V(t,\mathbf{x})=\bar{\mathbf{x}}^T\mathbf{S}(t)\bar{\mathbf{x}}$ where $\mathbf{S}(t)=\mathbf{S}_0(t)+\Phi(t)$, $\mathbf{S}_0(t)$ comes from the differential Riccati equation, and I let $\Phi(t)$ be a piecewise polynomial such that

$$\Phi(t)=\Phi_i+(t-t_i)\dot{\Phi}_i, \quad i=\text{floor}\left(\frac{t-t_0}{\Delta t}\right) \tag{5.12}$$

$$\Phi_i^T=\Phi_i\succ 0, \quad \dot{\Phi}_i=\frac{\Phi_{i+1}-\Phi_i}{\Delta t}. \tag{5.13}$$

This ensures that $\mathbf{S}^T(t)=\mathbf{S}(t)\succ 0$. Including $\mathbf{S}_0$ in the parametrization of $\mathbf{S}(t)$ is extremely important for being able to find a feasible initialization for the bilinear SOS optimization when $\Phi(t)$ is zero.

To search for Φ_i and ρ_i, I once again replace the test for positivity with the test for sum-of-squares positivity. I then hold the scale of $\mathbf{S}(t)$ fixed and use bilinear alternations to search iteratively over ρ_i and μ_i in a manner similar to the steps shown in equations (6.19) and (6.20). In this instance, the alternations terminate when $\sum_{i=1}^{N}\rho_i$ attains a local maximum.

It is this ability to verify along trajectories that allows the application of these verification methods to the perching maneuver. The "funnels" found for this system can be seen in Figure 5-3. I also plot the case where the Lyapunov function is only parametrized by a re-scaling of $\bar{\mathbf{x}}^T\mathbf{S}_0(t)\bar{\mathbf{x}}$, as proposed in [102], to show the advantage over of parameterizing the Lyapunov function by the entire $\mathbf{S}$-matrix using the Φ_i terms.

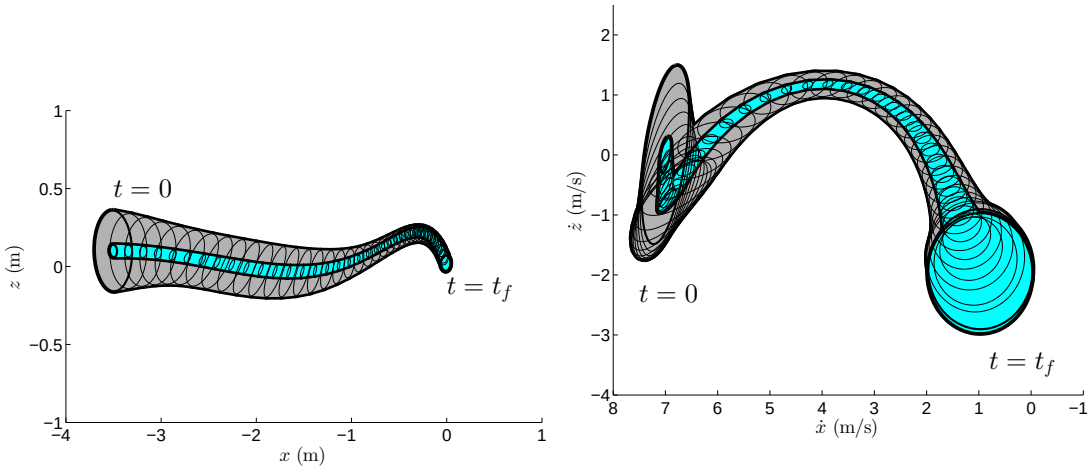


Figure 5-3: Slices of the high-dimensional funnel in the x-z and $\dot{x}$-$\dot{z}$ plane obtained by optimizing only over a rescaling of the level set as done in [102] (cyan), compared with the respective slices of the funnel obtained by optimizing over the entire Lyapunov function as done in [72] (gray).

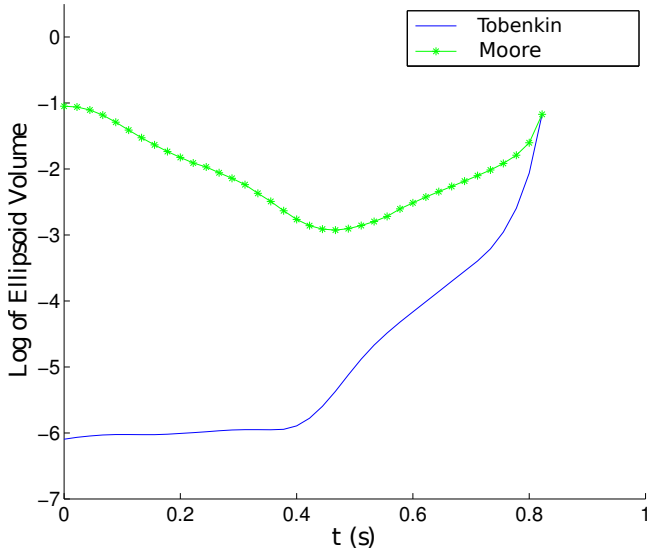


Figure 5-4: Comparison of the volumes of the funnels obtained from optimizing over $\rho(t)$ compared with optimizing over $\Phi(t)$.

Table 5.1: Funnel Comparison

Method	μ-step	V-step	No. Iter.	Total t	Vol.
Tobenkin	21.7s	13.9s	7	250 s	3.17×10^{-4}
Moore	30.3s	40.9s	14	997 s	1.47×10^{-2}

Table 5.2: This table shows that even though more variables are required to parameterize the entire S-matrix of the quadratic Lyapunov function, the required computation time is not substantially greater than parameterizing the Lyapunov function by a scalar multiplier.

5.4 SOS Verification Compared to Exhaustive Numerical Simulation

To test the funnels, I repeated the exhaustive simulation approach to verification which is more exact but restricted to two dimensional slices. Since the funnel is defined over the entire finite time interval of the trajectory, I repeated these simulations at ten time slices along the nominal trajectory. Sampling within the specified slices, I then ran the LQR controller forward in time from each slice and tested if it reached the goal. The results can be seen in Figure 5-5. Given all of the potential conservatism taken along the length of the trajectory, the resulting quadratic approximation of the funnels is surprisingly tight. The fact that they verify a relatively large subset of initial conditions makes them very useful in producing a library of controllers.

5.5 LQR-Tree Controller

As demonstrated in [72], these verified regions can be combined together to form an LQR-Tree to cover a sufficiently large region of state space. This is achieved by sampling randomly from a set of initial conditions and checking if the sample is already in the verified region. If it is, then the set of initial conditions is re-sampled. However, if the sample is not in the verified region, then a new trajectory is designed, where the trajectory's terminal constraints are specified to be along the tree's already existing trajectories. Once a trajectory from the initial condition to the goal is found, this trajectory is verified and a new funnel is added to the tree. This process repeats until no sampled initial conditions are outside the tree. An LQR-Tree for the glider is shown in Figure 5-6.

Notice how the LQR-Tree in Figure 5-6 verifies a much larger range of initial velocities when compared to the region verified by any one funnel. It is also interesting to note that, although attempts were made to reconnect initial conditions to the nearest point along the original nominal trajectory, the algorithm still connected all the new trajectories to the goal region. This is most likely because, given the aircraft's extremely short flight time, it is more optimal for the trajectories to terminate at the goal region rather than at some earlier

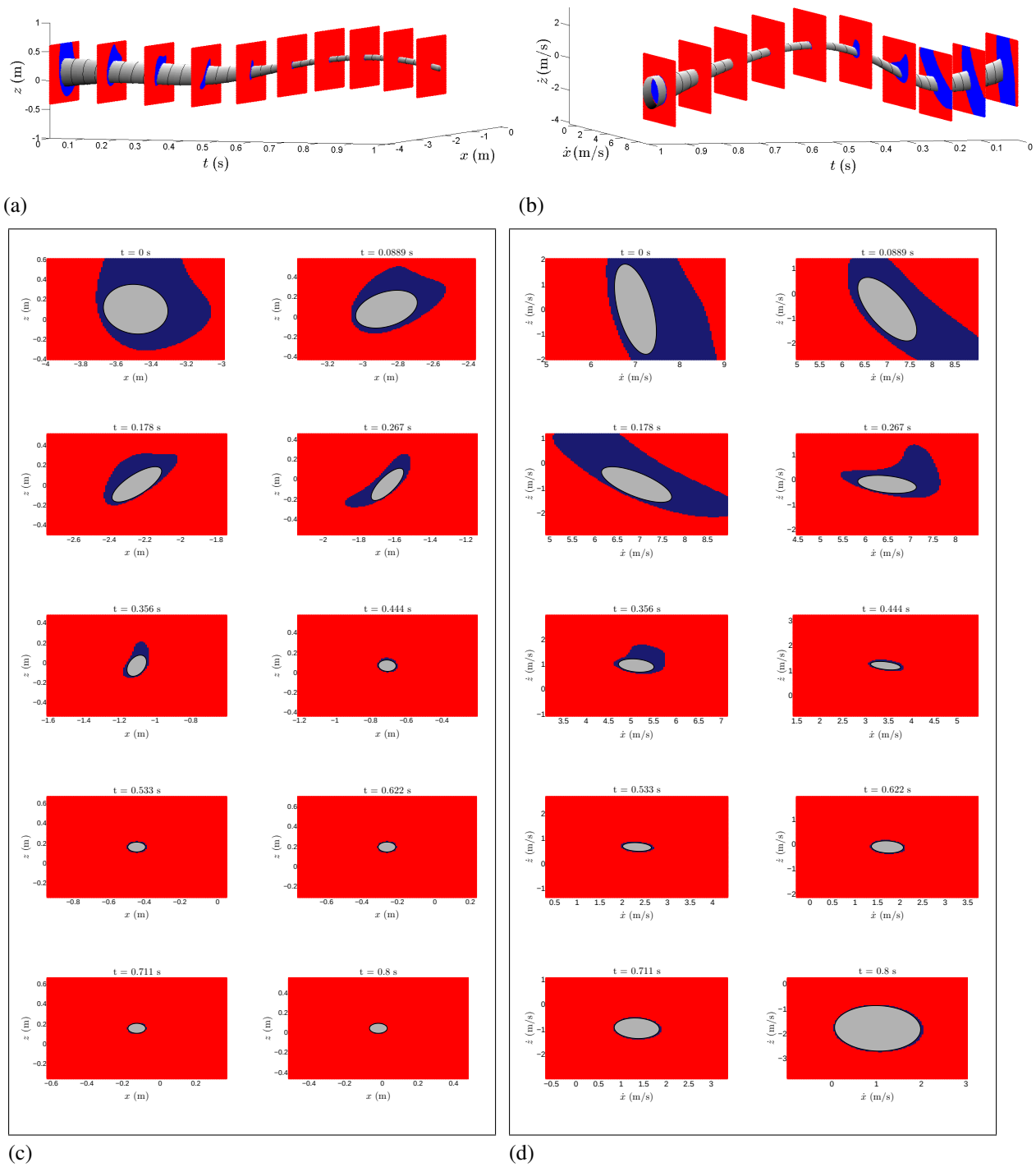


Figure 5-5: **(a-b)** Comparison of SOS funnel's x-z and $\dot{x}$-$\dot{z}$ slices with the positively invariant set found by simulating the full non-polynomial, nonlinear system forward from the relevant initial conditions. **(c-d)** 2D plots of the x-z and $\dot{x}$-$\dot{z}$ funnel slice time cross-sections found in (a-b). Red represents simulated trajectories which failed to reach the goal region, blue represents those that succeeded, and gray represents the final SOS funnel.

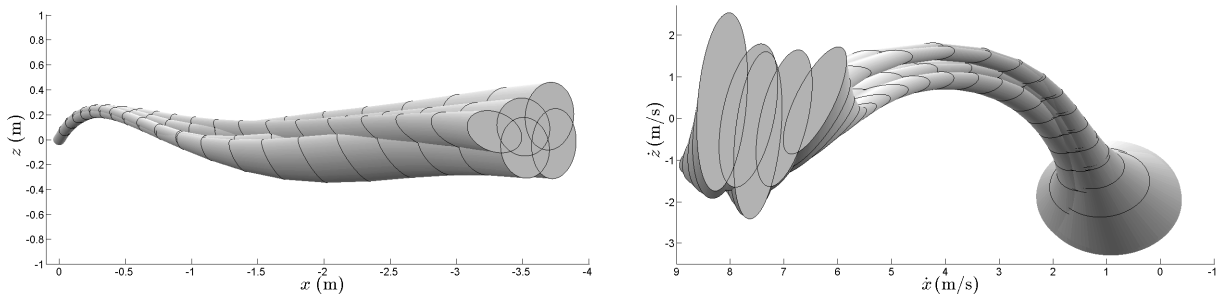


Figure 5-6: Slice of the high-dimensional LQR-Tree in the x-z plane (left) and the $\dot{x}$-$\dot{z}$ plane (right). Notice the increased initial condition coverage

point along the nominal trajectory.

Chapter 6

Verification of Uncertain Systems

Although I have demonstrated that the LQR-Tree approach can increase the set of initial conditions for which the UAV can perch successfully, I have not addressed model uncertainty. In Figure 6-1, I show simulations of the aircraft from a range of initial velocities with a parametric offset in the x-accelerations and z- accelerations. Notice that as the parametric uncertainty increases, the admissible initial condition set begins to decrease in volume. This is because the parametric uncertainty adversely affects perching performance, requiring the aircraft to start closer to the nominal initial conditions to achieve successful perching. In the following sections, I discuss methods for incorporating these parametric and dynamic uncertainties in to the sum-of-squares verification framework.

6.1 Prior Work

A number of approaches have been explored for robust region of attraction analysis. Prior to the use of SOS programming, LMI methods were applied to try to estimate the robust regions of attraction for uncertain polynomial systems [101]. However, with the advent of SOS methods, researchers began to explore finding true regions of attraction for polynomials with bounded uncertain inputs as well as parametric uncertainty [104, 106, 105]. These approaches have also been applied to hybrid systems [76] as well as to finite-time polynomial systems with variable goal regions [61]. Here, I carry out robust verification for a highly underactuated system by exploring the verification of the closed-

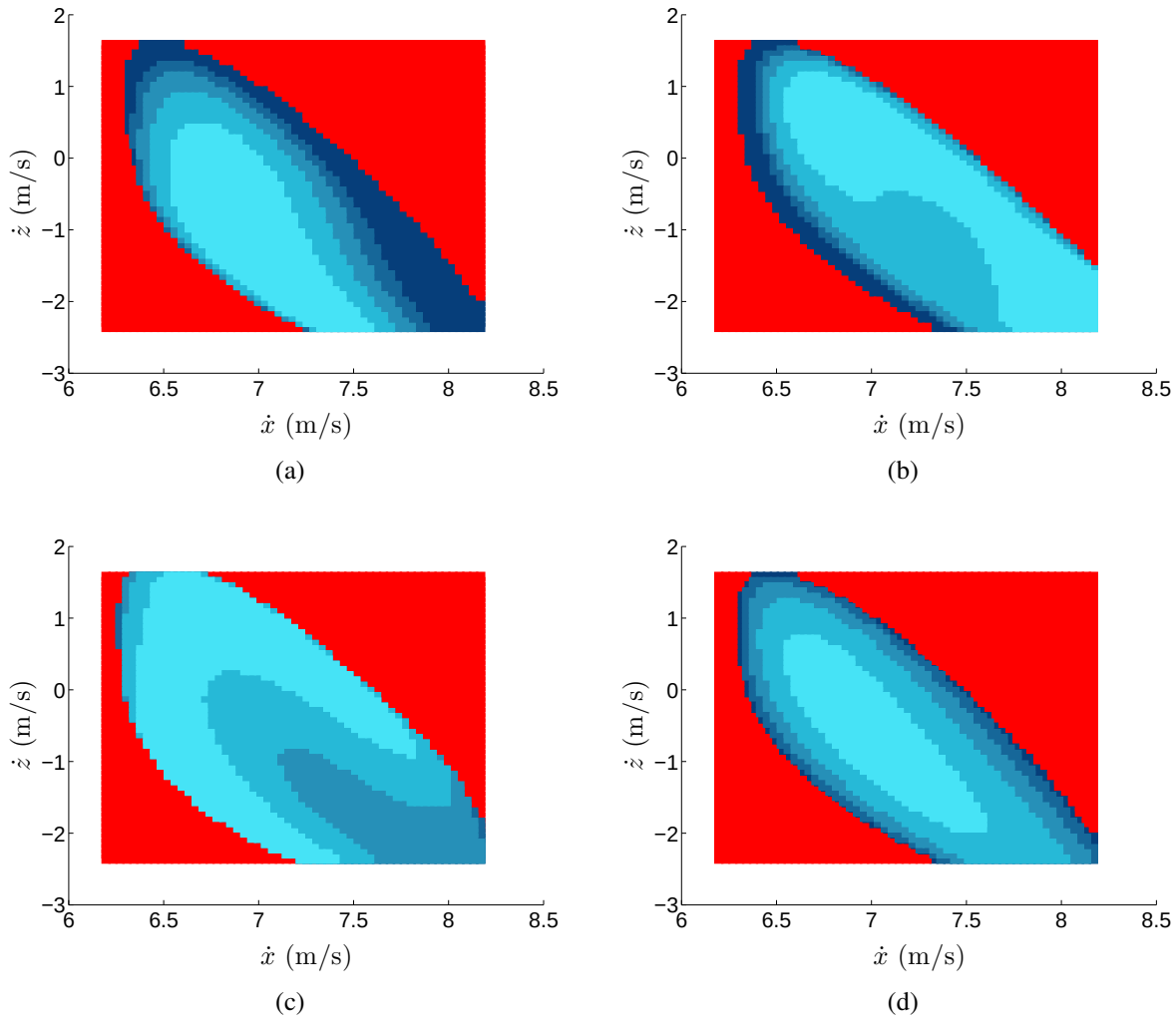


Figure 6-1: These figures display the admissible initial velocity conditions for the aircraft when an uncertain constant offset is added to the x and z accelerations. The acceration offsets are applied from 0-1 $\frac{m}{s^2}$ starting with the largest admissible offset in that range and descreasing in increments of 0.1 $\frac{m}{s^2}$. Lighter blues represent a larger acceleration offset. (a)-(b) represent a postive and negative offset in x-acceleration respectively and (c)-(d) represent a positive and negative offset in z-acceleration respectively. Notice how the basins decrease with increasing acceleration error.

loop perching glider system under both parametric and dynamic uncertainty.

6.2 Robust Verification

In this thesis, I divide model uncertainty in to two classes: parametric uncertainty and dynamic uncertainty [105]. Parametric uncertainty exists when there is a constant parame-

ter in the system dynamics that is unknown, or is only known to be within some bounded set. Dynamic uncertainty, however, is uncertainty which varies with time and arises due to unmodeled dynamics. In the following sections, I present verification approaches for both uncertainty types.

6.2.1 Parametric Uncertainty

One common means of representing uncertainty in dynamical systems is by modeling that uncertainty as an unknown parameter. The verification methods described in the previous chapter can easily be applied to situations where parametric uncertainty exists. All that is required is to add extra variables to the SOS program, one for each unknown parameter.

Consider the uncertain system dynamics

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, \theta), \tag{6.1}$$

where θ is a vector of uncertain constant parameters.

I can then write the following "common" Lyapunov function as

$$V(t, \mathbf{x}, \theta) = V_0(t, \mathbf{x}) + \theta^T \Gamma(t) \theta, \tag{6.2}$$

where $V_0 = \mathbf{x}^T S(t) \mathbf{x}$. Since θ is constant

$$\dot{V}(t, \mathbf{x}, \theta) = \frac{\partial V_0(t, \mathbf{x})}{\partial \mathbf{x}} \dot{\mathbf{x}} + \frac{\partial V_0(t, \mathbf{x})}{\partial t}. \tag{6.3}$$

Once again, I can write the constraint

$$V(t, \mathbf{x}, \theta) = \rho(t) \implies \dot{V}(t, \mathbf{x}, \theta) - \dot{\rho}(t) \leq 0. \tag{6.4}$$

and search for an invariant set using SOS. This is achieved through the same two step bilinear optimization routine described in Section 5.2. To initialize this bilinear optimization program, $\Gamma(t)$ must be made very large so that, to start, the uncertain parameter is constrained to a set of small values. A robust funnel for a constant offset in the glider's

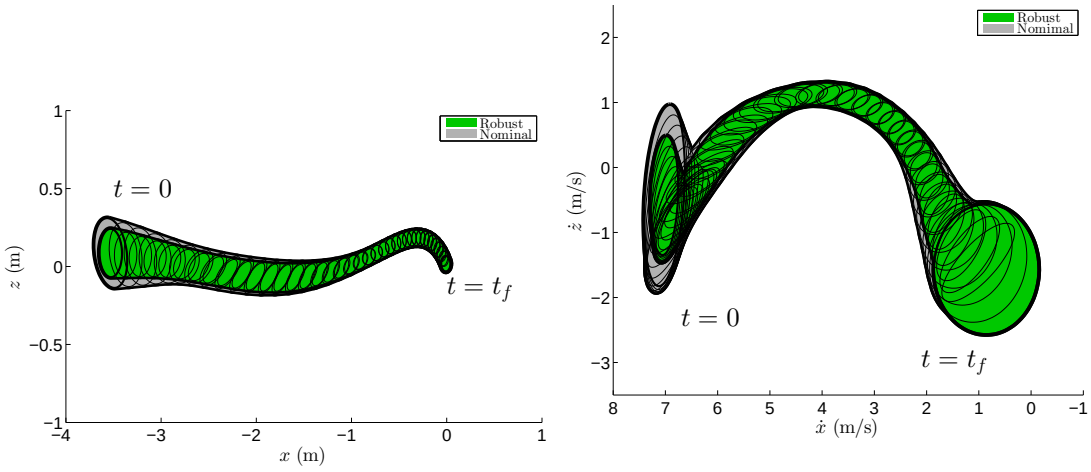


Figure 6-2: Slices of the high-dimensional funnel in the x-z and $\dot{x}$-$\dot{z}$ plane with (green) and without (gray) accounting for uncertainty. For the robust funnel, the uncertainty is modeled as a constant offset on the x-acceleration and reaches a maximum offset value of about 0.22 $\frac{m}{s^2}$.

x-acceleration can be observed in Figure 6-2.

One feature of this approach is that the parameter θ is part of the Lyapunov function. This means that when evaluating the Lyapunov function, one must use the largest possible value for θ, which has the potential to drastically reduce the volume of the funnel. In Figure 6-2, θ is set to zero to show the largest possible funnel.

Notice also that this particular approach can not be directly extended to the case where dynamic uncertainty exists, because, in the case of dynamic uncertainty, the derivative of the uncertainty is unknown. Here, because the analysis assumes parametric uncertainty, the derivative of the uncertainty is zero, and the above formulation holds.

6.2.2 Dynamic Uncertainty

As mentioned above, the verification method presented in the previous section will not hold if the derivative of the uncertainty is unknown. However, this SOS verification approach can also be extended to situations where dynamic uncertainty exists, for instance, in the case of some bounded, time-varying input noise on the system accelerations. To do this, I search for a common Lyapunov function by using additional multipliers to constrain the uncertainty to a bounded set. I also reduce the number of decision parameters in the SOS formulation by making the Lyapunov function only depend on state (i.e., $\Gamma = 0$). In

principle, the Lyapunov function could also depend on the uncertainty if multipliers were used to bound the derivative of the uncertainty as well.

The constraints for this formulation can be written as

$$\forall \theta_i \in [\theta_{i,min}, \theta_{i,max}], \quad V(t,\mathbf{x}) = \rho(t) \implies$$
$$\dot{V}(t,\mathbf{x},\theta) = \frac{\partial V(t,\mathbf{x})}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x},\theta) + \frac{\partial V(t,\mathbf{x})}{\partial t} < \dot{\rho}(t) \tag{6.5}$$

where θ_i is the i^{th} bounded uncertain parameter and $\rho(t)$ is the value of the level set of interest. As before, SOS can be used to search for a positively invariant set about the system's nominal trajectory.

A challenge with this approach is finding a suitable initial Lyapunov function with which to initialize the SOS optimization routine. The Lyapunov function used to initialize the SOS optimization routine is no longer necessarily valid for a desired uncertainty bound. For this reason, I use a three step bilinear optimization routine, which allows for maximizing the size of the uncertainty bounds. The first two steps of this optimization routine are similar to the two steps mentioned in the prior section. In the first step, I initialize the search for the multipliers with the uncertainty constrained to be zero. Then, in the third step, I begin to grow the uncertainty bound. These three steps continue until the uncertainty bounds reach the desired values or the optimization fails to find a feasible solution. It is also important to note, that, in the third step, I only allow the level set and the uncertainty bounds to change. This preserves the funnel volume increase achieved in the second step due to searching over the entire Lyapunov function and only rescales the resulting Lyapunov function to increase the uncertainty bound. These steps are summarized as follows:

STEP 1:

$$\begin{aligned} \underset{s_1, s_{i+1}, \gamma}{\text{maximize}} \quad & \gamma \\ \text{subject to} \quad & \gamma > 0, \\ & -\dot{V} + \dot{\rho} - s_1(V-\rho) + \sum_{i=1}^{N} s_{i+1}(\theta_i^2 - \delta_i) - \gamma \in \Sigma[\mathbf{x}]. \end{aligned}$$

STEP 2:

$$\begin{aligned} \underset{\rho,V}{\text{maximize}} \quad & \sum \rho \\ \text{subject to} \quad & \rho > 0, \\ & -\dot{V} + \dot{\rho} - s_1(V-\rho) + \sum_{i=1}^{N} s_{i+1}(\theta_i^2 - \delta_i) \in \Sigma[\mathbf{x}]. \end{aligned}$$

STEP 3:

$$\begin{aligned} \underset{\rho,\delta}{\text{maximize}} \quad & \sum_{i=1}^{N} \delta_i \\ \text{subject to} \quad & \rho > 0, \\ & \delta_i > 0, \\ & -\dot{V} + \dot{\rho} - s_1(V-\rho) + \sum_{i=1}^{N} s_{i+1}(\theta_i^2 - \delta_i) \in \Sigma[\mathbf{x}]. \end{aligned}$$

For the glider, I applied the bounded uncertainty term as a bounded process noise on the system dynamics. The resulting robust funnel is shown in Figure 6-3, where it is compared to the funnel for the system without uncertainty. Note that, because the Lyapunov function $V(t,\mathbf{x})$ does not depend on θ (i.e., $\Gamma = 0$), the Lyapunov function can be exactly evaluated at runtime.

6.3 Stochastic Verification

In a manner analogous to robust verification, stochastic verification can be applied to account for model uncertainty. Typically, this is useful for handling dynamic uncertainty, as it could provide less conservative results than the robust verification methods if the dynamic uncertainty can be better approximated as some colored noise with known mean. Following the methods outlined in [93, 82, 81], if one has a system driven by white noise, one can compute backwards reachable sets which guarantee a probability of successfully reaching the goal from a set of initial conditions.

To achieve this, I define a non-negative function $V(t,\mathbf{x})$, also known as a *supermartin-*

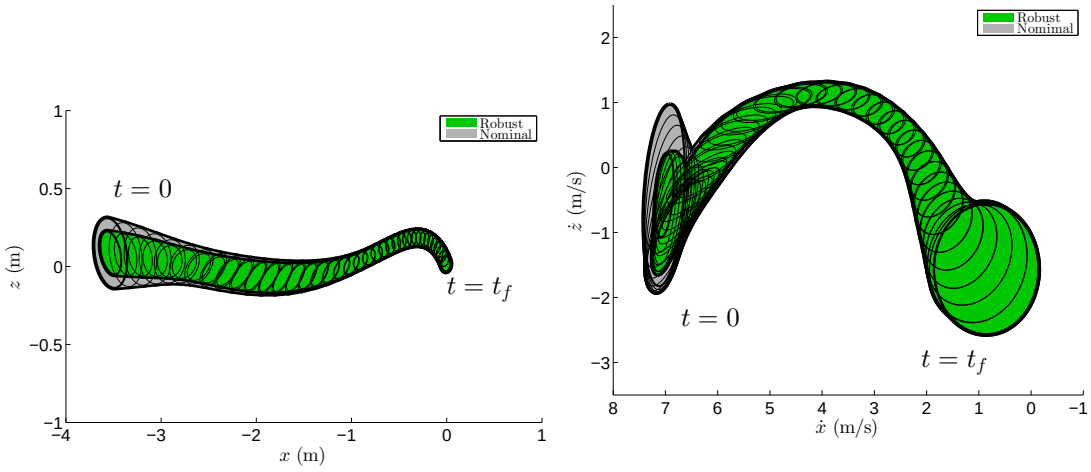


Figure 6-3: Slices of the high-dimensional funnel in the x-z and $\dot{x}$-$\dot{z}$ plane with (green) and without (gray) accounting for uncertainty. For the robust funnel, the uncertainty is modeled as a bounded input disturbance on the x-acceleration with a +/- 0.2 $\frac{m}{s^2}$ bound on the acceleration uncertainty.

gale , where $\mathbb{E}[\dot{V}(t,\mathbf{x})] \leq c$ and c is some non-negative constant. From Markov's inequality, it can be observed that the probability that $V(t,\mathbf{x}(t)) \geq 1$ is at most $V(0,\mathbf{x}(0)) + ct$.

Consider the stochastic differential equation $d\mathbf{x}(t) = \mathbf{f}(\mathbf{x})dt + \mathbf{g}(\mathbf{x})d\mathbf{w}(t)$. Using the Itô differential operator, I can show that

$$\mathbb{E}[\dot{V}(t,\mathbf{x})] = \frac{\partial V}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}) + \frac{\partial V}{\partial t} + \frac{1}{2}\text{trace}(\mathbf{g}(\mathbf{x})^T \frac{\partial^2 V}{\partial \mathbf{x}^2}\mathbf{g}(\mathbf{x})).$$

I can then set up the constraint

$$V(t,\mathbf{x}) < 1 \implies \mathbb{E}[\dot{V}(t,\mathbf{x})] < c.$$

As before, this constraint is easily enforced using sum-of-squares techniques. For this, I utilize a two step SOS optimization routine. The optimization is initialized by making the initial value for c very large. In the second step of the optimization routine, I seek to both minimize c and increase ρ. This optimization routine is summarized as follows:

STEP 1:

$$\begin{aligned} \underset{s_1,\gamma}{\text{maximize}} \quad & \gamma \\ \text{subject to} \quad & \gamma > 0, \\ & -\mathbb{E}[\dot{V}] + s_1(V-1) + c - \gamma \in \Sigma. \\ & s_1 \in \Sigma \end{aligned} \tag{6.6}$$

STEP 2:

$$\begin{aligned} \underset{V,c}{\text{maximize}} \quad & Vol(V) \\ \text{subject to} \quad & V \in \Sigma, \\ & c > 0, \\ & -\mathbb{E}[\dot{V}] + s_1(V-1) + c \in \Sigma. \end{aligned} \tag{6.7}$$

$$\tag{6.8}$$

The stochastic funnel with Gaussian uncertainty on the x acceleration was computed and plotted against the funnel for the nominal system. The results can be observed in Figure 6-4. These stochastic funnels can be interpreted as guaranteeing that, given initial condition $\mathbf{x}(0)$, the glider will land on the perch with a probability of $1-(V(\mathbf{x}(0))+cT)$, where T is the duration of the maneuver in seconds.

6.4 Design and Verification of Adaptive Controllers

A third approach for managing parametric uncertainty in control system design is through adaptation. While I have demonstrated that it is possible to generate robust funnels for various classes of uncertainty for the perching glider system, it is also possible that, if the uncertainty is parameteric (i.e. it does not vary with time), better performance could be achieved by estimating the uncertainty during the course of the perching maneuver and using those estimates to modify the controller online. In this section I explore developing

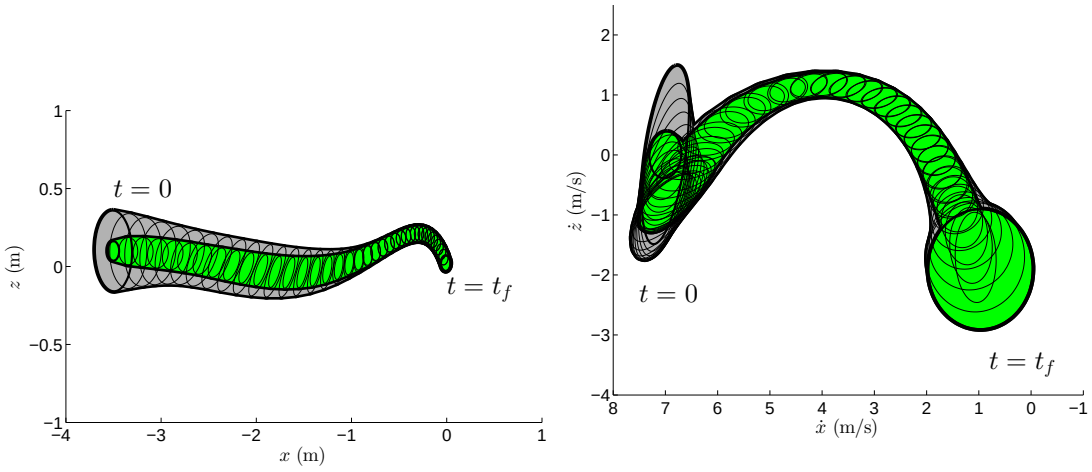


Figure 6-4: Slices of the high-dimensional stochastic funnel in the x-z and $\dot{x}$-$\dot{z}$ plane. For this funnel c=0.09 and there is Gaussian noise on the x-accelerations with a standard deviation of 0.1 $\frac{m}{s^2}$. Here, the stochastic funnel is shown in green and the nominal, non-robust funnel is shown in gray.

adaptive controllers for underactuated systems and demonstrate on several systems, including the perching glider system, that it is possible to increase the range of the system's tolerance to uncertainty when that uncertainty does not vary with time.

Unfortunately, there are relatively few methods for handling underactuated systems in adaptive control design. In [87], the authors addressed adaptive control of underactuated systems by applying partial feedback linearization to the collocated joint and carrying out region of attraction (ROA) analysis on the non-collocated joint. In [88], the authors applied backstepping to find a suitable Lyapunov function with which they can design an adaptive controller for their autonomous underwater vehicle. Similarly, in [31], the authors developed an adaptive controller for an underactuated quadrotor UAV capable of compensating for uncertain mass.

In this section, I follow the approach described in [45], which makes use of an adaptive control Lyapunov function (acLf) which can be found via backstepping. Using this Lyapunov function, the authors were then able to derive a nonlinear adaptive controller in a manner similar to that which is carried out in [92]. In this thesis, I replace the back-stepping approach by automating the search for the controller and Lyapunov functions using SOS optimization.

6.4.1 Local Adaptive Control About a Fixed-Point

Before considering the design of an adaptive controller for the time-varying glider system, I first consider the case of designing an adaptive controller which is only valid in a bounded set about an operating point. To do this, I find a *local* Lyapunov function for the adaptive system.

Consider the system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{F}(\mathbf{x})\theta + \mathbf{g}(\mathbf{x})\mathbf{u}. \tag{6.9}$$

The adaptive controller should have the form

$$\mathbf{u} = \alpha(\mathbf{x}, \hat{\theta}) \tag{6.10}$$

$$\dot{\hat{\theta}} = \tau(\mathbf{x}, \hat{\theta}). \tag{6.11}$$

Now consider the Lyapunov function

$$V = V_a(\mathbf{x}, \hat{\theta}) + \tilde{\theta}^T \Gamma \tilde{\theta} + \theta^T \Psi \theta, \tag{6.12}$$

where $\tilde{\theta} = \theta - \hat{\theta}$. In many cases, V_a can be chosen as $\mathbf{x}^T\mathbf{S}\mathbf{x}$, where $\mathbf{S}$ comes from solving the Riccati equation. Taking the derivative of the Lyapunov function yields

$$\dot{V} = \frac{\partial V_a(\mathbf{x}, \hat{\theta})}{\partial \mathbf{x}}\dot{\mathbf{x}} + \frac{\partial V_a(\mathbf{x}, \hat{\theta})}{\partial \hat{\theta}}\dot{\hat{\theta}} + \tilde{\theta}^T \Gamma \dot{\tilde{\theta}}. \tag{6.13}$$

If I choose the adaptive law such that

$$\dot{\hat{\theta}} = \tau(\mathbf{x}, \hat{\theta}) = \Gamma^{-1}\left(\frac{\partial V_a(\mathbf{x}, \hat{\theta})}{\partial \mathbf{x}}\mathbf{F}(\mathbf{x})\right)^T, \tag{6.14}$$

then

$$\dot{V} = \frac{\partial V_a(\mathbf{x},\hat{\theta})}{\partial \mathbf{x}}\dot{\mathbf{x}} + \frac{\partial V_a(\mathbf{x},\hat{\theta})}{\partial \hat{\theta}}\dot{\hat{\theta}} + (\theta - \hat{\theta})^T \Gamma(-\dot{\hat{\theta}}) \tag{6.15}$$

$$= \frac{\partial V_a}{\partial \mathbf{x}}(\mathbf{f}(\mathbf{x}) + \mathbf{F}(\mathbf{x})\theta) + \frac{\partial V_a}{\partial \hat{\theta}}\Gamma^{-1}\left(\frac{\partial V_a}{\partial \mathbf{x}}\mathbf{F}(\mathbf{x})\right)^T + (\theta - \hat{\theta})^T\left(-\frac{\partial V_a}{\partial \mathbf{x}}\mathbf{F}(\mathbf{x})\right)^T \tag{6.16}$$

$$= \frac{\partial V_a}{\partial \mathbf{x}}\left(\mathbf{f}(\mathbf{x}) + \mathbf{F}(x)\left(\hat{\theta} + \Gamma^{-1}\frac{\partial V_a}{\partial \hat{\theta}}^T\right) + \mathbf{g}(\mathbf{x})\alpha(\mathbf{x},\hat{\theta})\right). \tag{6.17}$$

Thus, I am left with the conditions

$$V = V_a(\mathbf{x},\hat{\theta}) + \tilde{\theta}^T\Gamma\tilde{\theta} + \theta^T\Psi\theta < \rho$$

$$\Longrightarrow$$

$$\dot{V} = \frac{\partial V_a}{\partial \mathbf{x}}\left(\mathbf{f}(\mathbf{x}) + \mathbf{F}(\mathbf{x})\left(\hat{\theta} + \Gamma^{-1}\frac{\partial V_a}{\partial \hat{\theta}}^T\right) + \mathbf{g}(\mathbf{x})\alpha(\mathbf{x},\hat{\theta})\right) < 0. \tag{6.18}$$

In the above constraints, many of the decision parameters (i.e., Γ) appear nonlinearly and constructing a series of bilinear alternations is non-trivial. However, if one is willing to let $\frac{\partial V_a}{\partial \hat{\theta}} = 0$, then constructing a series of bilinear alternations is possible and these constraints are easily verified with sum-of-squares techniques using a three step optimization routine to search for the multipliers, V and $\alpha(\mathbf{x},\hat{\theta})$. This optimization routine can be described as:

STEP 1:

$$\begin{aligned} \underset{s_1,\gamma}{\text{maximize}} \quad & \gamma \\ \text{subject to} \quad & \gamma > 0, \\ & -\dot{V} + s_1(V-\rho) - \gamma \in \Sigma. \\ & s_1 \in \Sigma \end{aligned} \tag{6.19}$$

STEP 2:

$$\begin{aligned} \underset{V,\Gamma,\Psi,\rho}{\text{maximize}} \quad & \rho \\ \text{subject to} \quad & \rho > 0, \\ & -\dot{V} + s_1(V-\rho) \in \Sigma. \end{aligned} \tag{6.20}$$

STEP 3:

$$\begin{aligned} \underset{\alpha,\rho}{\text{maximize}} \quad & \rho \\ \text{subject to} \quad & \rho > 0, \\ & -\dot{V} + s_1(V-\rho) \in \Sigma \end{aligned} \tag{6.21}$$

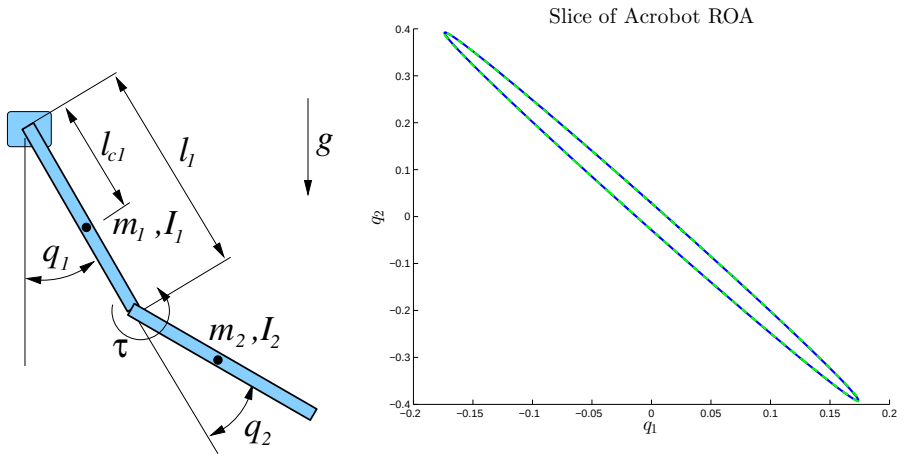
These steps then repeat until convergence is observed in ρ.

Adaptive Controller Results

Because the glider model does not have a fixed-point I can use to test the adaptive control design approach, I tested my method on a canonical underactuated control task, balancing the acrobot (see Figure 6-5) about the upright.

The equations of motion of the Acrobot have trigonometric terms in them; since these appear simply, it is possible to either perform an exact change of coordinates to a polynomial vectorfield, or to approximate the dynamics using Taylor expansion. In a manner consistent with the previous SOS approaches, I used a Taylor expansion to third order. For this problem, I parameterized the controller as $\alpha(\mathbf{x},\hat{\theta}) = \mathbf{K}_{LQR}\mathbf{x} + \hat{\theta}\mathbf{K}_{\theta}\mathbf{x}$, where $\mathbf{K}_{LQR}$ comes from the solution for the linear quadratic regulator for the nominal system when $\mathbf{q} = \mathbf{q}_0$. θ was specifically chosen to be the (scalar) unknown damping coefficient on the *noncollocated* (unactuated) shoulder joint.

To initialize the iterations, I chose $\mathbf{S}_a = \mathbf{S}$, where $\mathbf{S}$ comes from solving the Riccati equation. I also chose ρ to be very small and $\det(\Gamma)$ and $\det(\Psi)$ to be very large. This is a reasonable initialization procedure because $V = \mathbf{x}^T\mathbf{S}\mathbf{x}$ is a valid Lyapunov function with



(a) (b)

Figure 6-5: (a) The Acrobot system: The only actuation is at the elbow joint.(b) Slice of four dimensional Acrobot region of attraction in the q_1-q_2. Notice that there is little difference between the size of the ROA for the LQR controller (green) and the adaptive controller (blue).

some robustness to parametric uncertainty in a small neighborhood around the nominal point. The multipliers were chosen to be up to 4th order.

The results of searching for the controller can be observed in Figure 6-5, plotted as a slice of the 4 dimensional ellipsoidal level-set $V = \rho$. While non-negligible gains $\mathbf{K}_\theta$ were found by the optimization procedure, these gains had little affect on the size of the region of attraction. This is most likely due to the favorable robustness properties of LQR for a time-invariant system.

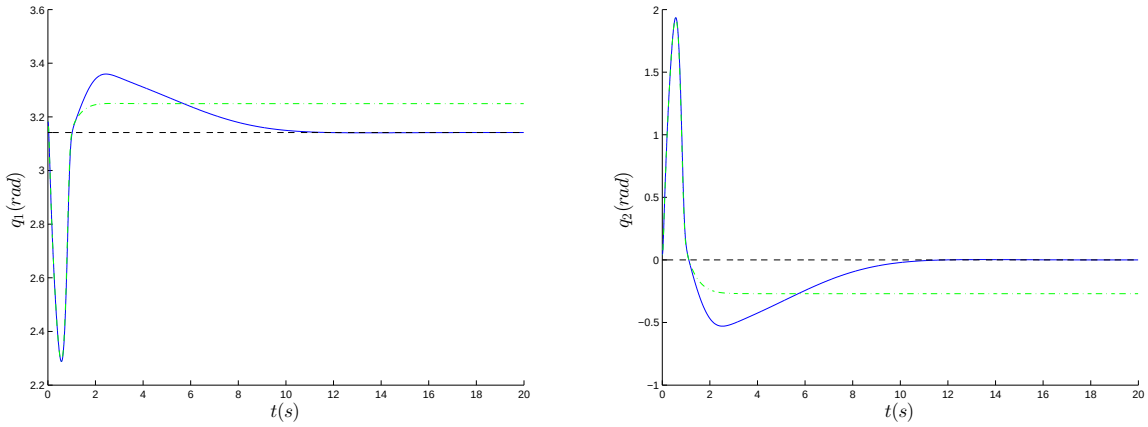
This being said, there are cases where unknown constant parameters can cause LQR to fail to achieve asymptotic stability. In these cases, we claim that our methods can design an adaptive controller which enables a region of attraction to exist. Consider the case of a constant, unknown offset appearing in the measurements provided to a LQR Controller. I write these closed loop dynamics as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})(\mathbf{K}(\mathbf{x} + \theta)). \tag{6.22}$$

Since $\mathbf{K}$ is a constant matrix, I can rewrite this as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\alpha(\mathbf{x}, \hat{\theta}) + \mathbf{g}(\mathbf{x})\mathbf{K}\theta, \tag{6.23}$$

which is identical to equation 6.9 when $\mathbf{F}(\mathbf{x}) = \mathbf{g}(\mathbf{x})\mathbf{K}$. To test my approach, I again implemented this case using the Acrobot. I applied a single measurement offset to the Acrobot's shoulder joint and designed the adaptive controller using SOS. The results can be seen in Figures 6-6, and 6-7. The ROA for the this adaptive controller is comparable to the ROAs shown in Figure 6-5 above while the ROA about the upright for the system using the LQR Controller does not exist.



(a) (b)

Figure 6-6: (a) A plot of q_1 with time. Notice how the LQR Controller (green) fails to converge to π, while the adaptive controller (blue) does.(b) A plot of q_2 with time. Notice how the adaptive controller (blue) converges to zero while the LQR Controller (green) does not.

6.4.2 Adaptive Control Along Trajectories

Without feedback linearization, designing adaptive controllers with provable stability properties becomes more complex. Given a desired trajectory, $\mathbf{x}_d(t)$, defined for $t \in [t_0, \infty]$, and unknown model parameters, a natural goal is to design an adaptive control law for the closed-loop system that converges asymptotically to the desired trajectory for all possible parameters in some family. However, with no guaranteed ability to directly cancel para-

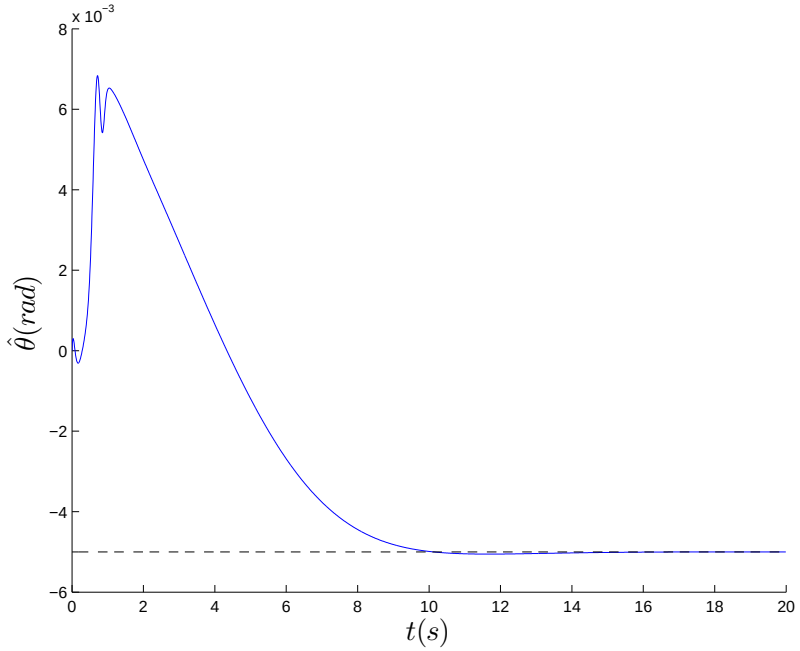


Figure 6-7: A plot of $\hat{\theta}$ with time. Even though it is not guaranteed to converge, the estimated parameter (blue) does converge to the true parameter (black).

metric uncertainty using feedback, for some systems it may be difficult or impossible to achieve asymptotic convergence to a desired trajectory. Furthermore, I would argue that for most real-world tasks this level of performance is not necessary.

Instead, I will use the notion of a funnel and only verify set invariance to design an adaptive controller for the perching glider. As described in Section 4.3, I use direct transcription to design the nominal trajectory. I then use TVLQR and the third step of the SOS optimization routine described above to generate the control law dependent on the parameter estimates.

Controller Parametrization

Because the control requirement is not exact trajectory following but staying in the funnel while moving between two regions of state space, I found it advantageous to allow for a $\hat{\theta}$-dependent shift of the nominal trajectory,

$$\mathbf{x}_d = \mathbf{x}_0 + \mathbf{w}(\hat{\theta}), \quad \mathbf{u}_d = \mathbf{u}_0 + \mathbf{v}(\hat{\theta}),$$

where $\mathbf{w}(\hat{\theta})$ and $\mathbf{v}(\hat{\theta})$ are a function that I can potentially design along with the controller.

Consider, as before, the system

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

If $\tilde{\mathbf{x}} = \mathbf{x} - \mathbf{x}_d$, $\tilde{\mathbf{u}} = \mathbf{u} - \mathbf{u}_d$, and $\tilde{\mathbf{u}} = \mathbf{K}\tilde{\mathbf{x}}$, then I can write

$$\begin{aligned}\dot{\tilde{\mathbf{x}}} &= \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})(\mathbf{K}\tilde{\mathbf{x}} + \mathbf{u}_d) - \dot{\mathbf{x}}_d \\ \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})(\mathbf{K}(\mathbf{x} - \mathbf{x}_0 - \mathbf{w}(\hat{\theta})) + \mathbf{u}_0 + \mathbf{v}(\hat{\theta})).\end{aligned}$$

Now, if $\bar{\mathbf{x}} = \mathbf{x} - \mathbf{x}_0$, then

$$\begin{aligned}\dot{\tilde{\mathbf{x}}} &= \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})(\mathbf{K}(\bar{\mathbf{x}} - \mathbf{w}(\hat{\theta})) + \mathbf{u}_0 + \mathbf{v}(\hat{\theta})) - \dot{\mathbf{x}}_0 \\ \dot{\bar{\mathbf{x}}} &= \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})(\mathbf{K}\bar{\mathbf{x}} + \mathbf{u}_0 - \mathbf{K}\mathbf{w}(\hat{\theta}) + \mathbf{v}(\hat{\theta})) - \dot{\mathbf{x}}_0.\end{aligned}$$

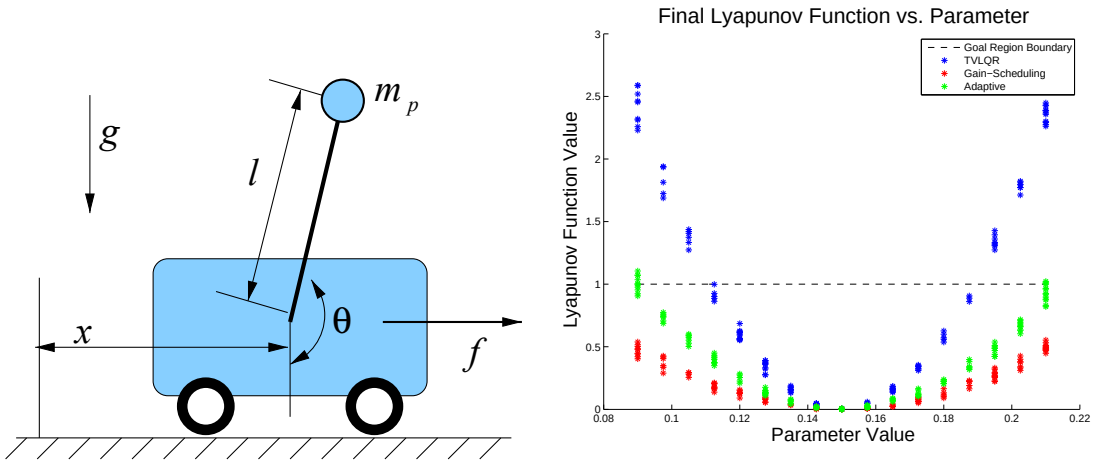
For simplicity, here I fix $\mathbf{w}(\hat{\theta}) = \gamma\hat{\theta}$ and $\mathbf{v}(\hat{\theta}) = \psi\hat{\theta}$ and let $\mathbf{K}_\theta = \mathbf{K}\gamma - \psi$. Then I have

$$\dot{\bar{\mathbf{x}}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})(\mathbf{K}\bar{\mathbf{x}} + \mathbf{u}_0 - \mathbf{K}_\theta\hat{\theta}) - \dot{\mathbf{x}}_0.$$

and thus achieve the controller parametrization $\mathbf{u} = \mathbf{K}\bar{\mathbf{x}} + \mathbf{u}_0 - \mathbf{K}_\theta\hat{\theta}$, where $\mathbf{K} = \mathbf{K}_{LQR}$, $\hat{\theta}$ comes from integrating $\dot{\hat{\theta}}$ derived above, and $\mathbf{K}_\theta$ is found using SOS optimization. This control law allows for adjusting the nominal trajectory online using $\hat{\theta}$ and thus provides the controller with a more reasonable $\mathbf{x}_d$ to follow.

Adaptive Controller Results

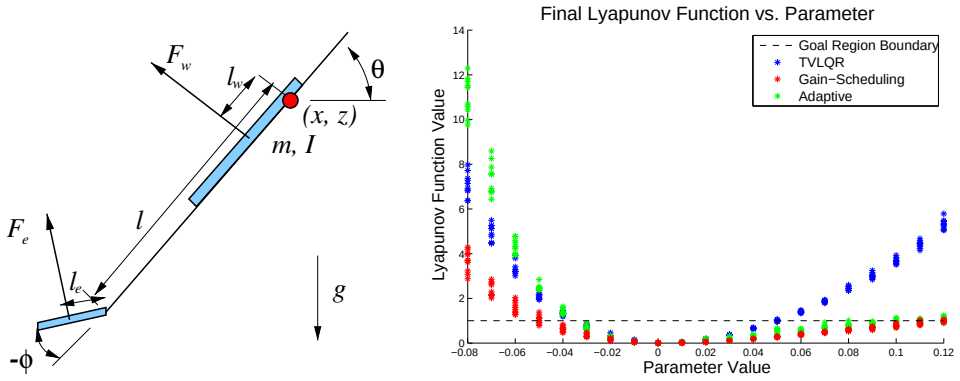
To test this adaptive control design algorithm, I first implemented it on a simple cart-pole system, where viscous friction at the non-collocated joint is unknown. I then tested the adaptive control design algorithm on the perching glider system and considered the case of an uncertain lift coefficient. This uncertain lift coefficient is created by adding an unknown constant offset to the C_L of the wing described in Section 3.1. Figures 6-8 and 6-9 demonstrate that the adaptive controllers provide a clear improvement in performance



(a) (b)

Figure 6-8: (a) The cart-pole system. Here, the uncertain parameter is at the pole's unactuated pivot. (b) Plot showing the cartpole system simulated from a range of initial trajectories and parameter values. Both the gain-scheduling controller (red) and the adaptive controller (green) provide an advantage over the TVLQR design (blue).

by increasing the given system's tolerance to deviations in its uncertain parameter. In both cases, even though the estimated parameter did not converge to the true parameter, the parameter estimates generally converged to a constant value and evolved in such a way that the overall performance of the system improved. However, lack of parameter convergence is not surprising, since parameter convergence requires persistent excitation, and this metric was not considered when designing the nominal trajectory for either system.



(a) (b)

Figure 6-9: (a) The perching glider system. Here, the uncertain parameter is the lift coefficient of the aircraft.(b) Plot showing the perching glider system simulated from a range of initial trajectories and parameter values. Both the gain-scheduling controller (red) and the adaptive controller (green) provide an advantage over the TVLQR design (blue).

Chapter 7

Robust LQR-Tree Controller

Because the the acceleration uncertainty is dynamic in nature, using the adaptive method presented in the preceding chapter is challenging, since it only accounts for parametric uncertainty. While both the robust and stochastic verification approaches have the ability to handle dynamic uncertainty, the robust methods also possess the ability to handle parametric uncertainty. Because of its ability to handle both types of uncertainty, I chose to proceed with the robust verification approach. For the initial study, I added bounded uncertainty to the system's x-accelerations.

7.1 Verification Experiments

To test the robust funnels, I carried out a number of flights, varying the initial speed between 6.5 m/s and 7.5 m/s. I executed a TVLQR controller designed for a single trajectory with a nominal speed of 7.0 m/s. As can be seen in Figure 7-2, 80 percent of the trajectories that originated in the funnel, stayed in the funnel. And while a few of the trajectories did leave the funnel, this is to be expected, since accounting for all sources of uncertainty is not possible. Nevertheless, these results indicate that the robust verification can in fact provide reasonably accurate guarantees of system performance when robustness is added.

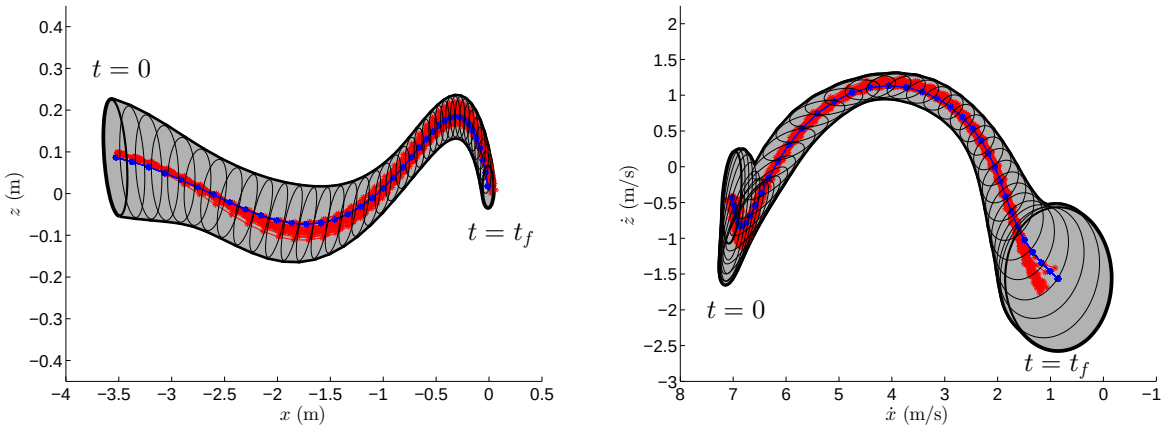
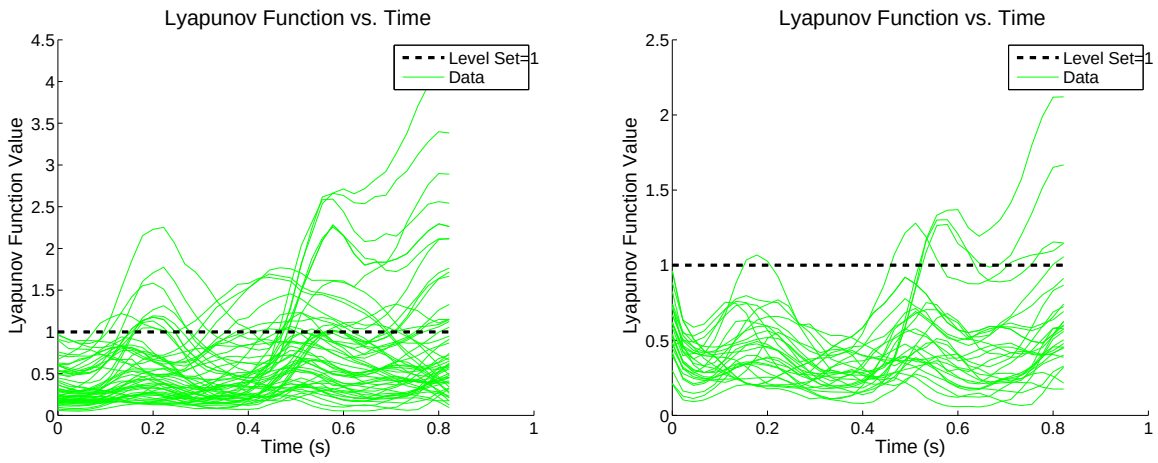


Figure 7-1: These figures show the robust funnel slices in the x-z plane and the $\dot{x}$-$\dot{z}$ plane with the experimental trajectories superimposed on top of them. Note: These plots only serve as a visual aide. Because these are two-dimensional slices, one can not tell whether or not the trajectory is truly inside of the funnel from these plots.

7.2 LQR-Tree Results

Following the verification experiments, I proceeded to generate an LQR-Tree using the robust funnels. I then carried out a number of experiments in a Vicon motion capture studio, varying the initial speed of the perching maneuver between 6 and 8 m/s. Figure 7-3 shows the results of this experiment when only an open loop trajectory is used to control the aircraft. When the LQR-Tree algorithm is applied, astounding perching performance is obtained (see Figure 7-4). Out of the approximately 150 launches, 94 percent landed in the 6.5 centimeter capture region. Figure 7-5 compares the final perching positions of open loop control, time-varying linear control for a single trajectory, and the LQR-Tree approach. The LQR-Tree demonstrates a clear improvement over a single stabilized trajectory for the higher initial launch speeds.

To further demonstrate the performance advantages of the LQR-Tree approach, I launched the glider by hand. These hand thrown trajectories have a much larger variation in initial pitch than those launched from a crossbow and therefore benefit from an LQR-Tree which has funnels to accommodate these initial conditions. Figure 7-6 shows a video stills from three sequential handthrown perching experiments, all of which have significantly different initial pitch conditions. The results of these experiments can be seen in Figure 7-7. The



(a) (b)

Figure 7-2: (a) Experimental trajectory verification for single stabilized trajectory using the nominal funnel. (b) Experimental trajectory verification for single stabilized trajectory using a robust funnel. For the nominal funnel, approximately 50 percent of the trajectories that start in the funnel stay in the funnel for all time, while 80 percent of the trajectories stay in the robust funnel for all time.

figure shows that when additional funnels are added to the LQR-Tree to capture the high initial pitch conditions associated with throwing the glider by hand, the perching performance noticeably improves.

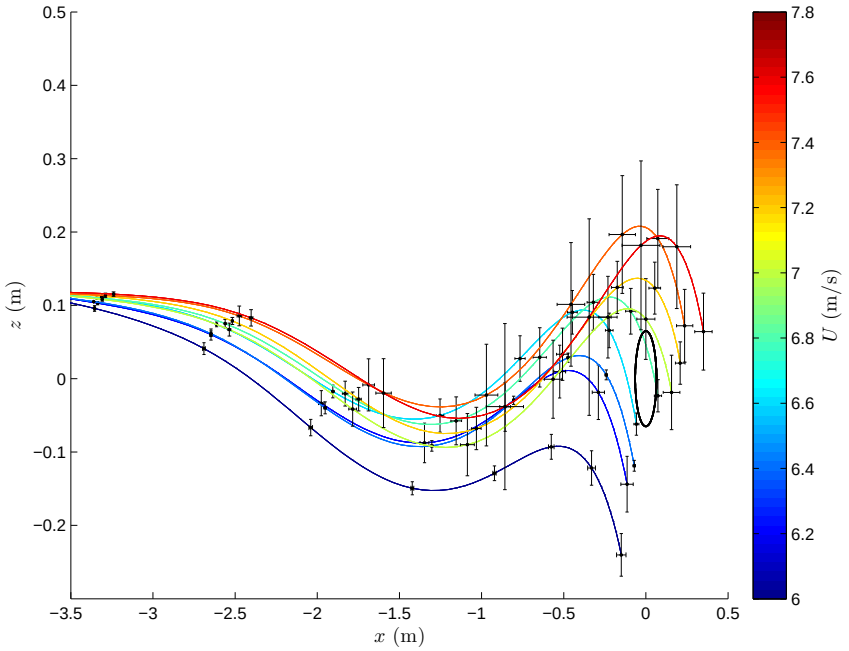


Figure 7-3: Averaged x-z trajectories from open loop perching experiments. Each curve represents an average of multiple trajectories with similar initial conditions; error bars are included to show the spread of the trajectories. The colors, from cold to hot, represent the variation in the initial speed of the aircraft. Only 31 percent of the trajectories, terminate in the 6.5 cm goal region.

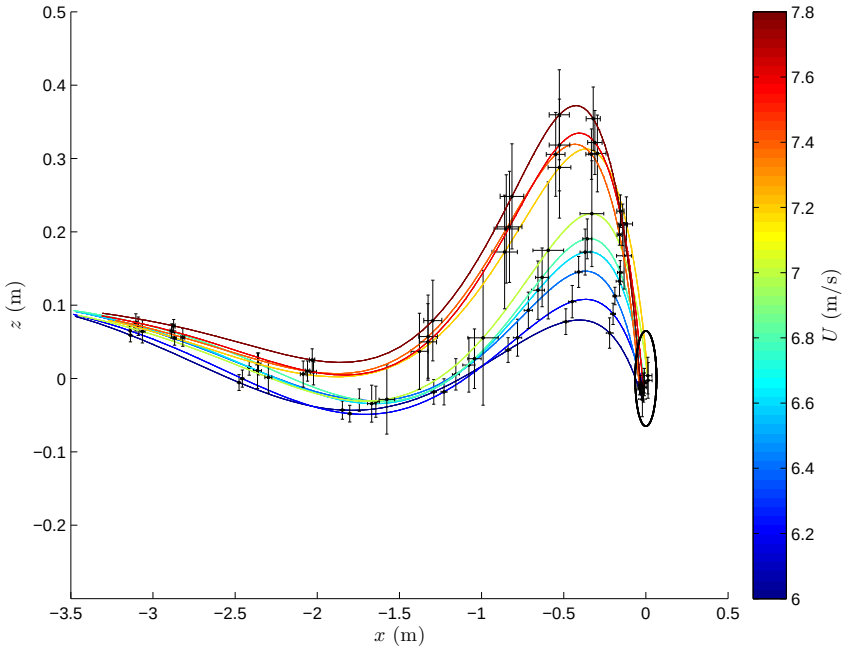


Figure 7-4: Averaged x-z trajectories from LQR-Tree perching experiments. Each curve represents an average of multiple trajectories with similar initial conditions; error bars are included to show the spread of the trajectories. The colors, from cold to hot, represent the variation in the initial speed of the aircraft. 94 percent of the trajectories, despite their various initial speeds, terminate in the 6.5 cm goal region

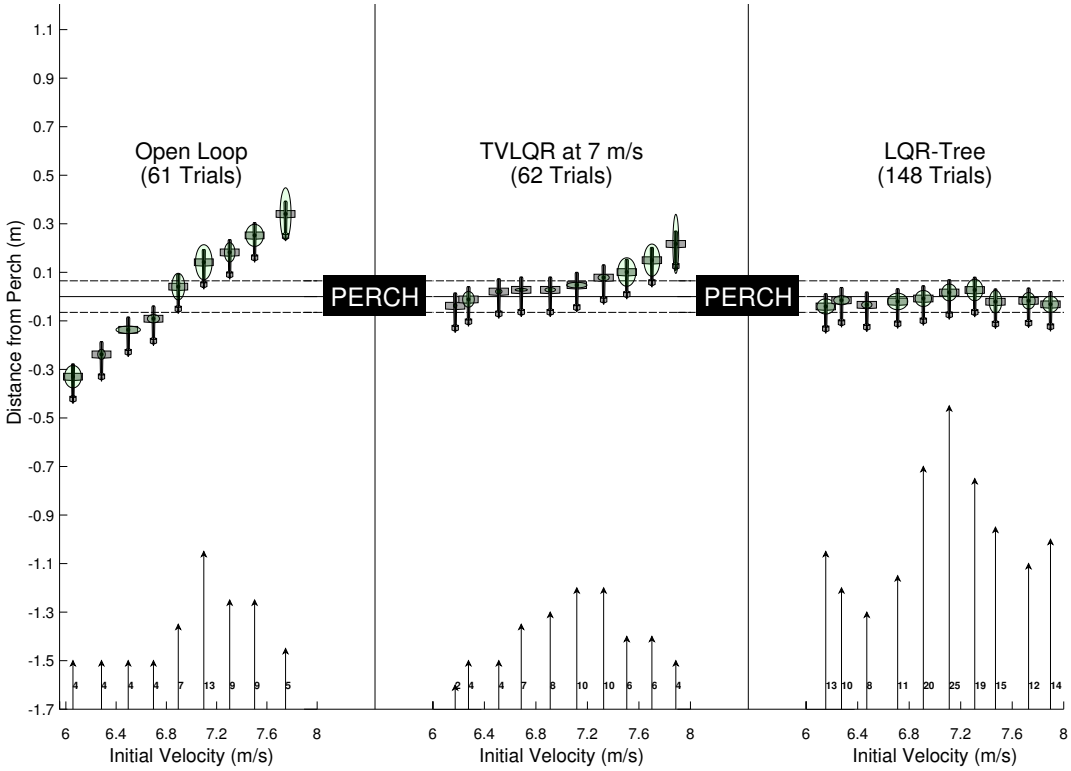
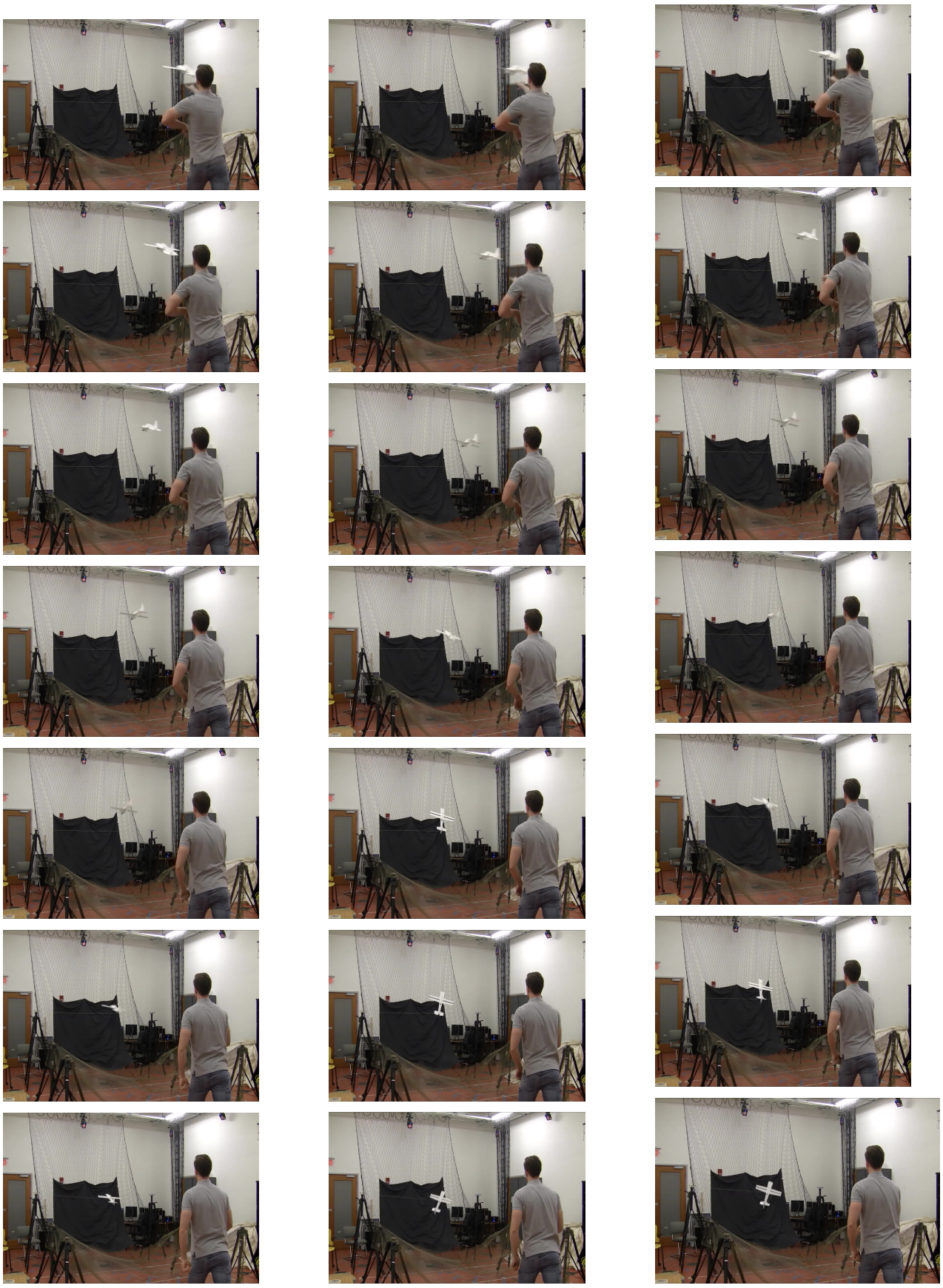
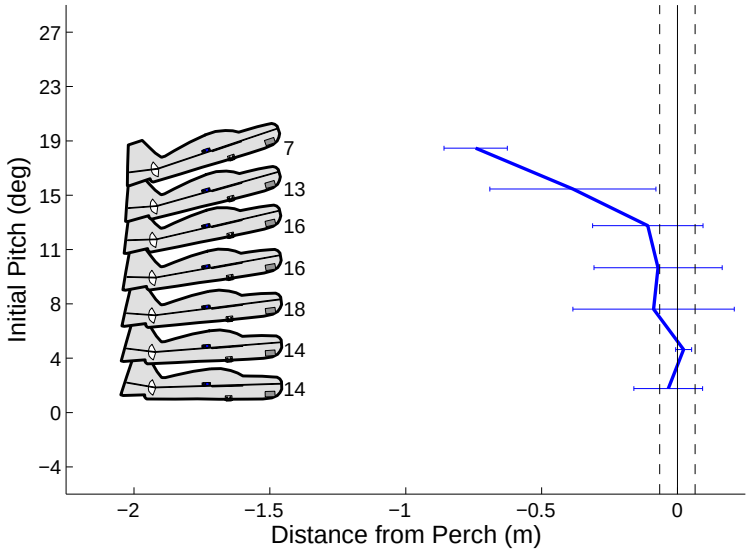


Figure 7-5: Plot shows perching performance of LQR-Tree controller versus the open-loop controller and a single TVLQR Controller at 7 m/s. Each plane represents a set of trials grouped by initial speed. The ellipses represent the variation in position error (x-axis) and the variation in the initial speed (y-axis). The arrows represent the flight direction as well as the number of flights in each bin. 94 percent of the trials using the LQR-Tree controller landed in the 0.065 cm target range.

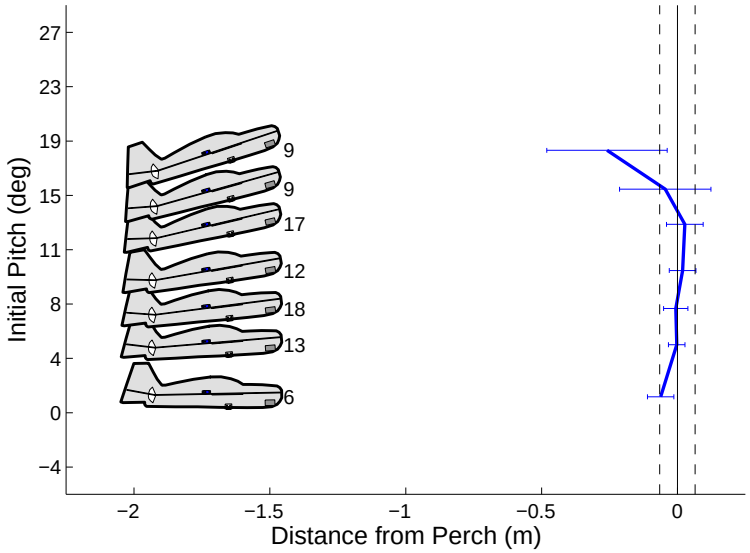


(a) (b) (c)

Figure 7-6: These movie stills demonstrate three sequential handthrown perching trials. Despite the noticeable initial pitch variation, the LQR-Tree is still able to guide the aircraft to the perch.



(a)



(b)

Figure 7-7: Handthrown perching results using LQR-Tree without (a) and with (b) high-pitch funnels. The glider icons which depict the variation in initial pitch are shown on the left and the final position of the aircraft with respect to the perch is shown on the right. The plot points and error bars show the mean and variance in the final positions respectively.

Chapter 8

Powerline Perching and Outdoor Flight

The previous chapters demonstrated that it was possible to achieve robust perching indoors using a Vicon motion capture studio. However, no effort was made to address the impact of significant sensor noise or external disturbances on the control methods presented. To do this, I tested the control approach outlined in this thesis using onboard sensors and in an outdoor environment.

In recent years, there has been significant interest in perching UAVs on powerlines to recharge and conduct perch and stare surveillance [17]. Not only is it possible to harvest energy from the powerlines, but the magnetic field generated by the powerline provides a unique opportunity to localize the aircraft during a perching maneuver. Using this potential real-world perching scenario as a sensing framework, I constructed an experimental powerline and onboard sensing unit to estimate the position of the aircraft relative to the powerline and explore the impact of wind gusts on the perching aircraft.

8.1 Experimental Set-up

Here, I describe the experimental powerline used in the outdoor perching experiments. To construct this powerline, I used a 4 gauge welding wire looped three times to carry an effective peak current near 100 amps. This current level was selected to represent the current flowing through a 10 kV distribution line driving a 1 MW load. To generate the AC current, I used a PWM motor amplifier fed by a 600 Watt DC power supply and driven by

a conventional signal generator to adjust the frequency of the current. This feature allowed for running the powerline at 80 Hz instead of 60 Hz to avoid the 60 Hz noise ubiquitous in indoor environments. To create the dipole configuration, I built a wooden stand to support a rectangular loop of wire 2.4 meters long by 0.3 meters wide. Furthermore, the wire leads bringing current to and from the power electronics were arranged in a twisted pair configuration so as to ensure that the only magnetic field being generated by the wire would be due to the loop. Figure 8-1 shows a block diagram of the experimental powerline set up.

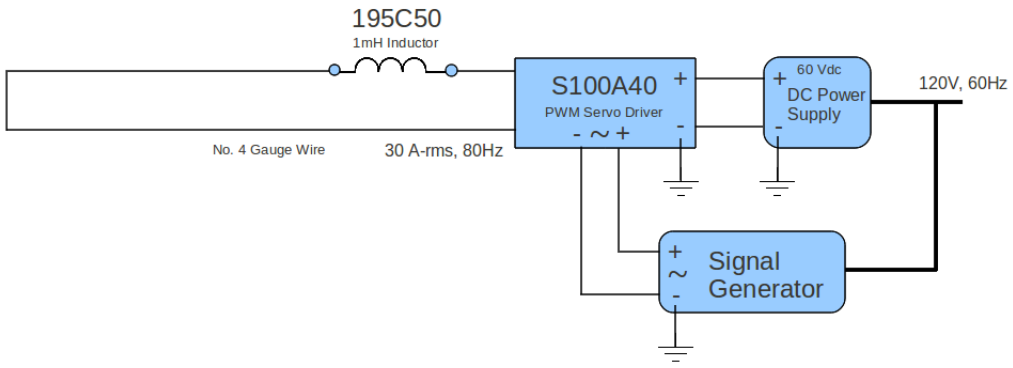


Figure 8-1: Powerline block diagram

8.2 Onboard System Architecture

After constructing the experimental powerline system, I developed specialized hardware for real-time sensing, estimation, and control which could be carried onboard the aircraft. The overall system architecture can be seen in Figure 8-2.

8.2.1 Hardware Design

The $\frac{1}{r^3}$ drop off in magnetic field strength and the sub-second duration of the perching maneuver place strict design requirements on the hardware. For this reason, I developed hardware capable of sensing low-level magnetic fields and providing reliable state estimation at high update rates. The GOSHAWK magnetic sensing system, which can be seen in Figures 8-3 and 8-4, emerged as a low-cost, light-weight solution. The GOSHAWK magnetic sensing system consists of a Honeywell magnetometer (HMC2003), a 24-bit high

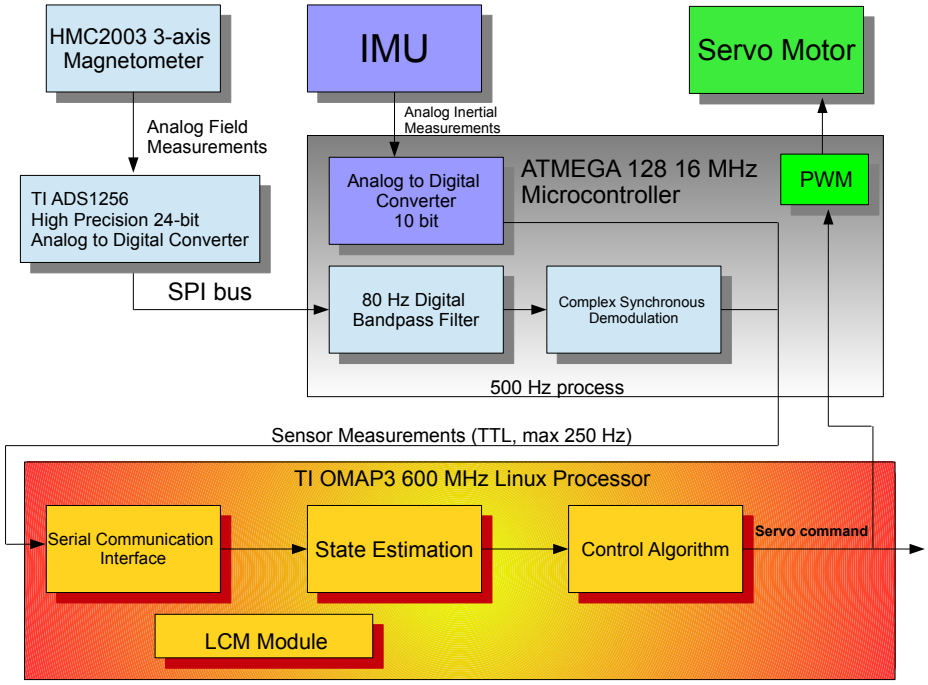


Figure 8-2: System Block Diagram

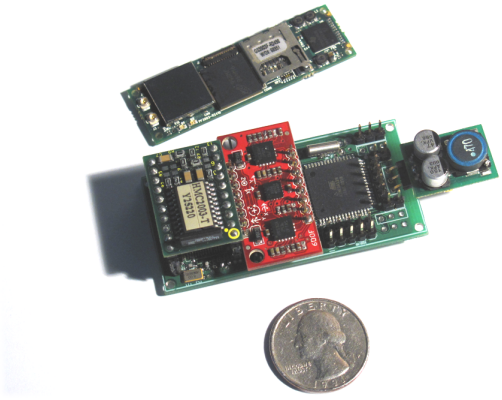


Figure 8-3: Top view of GOSHAWK sensing board.

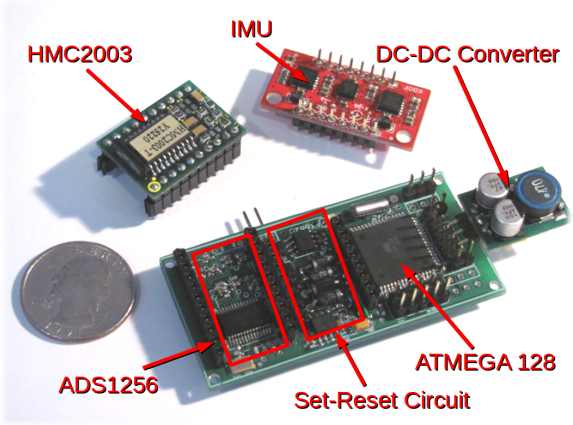


Figure 8-4: Exploded-view of GOSHAWK sensing board.

fidelity analog-to-digital converter (ADS1256), an ATMEGA128 microcontroller for low-level signal processing, a Gumstix Overo motherboard for high level computation, and an IMU for inertial measurement. Together, the sensing board and Gumstix Overo motherboard weigh less than 35 grams and are able to measure magnetic field signals down to 100 μG with reasonable precision. Overall, the entire aircraft weighs around 115 grams.

8.3 Magnetic Field Modeling

Once the sensing hardware was developed, I turned my attention to generating position estimates from the magnetic field measurements. To do this, it was necessary to build a mathematical model of the magnetic field.

As mentioned in the previous section, a rectangular current loop was chosen as the powerline configuration of interest. Here, I first describe my approach for modeling the current loop's static magnetic field, i.e., the field that would arise if the current were held constant, as a function of position. Following this I then describe the slight modifications made to the model to represent fields generated by alternating current sources.

8.3.1 Static Magnetic Field Model

In [71], a static model was derived for the rectangular current loop configuration using Ampere's Law by approximating the loop as two parallel wires (see Figure 8-5). However, this model proved to be inadequate, since it did not take into consideration the full

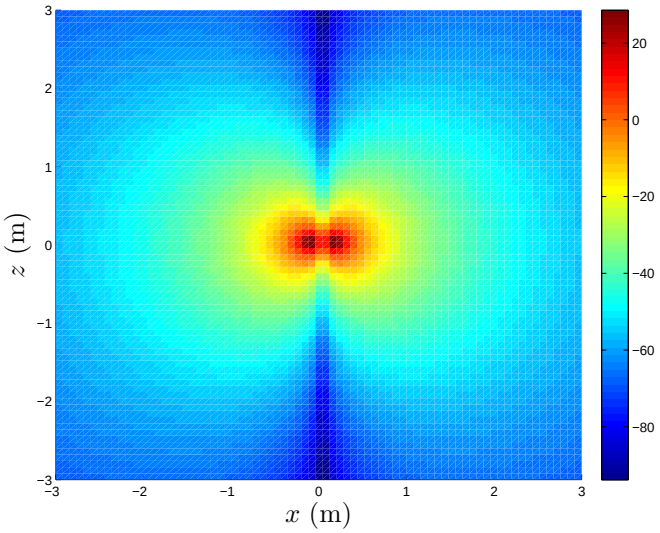


Figure 8-5: Plot of the magnetic field magnitude generated by two infinite current carrying wires.

rectangular loop configuration.

To model the rectangular current loop more accurately, I represented the loop as a summation of infinitesimal current elements and used the Biot-Savart law to compute the total magnetic field.

Following the conventions displayed in Figure 8-6, the Biot-Savart Law can be formulated as,

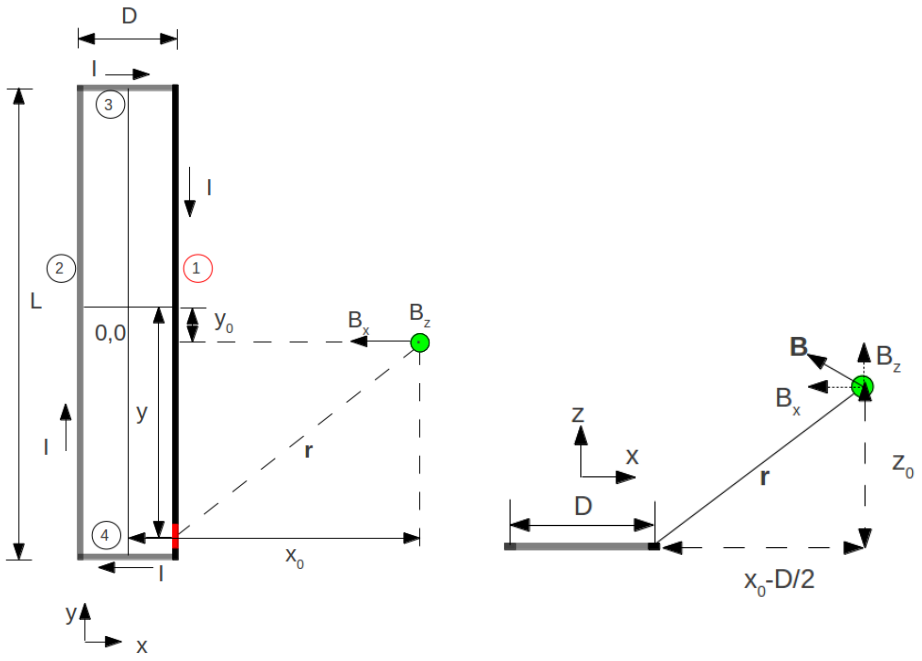
$$\mathbf{B} = \int \frac{\mu_0 I d\mathbf{l} \times \mathbf{r}}{4\pi r^3}, \tag{8.1}$$

where μ_0 is the magnetic permeability of free space, I is the current in the wire, D is the distance between the long parallel wires, L is the distance between the end wires, $d\mathbf{l}$ is the infinitesimal current element, and r is the distance from the infinitesimal current element to the aircraft location. In the case of the first wire, as shown in Figure 8-6, these terms can be rewritten as,

$$\mathbf{l} = -y\hat{\mathbf{j}} \tag{8.2}$$

$$d\mathbf{l} = -dy\hat{\mathbf{j}} \tag{8.3}$$

$$\mathbf{r} = (x_0 - \tfrac{D}{2})\hat{\mathbf{i}} + (y_0 - y)\hat{\mathbf{j}} + z_0\hat{\mathbf{k}}, \tag{8.4}$$



(a) (b)

Figure 8-6: (a) Top view of rectangular current loop. (b) Side view of rectangular current loop. In both images, the location of the magnetic field sensor is represented by the green circle.

where x_0, y_0, and z_0 are the position of the aircraft, and $\hat{\mathbf{i}}$, $\hat{\mathbf{j}}$, and $\hat{\mathbf{k}}$ represent the Cartesian unit vectors. Taking the cross product between $d\mathbf{l}$ and $\mathbf{r}$ yields,

$$d\mathbf{l} \times \mathbf{r} = (-z_0\hat{\mathbf{i}} + (x_0 - \tfrac{D}{2})\hat{\mathbf{k}})dy. \tag{8.5}$$

Substituting into the Biot-Savart Integral yields,

$$\mathbf{B} = \int_{-\frac{L}{2}}^{\frac{L}{2}} \frac{\mu_0 I(-z_0\hat{\mathbf{i}} + (x_0 - \frac{D}{2})\hat{\mathbf{k}})dy}{4\pi(\sqrt{(x_0 - \frac{D}{2})^2 + (y_0 - y)^2 + z_0^2})^3}, \tag{8.6}$$

which is a vector containing the x and z components of the magnetic field, $B_{x,1}$ and $B_{x,2}$, as a function of position from the first wire. In the same way, three other integrals can be evaluated to obtain $B_{y,3}, B_{y,4}, B_{z,1}, B_{z,2}, B_{z,3}$ and $B_{z,4}$, which are the vector field contributions

from the three remaining wires. The full *static* magnetic field equations then become

$$B_x = B_{x,1} + B_{x,2} \tag{8.7}$$

$$B_y = B_{y,3} + B_{y,4} \tag{8.8}$$

$$B_z = B_{z,1} + B_{z,2} + B_{z,3} + B_{z,4}, \tag{8.9}$$

where $\mathbf{B} = \begin{bmatrix} B_x & B_y & B_z \end{bmatrix}^T$. To simplify the model, I only considered the two-dimensional plane where $y = 0$. (Note that this model and its derivations could be easily re-generalized for a three-dimensional system if needed.) The model then becomes $\mathbf{B} = \begin{bmatrix} B_x & B_z \end{bmatrix}^T$.

8.3.2 Time-varying Magnetic Field Model

Although the model derived in the previous section captures the three dimensional effects of the current loop, it still does not capture the time-varying (60 Hz) nature of the field. To do this, I modeled the spatial magnetic field variations as being modulated by a 60 Hz sine wave. I defined the time-varying magnetic field $\mathbf{B}_m = \mathbf{B}\cos(\omega_c t + \phi)$, where ω_c is the 60 Hz carrier frequency and ϕ is the phase of the current in the wire. (It is important to note that even though I considered time-varying magnetic fields, I only explored ELF fields, which can still be modeled using *quasi-static* approximations, i.e., $\nabla \times \mathbf{E} \approx 0$).

8.4 Signal Processing

Even though this time-varying magnetic field model could be used directly, there are some significant difficulties with including this modulated magnetic field in the estimator's measurement model. First of all, the 60 Hz carrier frequency is high enough to potentially be a problem for real-time implementation. In fact, just to avoid aliasing, the estimator would have to run at 120 Hz. Such a sparse sampling rate (2 samples per cycle) makes tracking a nonlinear sinusoidal signal challenging, especially if an EKF were used. Ideally, for a 60 Hz signal, sampling frequencies of 600 Hz or more would be desirable if one wanted to preserve reasonable linearizations and good noise rejection. The second major problem with this approach is that phase, ϕ, is unknown.

To address the first challenge, I eliminated the carrier frequency altogether though signal processing. By separating the high frequency 60 Hz components from the aircraft dynamics, this approach allowed for executing the estimator at a much more reasonable update rate.

Complex Synchronous Demodulation

It is well known that any sinusoidal signal with a given phase can be decomposed into both a cosine component and sine component [75]. Furthermore, the modulated message signal, y_m, is equivalent to,

$$y_m(t) = m(t)\cos(\omega_c t + \phi), \tag{8.10}$$

where ω_c is the carrier frequency in rad/s, t is time, m is the message signal, and ϕ is the signal phase. When y_m is multiplied by $\sin\omega_c t$ and $\cos\omega_c t$, the result yields

$$y_{m,c}(t) = m(t)\frac{\cos(2\omega_c t + \phi) + \cos(\phi)}{2} \tag{8.11}$$

$$y_{m,s}(t) = m(t)\frac{\sin(2\omega_c t + \phi) + \sin(\phi)}{2}. \tag{8.12}$$

After these signals are low-pass filtered, the results are

$$y_{m,c}(t) = m(t)\frac{\cos(\phi)}{2} \tag{8.13}$$

$$y_{m,s}(t) = m(t)\frac{\sin(\phi)}{2}. \tag{8.14}$$

Notice that these two signal are 90 degrees out of phase with one another and together they represent the real and complex components of the message signal.

Demodulated Field Model

The new magnetic field model can now be constructed as follows:

$$B_{x,c} = B_x \cos(\phi) \tag{8.15}$$

$$B_{x,s} = B_x \sin(\phi) \tag{8.16}$$

$$B_{z,c} = B_z \cos(\phi) \tag{8.17}$$

$$B_{z,s} = B_z \sin(\phi) \tag{8.18}$$

$$\phi = mt + b, \tag{8.19}$$

where ϕ is the phase, m and b are unknown constants, and t is time. This could also be written in complex form as

$$B_{x,dm} = B_{x,c} + B_{x,s}i \tag{8.20}$$

$$B_{z,dm} = B_{z,c} + B_{z,s}i. \tag{8.21}$$

In this case, I can define the new field model as $\mathbf{B}_{dm} = \begin{bmatrix} B_{x,c} & B_{x,s} & B_{z,c} & B_{z,s} \end{bmatrix}^T$.

Although this method allows for a magnetic field model with much lower frequency content, it is still does not eliminate dependence on ϕ completely. Because of this, one might be tempted to ignore phase completely and use only the magnitudes of $B_{x,dm}$ and $B_{z,dm}$ as a magnetic field model. However, as is discussed in the next section, this approach has its own shortcomings.

8.5 Measurement Ambiguity

As mentioned in the previous section, knowledge of the powerline's phase does not exist onboard the aircraft. This deficiency negatively impacts the performance of the modulated and demodulated magnetic field models which depend explicitly on phase. As mentioned in the previous section, it is tempting to ignore phase altogether, and use the field model $\mathbf{B}_{dm} = \begin{bmatrix} |B_{x,dm}| & |B_{z,dm}| \end{bmatrix}^T$. However, when combined with the symmetry of the magnetic field,

this model leads to eight positions where the aircraft will obtain identical measurements (see Figure 8-7). If by some additional signal processing the relative sign of the x and z magnetometer signals is known, the number of ambiguous measurements is reduced to four. In contrast, when phase information is included, only two ambiguous measurements exist, and they are on opposite sides of the powerline. To reconstruct this phase information onboard the aircraft, reduce the measurement ambiguity, and provide the full system state required by the controller, I employed state estimation.

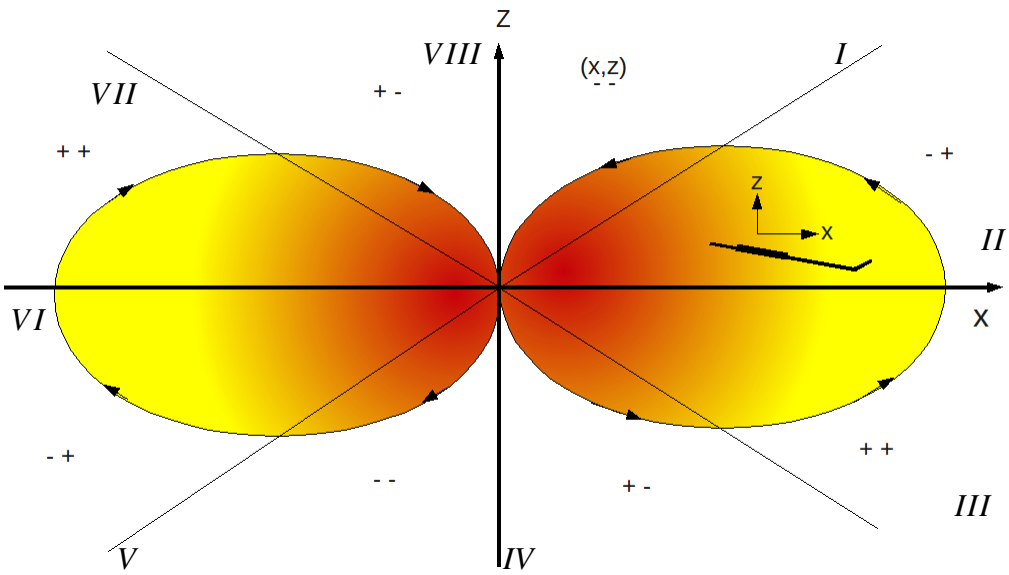


Figure 8-7: This figure graphically represents the phase-amplitude ambiguity which arises due to the field's inherent symmetries and its AC nature. Each octant represents a region of potential ambiguity.

8.6 State Estimation

Typically a state estimator, or observer, combines a dynamic model (also known as a *process model*) and a measurement model to reconstruct a system's state. Because many control methodologies rely on full-state feedback and perform best with minimal measurement noise, using an estimator is essential for successful powerline perching. Moreover, because the constant phase offset of the powerline is *observable* but can not be measured onboard the aircraft, the state estimator can be used to reconstruct the phase offset and resolve the measurement ambiguity.

During the course of this investigation, several state estimation methods were explored, including particle filters, Unscented Kalman Filters, and Extended Kalman Filters. All three state estimation algorithms used some variation of the model of the magnetic field as seen by the sensor as the measurement model. For the process model, both an aircraft model based on flat-plate theory and a double integrator were used. In the end, it was determined that the Extended Kalman Filter with an augmented state for phase tracking provided the best performance given the required computational cost.

8.6.1 Extended Kalman Filter

The discrete-time Extended Kalman Filter (EKF) is a nonlinear form of the classical discrete-time Kalman filter, which computes the observer gains by linearizing about the current state estimate. The discrete-time EKF algorithm is as follows:

To start, the state at the next time step is computed using a nonlinear *process model*,

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{f}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k), \tag{8.22}$$

where $\hat{\mathbf{x}}$ is the system state, k is the time-sample, $\mathbf{f}$ represents a nonlinear function, and $\mathbf{u}_k$ is the system input. Then, the error covariance matrix as well as the Kalman gains are determined in the following manner: First, the *a priori* error covariance matrix is computed as,

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_k \mathbf{F}_k^T + \mathbf{Q}_k, \tag{8.23}$$

where $\mathbf{Q}_k$ is the process covariance matrix and $\mathbf{F}_k$ represents the process gradients. Next, the observer gain matrix $\mathbf{K}_k$ is computed as,

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T [\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k]^{-1}, \tag{8.24}$$

where $\mathbf{R}_k$ is the measurement covariance and $\mathbf{H}_k$ represents the measurement gradients. Following this, the *a posteriori* error covariance matrix is computed as,

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1}. \tag{8.25}$$

Finally, the state estimate is updated by multiplying the Kalman gain matrix by the measurement error as follows:

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{y}_k - \mathbf{y}(\hat{\mathbf{x}}_{k|k-1})). \tag{8.26}$$

Here, $\mathbf{y}$ is the output of the *measurement model* and $\mathbf{y}_k$ is the actual measurement at timestep k.

8.6.2 Process Model

The estimator's process model is primarily what enables the EKF to smooth the state estimates. In most cases, it is a dynamical model which maps control inputs to state outputs. In this work, I used a double integrator process model of the form $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$ with an augmented state space to track the phase of the powerline's current. In keeping with the planarization of the magnetic field model, I also restricted the process model to two-dimensions.

The double integrator model can be described as a model with the form $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{u})$, where the state space is $\mathbf{x} = \begin{bmatrix} x & z & \theta & \dot{x} & \dot{z} \end{bmatrix}^T$ and the inputs, $\mathbf{u} = \begin{bmatrix} \ddot{x} & \ddot{z} & \theta \end{bmatrix}^T$, are noisy accelerometer and ptich measurements.

To track the powerline current's phase, I augmented the double integrator model with the unknown constant phase parameters m and b. The state then became $\mathbf{x} = \begin{bmatrix} x & z & \theta & \dot{x} & \dot{z} & m & b \end{bmatrix}^T$, where $\dot{m} = 0$ and $\dot{b} = 0$.

8.6.3 Measurement Model

The measurement model of an estimator maps the system states to all the available sensor outputs. This means that not only must the magnetic field model itself be considered, but also how the individual sensors (such as a magnetometer) view those magnetic field measurements must also be considered. I defined the static magnetic field in the frame of

the sensor as

$$\mathbf{B}_r = \mathbf{R}_\theta \mathbf{B}, \tag{8.27}$$

where $\mathbf{R}_\theta$ is the rotation matrix representing a change in pitch.

I then redefined $\mathbf{B}_m$ and $\mathbf{B}_{dm}$, letting $\mathbf{B} = \mathbf{B}_r$. In this work, I considered one measurement model, namely

$$\mathbf{y} = \begin{bmatrix} B_{x,c} & B_{x,s} & B_{z,c} & B_{z,s} & \theta & \dot{\theta} \end{bmatrix}^T. \tag{8.28}$$

8.7 Experimental Results

Using the newly constructed experimental powerline, I tested my approach using the hardware system described above in 8.2.1. With the estimator and the controller running on board the glider at 340 Hz, I demonstrated successful position estimation using the magnetic field-based state estimator. Furthermore, I demonstrated that these estimates were good enough to enable successful closed-loop perching for the fixed-wing UAV.

8.7.1 Offboard Estimation

Before testing the performance of the estimator during a perching maneuver, I first tested the estimator by taking a slow sweep in the space around the power line, using the on-board sensing electronics. After collecting the sensor data, I ran the estimator with phase tracking offline, using the double integrator with accelerometer inputs for the process model and the complex magnetometer signals as the measurement model. Because the purpose of this experiment was to test the impact of the magnetic field measurements on estimator performance, I used pitch information from Vicon motion capture instead of from the inertial measurement unit. Moreover, because I was in an indoor environment with ample possibility for magnetic field distortion, I added a simple function $\mathbf{B}_e(x,z) = \Theta\Phi$, where $\Phi = [xz, x, z, 1]^T$, to the magnetic field model $\mathbf{B}$ and fit its parameters using motion capture data to try to model these field distortions.

The results of the experiments described above are shown in Figure 8-8. It can be observed that reasonable position estimates were obtained up to about 4 meters from the wire. As would be expected, because of the $\frac{1}{r^3}$ dependence of the field magnitude, the signal to noise ratio is much better closer to the wire.

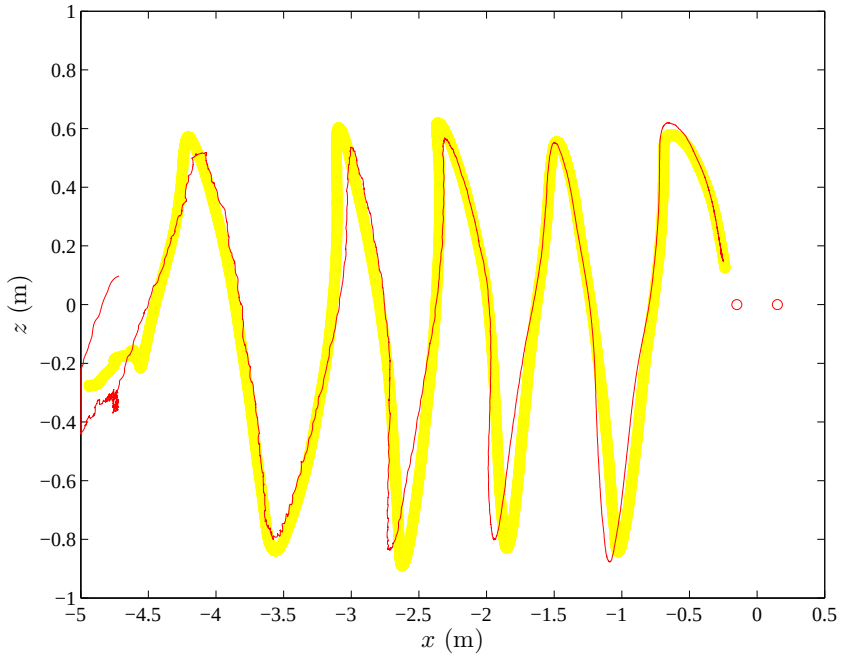


Figure 8-8: Position estimation of the sensor unit in the magnetic field during a slow sweep. The thin red line represents the position estimations while thick yellow line represents Vicon motion capture measurements. The red circles signify the position of the powerline.

Next, the electronics were placed on board the aircraft (see Figure 8-9). I then launched the aircraft from a crossbow at about 7.5 m/s and commanded it to execute an open loop perching trajectory. The maneuver occurred quickly enough ($<$ 1s) that planar approximations made earlier held. In this experiment I sought to examine the impact of high speed flight and an extremely limited number of time samples ($\approx$ 270) on position estimation. Therefore, as before, I ran the estimator offline, using pitch information from Vicon motion capture. As seen in Figure 8-10, the state estimation demonstrated tracking with a reasonable accuracy up to 4m from the wire.

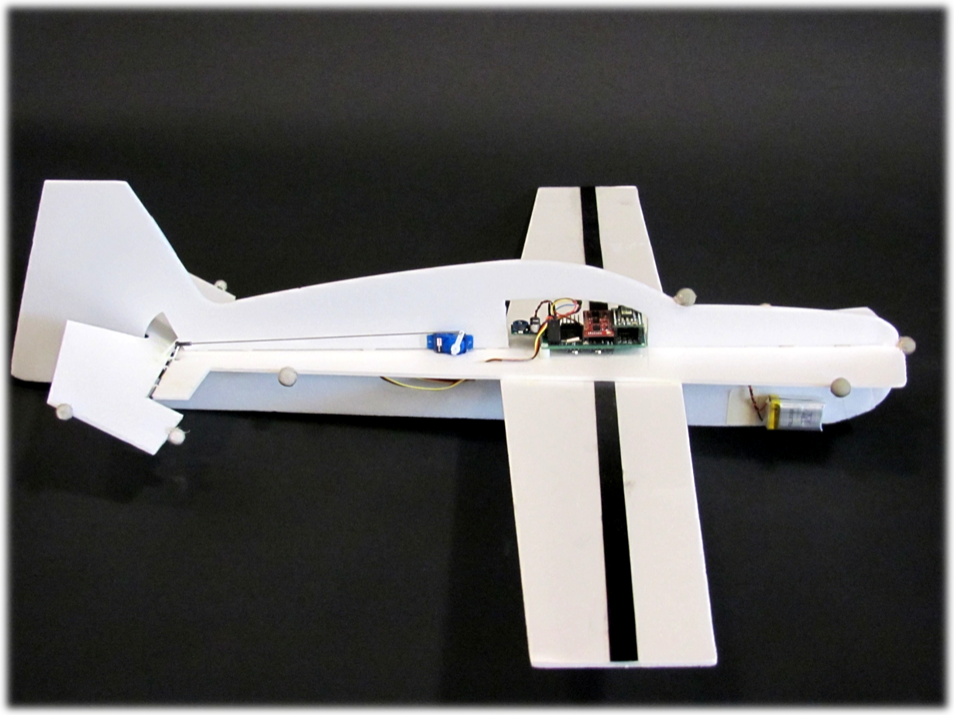


Figure 8-9: A photo of the instrumented glider.

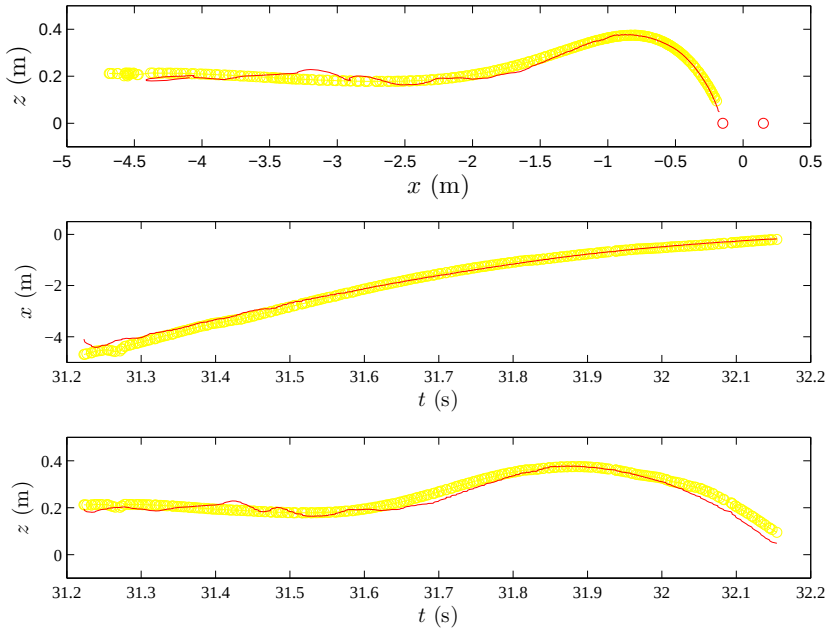


Figure 8-10: State estimation of aircraft position during a perching maneuver. The thin red line represents the position estimations while the thick yellow line represents the Vicon motion capture measurements.

8.7.2 Onboard Estimation

I then implemented the estimator in C and used a separate observer to estimate the pitch of the aircraft. To avoid the induced sensor noise associated with the high-g launches, I reset the position estimator after launch, using the positions estimated before launch as the initial conditions. I also employed a high-g accelerometer to obtain an initial velocity estimate for the vehicle after launch. The performance of this completely onboard, motion capture-free state estimator can be observed in Figure 8-12.

8.7.3 Closed-loop control

With the state estimator functioning onboard the aircraft, the final experiment was to close the loop on the state estimates. Following the work carried out in the previous chapters, I designed a new nominal perching trajectory for the heavier instrumented aircraft using direct collocation and closed the loop using TVLQR.

I first tested this controller in simulation, using simulated magnetic field measurements for feedback. When this was observed to yield satisfactory perching over a range of initial velocities, I implemented the controller on the real system. The results of the closed-loop, fixed-wing perching experiments using the magnetic field based state estimates for feedback can be seen in Figures 8-11 and 8-12. The results showed conclusively that the magnetic field based state estimates were sufficient to achieve fixed wing perching on a level comparable to the Vicon motion capture system when the initial speed of the aircraft was varied from 7.0-7.8 m/s.

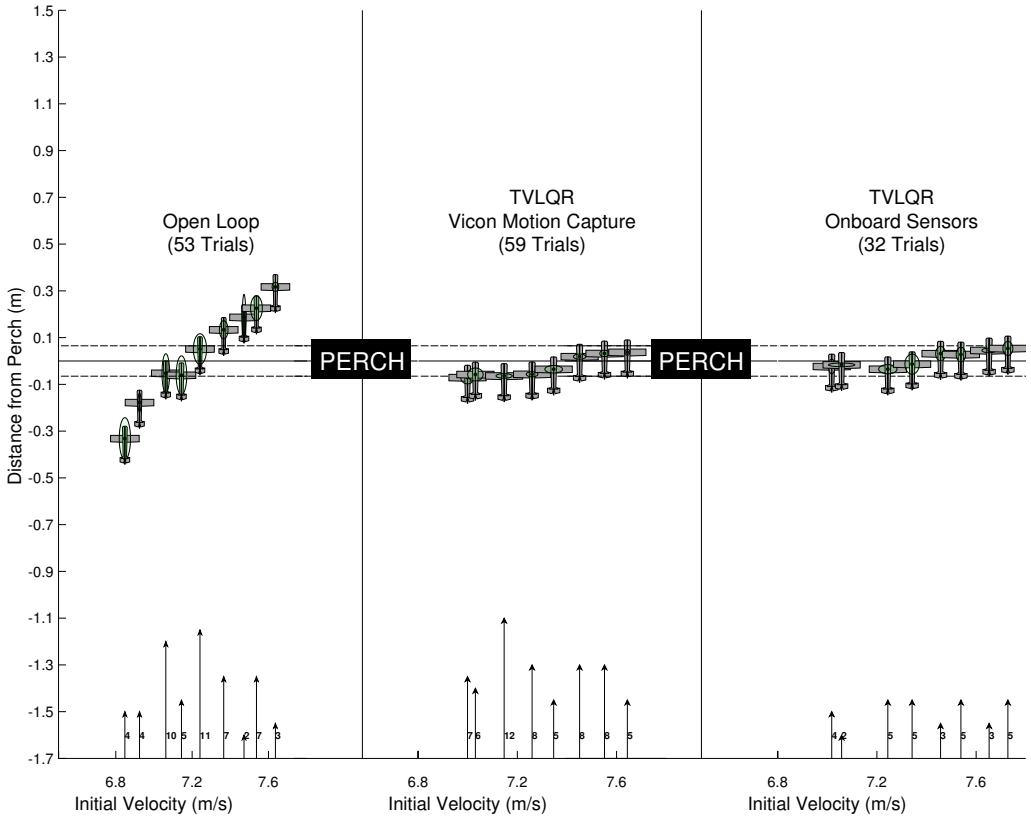


Figure 8-11: Comparison of the perching performance of open-loop control (left), feedback control using Vicon motion capture (middle), and feedback control using magnetic field sensing (right). For each case, final distance from perch is used as a performance metric, where 6.5 cm from the perch is considered to be a successful perch. As before, the individual markers represent the mean of multiple trials with similar initial velocities and the ellipses represent the spread in final positions and initial velocities.

8.7.4 Outdoor Flight Tests

After demonstrating that it is possible to successfully perch indoors *without* the Vicon motion capture system, the entire experimental powerline system was moved outdoors. I repeated the powerline perching experiments outdoors to test the performance of the perching UAV in the presence of wind gusts. As can be seen in Figure 8-13, when wind speeds were low, the perching experiments were successful. This figure shows three sequential perching trials using the magnetic field sensing system. However, when the vehicle experienced a significant wind disturbance, as shown in Figure 8-14, it failed quite catastrophically. In these experiments, no compensation was made by the controller for the wind gusts. This area of possible improvement will be explored further in the next two chapters.

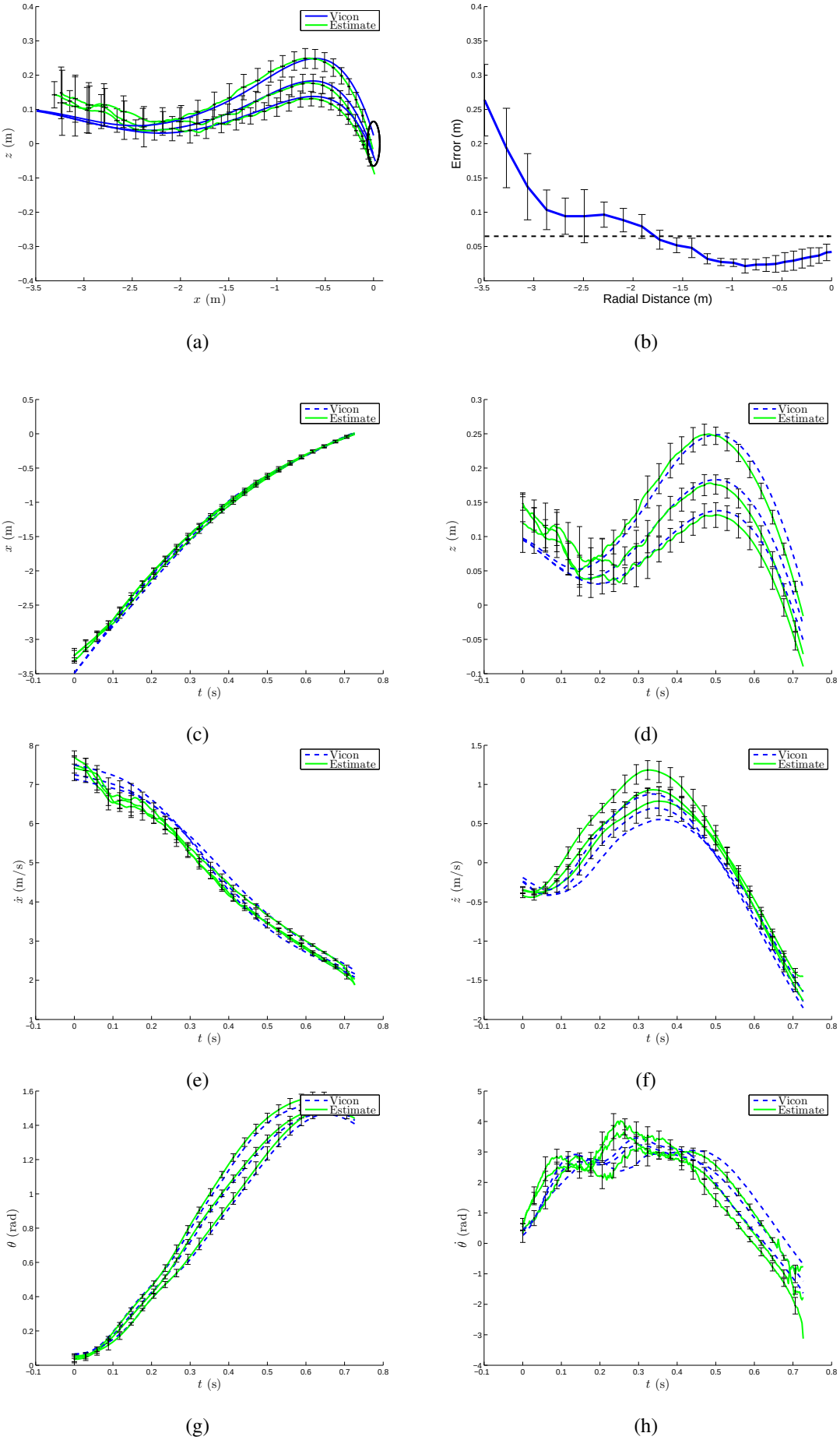
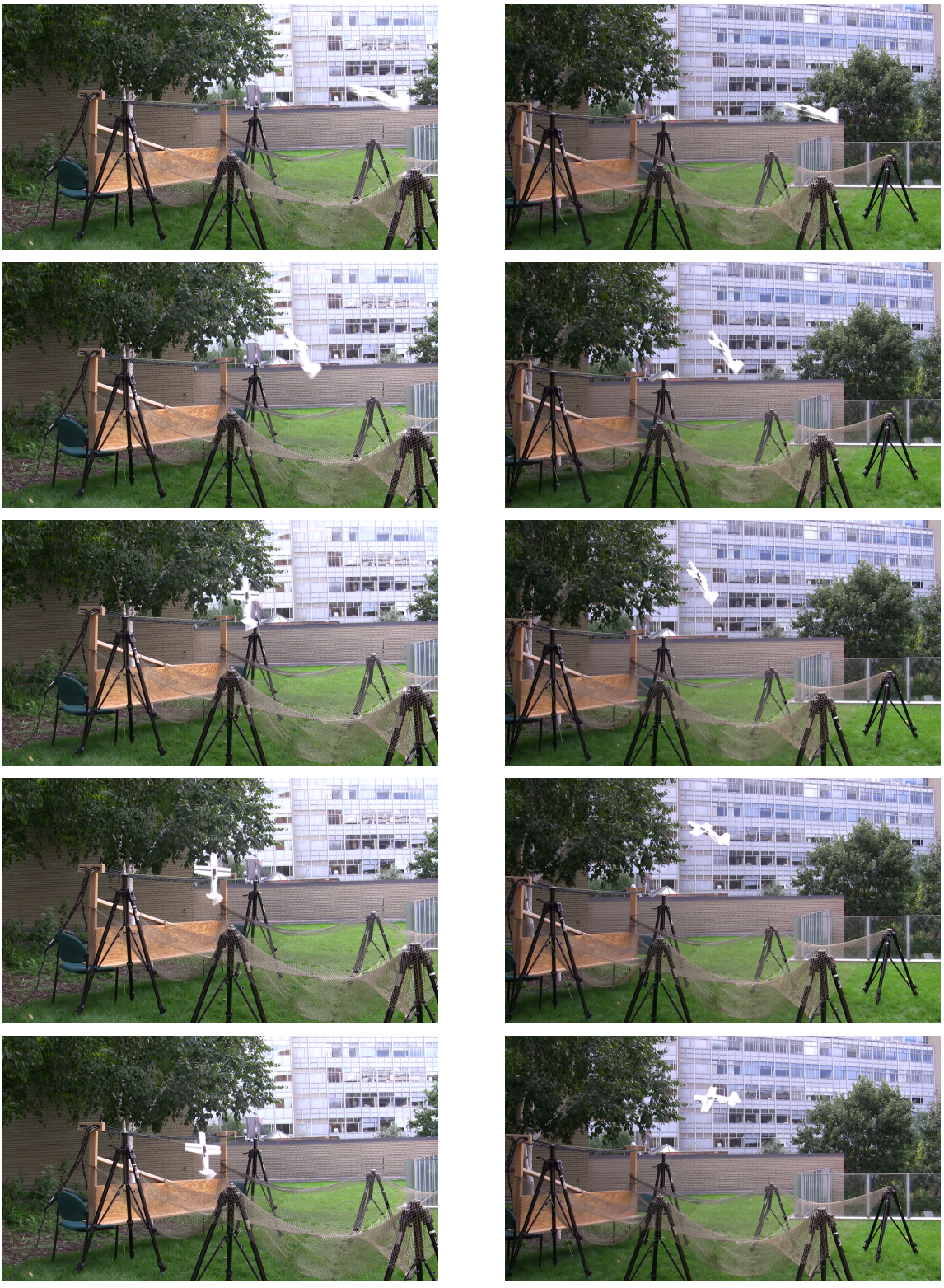


Figure 8-12: Plots comparing all major state estimates (green) with motion capture measurements (blue). Each curve represents the average of 4-6 trajectories with similar initial conditions. Error bars have been added to show estimate variation. The axis in the upper right corner shows a decrease in position estimation error as the aircraft approaches the perch.

(a) (b) (c)

Figure 8-13: These movie stills demonstrate three sequential outdoor perching maneuvers, using only onboard electronics. When there are not any significant winds, perching is successful.



(a) (b)

Figure 8-14: These movie stills demonstate the impact of winds gusts on outdoor perching. The frames in (a) show a successful outdoor perch. The frames in (b) show a subsequent flight where the aircraft is blown off course by wind.

Chapter 9

Wind Gust Modeling

As the outdoor experiments in the previous section have demonstrated, there is a considerable need to address the impact of external disturbances due to windgusts on the perching maneuver. One means of addressing these disturbances, which is particularly relevant to model-predictive control strategies, involves modeling the wind gusts themselves and then developing control strategies based on those models.

9.1 Wind Turbulence

The aerodynamics community has long history of developing models for windgusts and wind turbulence. While there are many ways to represent turbulent flows [16], the stochastic approach to modeling turbulence has the longest history in the literature [63]. These statistical models of turbulence find their origins in the work of Reynolds [85], who concluded from studying turbulence transitions in fluids that turbulence could not be modeled using deterministic means. However, it is the work of Taylor, Von Karman, and Kolmogorov that forms the foundation for the wind turbulence models used in the aerodynamics communities today.

Inspired by Reynolds findings, G.I. Taylor was the first researcher to model turbulence by the application of statistical methods [96]. Taylor's methods were also supported experimentally by wind tunnel turbulence measurements in [97, 30]. Von Karman then built on Taylor's work by using correlation tensors to model the statistical relationship between

fluid velocities at different positions in a turbulent flow [24]. In [43], Kolmogorov also employed a statistical approach to turbulence modeling, but did so by deriving an Energy spectrum for turbulent flows based on the way energy is dissipated in turbulent fluid structures.

Currently, in the aerodynamics literature, two major wind turbulence models have emerged, namely the Dryden Wind Gust Model [54] and the Von Karman Wind Gust Model [27]. Following in the steps of [96], [24] and [43], these models make some key assumptions about the turbulent nature of wind gusts and seek to derive relevant power spectra to represent the frequency characteristics of the flow.

9.1.1 Dryden Wind Turbulence Model

The Dryden Wind Turbulence model, also known as the Liepmann spectrum [37], was first described in [54] to model the response of an aircraft flying in a turbulent flow field. To construct this model, Liepmann built upon the experimental work of Dryden [30] and the analytical work of Taylor and Von Karman [96, 24]. Like those who came before him, he assumed that the turbulence was both homogeneous (statistics do not vary with position) and isotropic (statistics do not vary with coordinate frame). He also assumed that the changes in the wind field due to turbulence were much smaller than an aircraft's nominal forward speed and that, for this reason, the turbulent flow field could be approximated as stationary (not varying with time) in the global frame.

Using these assumptions, Liepmann applied Von Karman's correlation tensor to model the spatial dependence of the turbulent flow. He then converted this correlation tensor into a correlation *function*, transformed the correlation function into the time domain and applied the Fourier transform to find the corresponding power spectra. Because Liepmann selected a Gaussian-Markov process for his correlation function, the Dryden Power Spectrum was found to be:

$$\Phi_u(\omega) = \sigma_u^2 \frac{2L}{\pi U} \frac{1}{1 + (\frac{L\omega}{U})^2} \tag{9.1}$$

and

$$\Phi_v(\omega) = \sigma_v^2 \frac{L}{\pi U} \frac{1+3(\frac{L\omega}{U})^2}{(1+(\frac{L\omega}{U})^2)^2}, \tag{9.2}$$

where U is the nominal wind speed, L is the length scale of the turbulence, σ_u is the *rms* of the longitudinal turbulence velocities, σ_v is *rms* the vertical turbulence velocities, and ω is the temporal frequency.

9.1.2 Von Karman Wind Turbulence Model

A clear derivation of the Von Karman Wind Turbulence Model can be found in [40]. It differs from the Dryden Wind Turbulence model in its choice of correlation function. To find this correlation function, Von Karman defined an energy function, $E(\Omega)$, where Ω is the spatial frequency defined as $\Omega = \frac{\omega}{U}$. To construct this energy function, he relied on observations of turbulence made by Kolmogorov and Loitsyansky. In [43], Kolmogorov states that at high wave numbers, for a given energy cascade rate ε,

$$E(\Omega) = f(\varepsilon, \Omega) \approx C\varepsilon^x \Omega^y \approx C\varepsilon^{2/3}\Omega^{-5/3} \tag{9.3}$$

and in [57], Loitsyansky holds that for low wave numbers

$$E(\Omega) \sim \Omega^4. \tag{9.4}$$

Using these principles, von Karman interpolated to construct the energy spectrum

$$E(\Omega) = C\frac{(\frac{\Omega}{\Omega_0})^4}{(1+\frac{\Omega^2}{\Omega_0^2})^{\frac{17}{6}}}, \tag{9.5}$$

where

$$C = \frac{55}{9}\frac{L}{\pi}\sigma_u^2 \tag{9.6}$$

and

$$\Omega_0 = \frac{1}{1.339L}. \tag{9.7}$$

The power spectra for the Von Karman turbulence model can then be derived from this energy spectrum and shown to be

$$\Phi_u(\omega) = \sigma_u^2 \frac{2L}{\pi U} \frac{1}{(1+(\frac{1.339L\omega}{U})^2)^{\frac{5}{6}}} \tag{9.8}$$

and

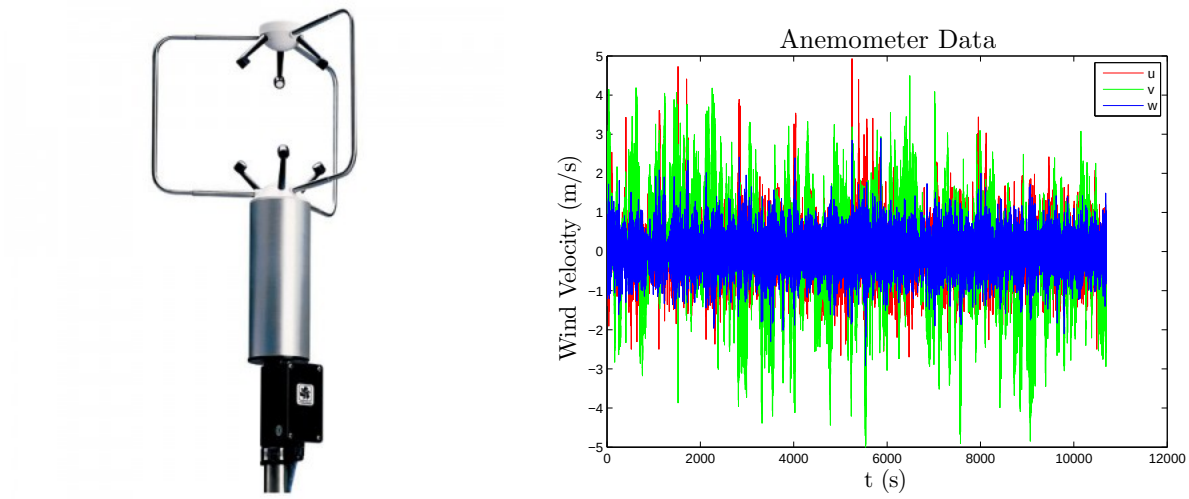
$$\Phi_v(\omega) = \sigma_v^2 \frac{L}{\pi U} \frac{1+\frac{8}{3}(\frac{1.339L\omega}{U})^2}{(1+(\frac{1.339L\omega}{U})^2)^{\frac{11}{6}}}. \tag{9.9}$$

9.2 ARMA Modeling

While these wind-gust models provide a reasonable means of modeling wind turbulence for large aircraft traveling at high speeds in a spatially varying turbulence field, they most likely do not provide the best representation of windgusts for a bird-scale perching UAV. Not only do they assume a very high forward speed for the vehicle compared to the wind measurements, but they also require second order models to capture the frequency characteristics of the turbulent flow. Because the aircraft will be executing a perching maneuver with a largely variable forward speed profile over the duration of the trajectory and because the SOS controller synthesis methods are limited in the number of systems states they can tolerate, I chose a different approach for modeling wind gusts. Instead of trying to fit the parameters in a second order model like the Dryden Wind Turbulence model, I decided to use tools from stochastic signal processing to fit a first order auto-regressive moving-average model (ARMA) to wind measurement data. This led to a reasonably straight forward and low-order approach for modeling the frequency characteristics of the wind. Also, unlike the Dryden and Von Karman wind turbulence models, my model assumes that, due to glider's short flight distance, the wind gusts are *not* varying with space, but with time.

9.2.1 Wind Turbulence Modeling Experiments

To validate my wind modeling approach, I acquired a 3D ultrasonic anemometer, RY-81000 which is able to measure velocities in all three axes down to 0.01 m/s with an update rate of 30 Hz. On a moderately windy day (i.e., weather forecast predicting $<$ 10 mph



(a) (b)

Figure 9-1: (a) RY-81000 ultrasonic anemometer. (b) Raw data measured over two hours on a moderately windy day.

winds), I placed the ultrasonic anemometer outside for about two hours and collected wind data. I then sought to fit a first order model to the wind dynamics. Using the MATLAB system identification toolbox, I was able to achieve models with an accuracy of approximately 75 percent. The result of these first order transfer function fits can be observed in Figures 9-2.

9.3 State Augmentation

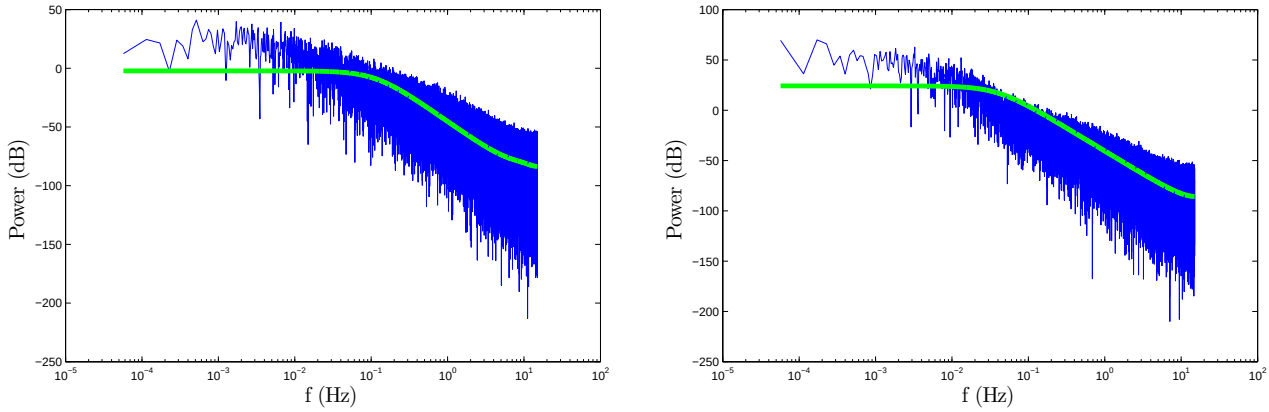
To integrate the ARMA models described in the previous section into the glider dynamics, I augmented the original glider states with the newly defined wind states and added the velocity states to the nominal air velocities acting on the glider's control surfaces in flight.

This can be further described as follows:

If the airspeed at the wing and elevator is given as

$$\dot{\mathbf{x}}_w = \begin{bmatrix} \dot{x} + l_w \dot{\theta} s_\theta \\ \dot{z} - l_w \dot{\theta} c_\theta \end{bmatrix} - \begin{bmatrix} u \\ v \end{bmatrix}, \quad \dot{\mathbf{x}}_e = \begin{bmatrix} \dot{x} + l\dot{\theta} s_\theta + l_e(\dot{\theta} + \dot{\phi}) s_{\theta+\phi} \\ \dot{z} - l\dot{\theta} c_\theta - l_e(\dot{\theta} + \dot{\phi}) c_{\theta+\phi} \end{bmatrix} - \begin{bmatrix} u \\ v \end{bmatrix},$$

using flat plate theory, the resulting aerodynamic forces on the vehicle can be approximated



(a) (b)

Figure 9-2: Plots compare experimental power spectra from real anemometer data with the first order AR models. (a) Shows the fit for the longitundinal component of the wind while (b) shows the fit for the vertical component of the wind.

by

$$\begin{aligned}
\alpha_w =& \theta - \tan^{-1}(\dot{z}_w, \dot{x}_w), \quad \alpha_e = \theta + \phi - \tan^{-1}(\dot{z}_e, \dot{x}_e) \\
\mathbf{F}_w =& \rho S_w |\dot{\mathbf{x}}_w|^2 \sin\alpha_w \mathbf{n}_w = f_w \mathbf{n}_w, \\
\mathbf{F}_e =& \rho S_e |\dot{\mathbf{x}}_e|^2 \sin\alpha_e \mathbf{n}_e = f_e \mathbf{n}_e.
\end{aligned}$$

The glider model then becomes

$$\begin{aligned}
m\ddot{x} =& -f_w s_\theta - f_e s_{\theta+\phi} \\
m\ddot{z} =& f_w c_\theta + f_e c_{\theta+\phi} - mg \\
I\ddot{\theta} =& -f_w l_w - f_e(l c_\phi + l_e),
\end{aligned}$$

where the wind model is given as

$$\begin{aligned}
\dot{u} =& a_x u + b_x w_x \\
\dot{v} =& a_z v + b_z w_z.
\end{aligned}$$

Here, u and v are the wind velocities, w_x and w_z are input uncertainty (bounded or stochastic), and a_i and b_i are low-pass filter coefficients determined from data.

Chapter 10

Aircraft Control in Wind

In the previous chapter, I described an approach for modeling the impact of windgusts on the fixed-wing glider. In this chapter, I will continue with this model to design a controller capable of including wind gusts measurements in the feedback law. I will then apply the verification methods described in Section 5.3 to create an LQR-Tree capable of compensating for significant wind disturbances. However, before I provide the details of this approach, I will first review some of the other aircraft control strategies that have existed for mitigating or even exploiting wind gusts.

10.1 Related Work

In the aerospace controls community, a number of efforts have been made to control UAVs in wind. One such approach has been *static soaring*, where the UAV harvests energy from atmospheric phenomena usually characterized by long-duration vertical airflows. By exploiting these phenomena, which include thermals, updrafts from large structures, and long period atmospheric oscillations, UAVs are able to gain altitude while maintaining fairly static, low acceleration flight conditions. Of all the static soaring techniques, exploiting thermals has probably received the greatest amount attention, most likely due to the use of thermals in cross country soaring. In [4], Allen et al. developed a UAV system capable of autonomously detecting and soaring on thermals. By keeping track of the system energy, their algorithm was able to estimate the center, radius, and drift of the thermal, and use that

information to apply human pilot-inspired controllers to ride the thermal and achieve prolonged flight time. In [68], Metzger et al. proposed an optimal approach for cross-country soaring where they sought to minimize the time to the goal by reducing time spent circling in the thermal while maintaining zero net altitude loss. Similarly, in [47], Langelaan used tree-based trajectory planning through a known wind field to minimize the energy needed to reach the goal region, and then extended this work in [15] through the use of an A* search.

Like static soaring, *dynamic soaring* has also been explored as an energy harvesting technique by many researchers. Unlike static soaring, which tends to keep the UAV in a fixed configuration, *dynamic soaring* takes advantage of a wind field gradient and the vehicle's dynamic capabilities to extract energy from the environment. One of the most famous examples of this occurs in nature where the wandering albatross uses the shear layer above the ocean to fly for long distances. Inspired by the albatross and other dynamic soaring birds, researchers have tried to implement these techniques on autonomous aircraft. In [9], Boslough flew sailplanes in a steep shear gradient and developed a computational toolkit for simulating and analyzing these energy harvesting maneuvers. In [112], Zhou generated optimal dynamic soaring trajectories for achieving different tasks such as traveling and loitering, and in [52], Lawrance et al. used a piecewise trajectory which consists of four flight phases to achieve dynamic soaring. In [108], Wharington moved away from optimal control methods and explored various heuristics for dynamic soaring, such as controlling pitch to achieve constant velocity and banking the vehicle away from horizontal air flows.

In addition to the aforementioned soaring methods, some researchers have even begun to consider harvesting energy from wind gusts. Unlike static soaring and dynamic soaring, this method of energy extraction requires reasoning about stochastic atmospheric conditions. In [78], Patel et al. developed a method for exploiting wind gusts by modeling a glider as a point mass and using the lift coefficient as a control input. Restricting themselves only to vertical wind gusts, the authors formulated the control law as linear gains applied to the aircraft's relative velocity and the wind gust velocity and then proceeded to search for these gains by running a genetic algorithm over multiple simulations. In [49], Langelaan et al. incorporated aircraft dynamics into the problem formulation and developed an

algorithm consisting of an outer-loop controller which maximized the instantaneous energy gain per distance and an inner-loop LQR controller which stabilized the aircraft's attitude. And while this approach did include the glider's dynamics, the authors constrained the vehicle to a near-level flight regime to keep the outer-loop controller from sending the vehicle in to a dive. In an effort to improve on this control law, Langelaan then explored another approach in [46], where instead of constraining the aircraft to trim conditions, he developed a controller with gains on the wind velocities which acted as a disturbance to a stabilizing controller. In a similar fashion to [78], Langelaan then ran a genetic algorithm on Dryden wind turbulence fields to find the optimal wind velocity gains.

To supplement the UAV energy harvesting tasks outlined above, a few efforts have been made to map the wind field itself, since knowledge of the spatial, time and stochastic characteristics of the wind field are crucial for successful energy extraction. In [48], Langelaan et al. developed a means of estimating the wind field with only air flow sensors, GPS, and an inertial measurement unit, and in [50], they demonstrated their approach experimentally by fitting a second-order polynomial to their wind field measurements. In [51], a different approach was used to achieve a similar goal. Here, Lawrance and Sukkarieh used Gaussian process regression to generate a stochastic map of the wind field and then used that map to find energy-gain paths via an expanding tree algorithm.

In considering the approaches to wind field and wind gust exploitation described above, it is important to note that, to my knowledge, there has not been any work done to exploit wind gusts for short, dynamic aerobatic maneuvers. Most of the previously cited work examine open-loop trajectory design in known wind fields or instantaneous energy harvesting in the presence of turbulence. To date, there has been little work done to reason about the vehicle's nonlinear dynamics under stochastic atmospheric disturbances or to incorporate wind gust velocities into nonlinear feedback design. In fact, most nonlinear feedback methods for aerial systems that have taken wind gusts in to consideration have been applied only to rotorcraft, which are not well suited to energy extraction due to their method of lift generation. Some of these rotorcraft-based approaches, such as in [3], have sought to design a controller without explicitly considering wind gusts and, instead, proceeded to verify its robustness to wind experimentally. Other approaches, such as [62, 18], have combined

nonlinear controllers, often derived from backstepping algorithms, with high gain input observers. These input observers estimate the perturbations due to wind on the rotorcraft as well as any unmodeled aerodynamics, and allow the controller to carry out adaptive disturbance cancellation. In addition to these adaptive approaches, there has been some work done, such as in [33], to make use of more classical H-infinity methods for the development a rotorcraft system robust to wind disturbances.

In contrast to the control approaches applied to rotorcraft, the design of controllers for fixed wing aircraft that take wind gusts into consideration have relied mostly on stochastic analysis. This work began with Liepmann in [54] and [53], where the impact of turbulence was addressed for a linearized aircraft model by modeling turbulence as white noise passed through a low pass filter. Linear stochastic theory was then used to derive the variance of the aircraft pitch from the power spectrum of the turbulence and the dynamics of the aircraft. In [39], Holley et al. built upon this work and developed a stochastic wind model which was then used to examine a closed-loop control system for landing maneuvers. Finally, Mutuel et al. applied a LQR controller to a linearized aircraft model in [73]. The authors compared controller design with and without the wind velocities added to the state space and used simulation to demonstrate that, when the wind velocities were added to the state space for a low bandwidth turbulence model, the system showed improved performance.

10.2 The Stochastic Regulator

As described in [12], the optimal control problem which seeks the minimization of a quadratic cost function in the presence of variable initial conditions, i.e. LQR, is identical to the solution of the optimal control problem which seeks to minimize a quadratic cost in the presence of Gaussian input disturbances. This is particularly advantageous here, since this means that one can expect the minimization of the system response when the wind states of the augmented fixed-wing glider model are being driven by white input noise. For this reason, to control the system in 9.3, I will continue to use TVLQR to apply local feedback along the nominal perching trajectories.

To test this feedback design, in Figures 10-1 and 10-2, I simulated the augmented glider

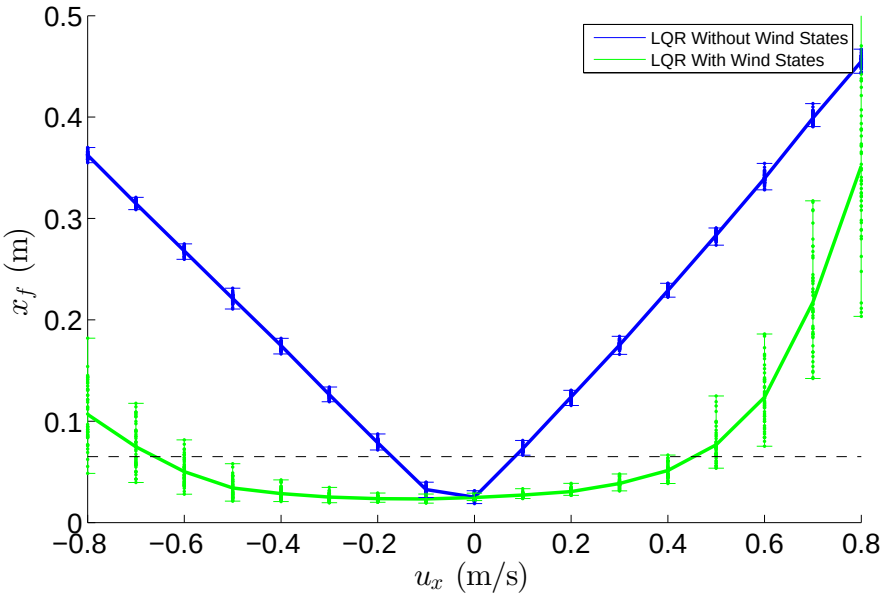


Figure 10-1: Performance of TVLQR with (green) and without (blue) wind state feedback for initial U wind velocity perturbation. The error bars indicate the variation with final cost due to random initial conditions and input uncertainty.

system over a range of initial wind conditions and allowed for random initial wind states along with random inputs to the wind state models. I used the first-order systems identified in Section 9.2.1 to generate the simulated wind states and I also allowed for random initial state conditions, which are indicated by the multiple sample points at each wind speed. The results in Figures 10-1 and 10-2 demonstrate that there is a significant performance advantage to including the wind states in the feedback design.

10.3 Robust Verification with Wind Models

10.3.1 Robust Verification

In Chapter 6, I demonstrated that funnels could be constructed for the case where the dynamics of the system can be described as having some bounded, unknown uncertainty. I can also apply this to the case of uncertain wind dynamics. Instead of modeling the wind dynamics as being a low-pass filter driven by some white noise input, I can model the wind as being a low-pass filter driven by bounded input uncertainty. I can then enforce the following constraints using SOS:

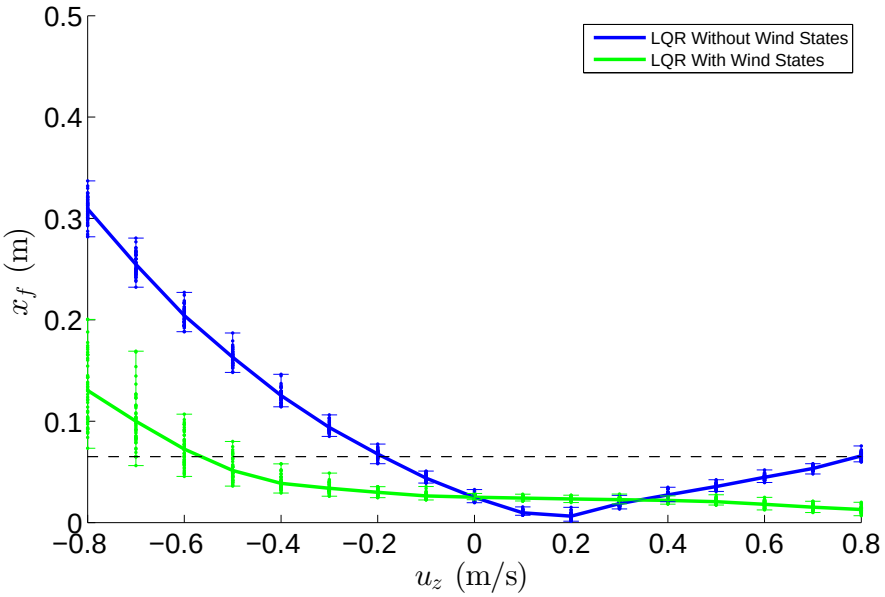


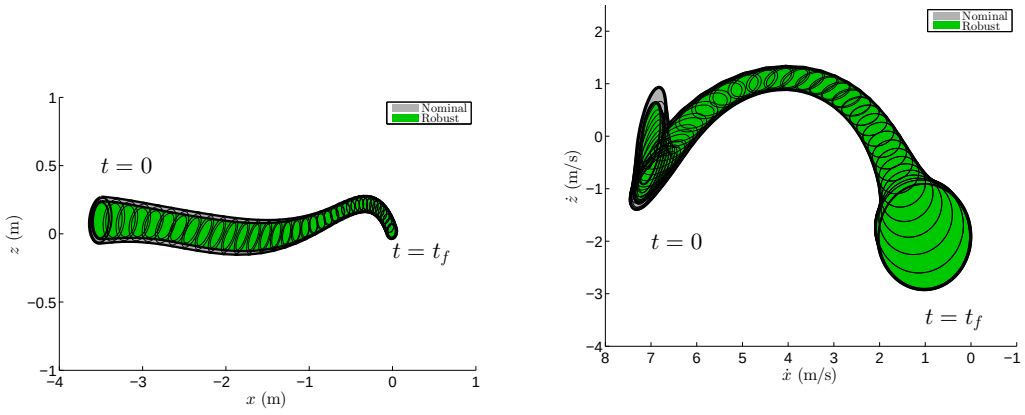
Figure 10-2: Performance of TVLQR with (green) and without (blue) wind state feedback for initial V wind velocity perturbation. The error bars indicate the variation with final cost due to random initial conditions and input uncertainty.

$$V(t,\mathbf{x}) < \rho(t) \implies$$
$$\dot{V}(t,\mathbf{x},\mathbf{w}) = \frac{\partial V(t,\mathbf{x})}{\partial \mathbf{x}}(\mathbf{f}(\mathbf{x},\mathbf{w}) + \mathbf{g}(\mathbf{x},\mathbf{w})\mathbf{u}) + \frac{\partial V(t,\mathbf{x})}{\partial t} < \dot{\rho}(t)$$
$$\forall \mathbf{w} \in [\mathbf{w}_1, \mathbf{w}_2]$$

where $\mathbf{w}$ is $\begin{bmatrix} w_x \\ w_z \end{bmatrix}$ and $\rho(t)$ is the value of the level set of interest.

Here I show an example of a robust funnel for a system which applies feedback to wind gust measurements, where the windgusts are modeled as a low-pass filter driven by a bounded input noise. The low pass filter has a time constant of 10 s, a gain of 1, and a bounded uncertain input of +/- 1.

As before, these funnels can be used to build an LQR-Tree over a space of nominal wind speeds to build a feedback law capable of controlling the glider over a much larger range of wind disturbances.



(a) (b)

Figure 10-3: (a) depicts a slice in the x-z and (b) a slice in the $\dot{x}$-$\dot{z}$ dimension.

10.3.2 Stochastic Verification

In a manner similar to robust verification methods described above, it is also possible to apply the stochastic verification methods described in Section 6.3. Instead of bounding the uncertain inputs to the wind gust models, I can instead incorporate the statistics of that white noise input directly in to the verification procedure. However, because the stochastic funnels in Section 6.3 were overly conservative in the case of dynamic uncertainty, I do not proceed with them any further here.

10.3.3 Adaptive Control Design and Verification

There is yet a third approach to achieve fixed-wing perching in wind gusts and that is through adaptive control. As mentioned in Section 6.4.2, it is possible to use the SOS verification approach to design adaptive controllers which can adapt to constant uncertain parameters that appear linearly in the dynamics. If mean wind speeds change very slowly compared to the length of the maneuver, instead of modeling the wind dynamics as white noise passed through a low-pass filter, the wind can be modeled as a constant unknown parameter which enters nonlinearly in to the dynamics. Although the methods described in Section 6.4.2 only apply when the uncertain parameter enters linearly in the dynamics, I can modify the approach to design adaptive controllers for the case where the parameter enters *non-linearly* in the dynamics.

Consider the system

$$\dot{\mathbf{x}} = f(\mathbf{x},\theta) + \mathbf{g}(\mathbf{x})\alpha(\mathbf{x},\hat{\theta}),$$

and the Lyapunov function

$$V = V_a + \frac{1}{2}\tilde{\theta}^T\Gamma\tilde{\theta} + \frac{1}{2}\theta^T\Psi\theta,$$

where $\tilde{\theta} = \hat{\theta} - \theta$. Taking the derivative, I then have

$$\dot{V} = \mathbf{x}^T\mathbf{S}_a(\mathbf{f}(\mathbf{x},\theta) + \mathbf{g}(\mathbf{x})\alpha(\mathbf{x},\hat{\theta})) + \dot{\tilde{\theta}}^T\Gamma\tilde{\theta}.$$

If I choose $\alpha(\mathbf{x},\hat{\theta}) = \mathbf{K}_{LQR}\mathbf{x} + \mathbf{z}(\mathbf{x})\hat{\theta}$, I can then write

$$\begin{aligned}\dot{V} &= \mathbf{x}^T\mathbf{S}_a\mathbf{f}(\mathbf{x},\theta) + \mathbf{x}^T\mathbf{S}_a\mathbf{g}(\mathbf{x})(\mathbf{K}_{LQR}\mathbf{x} + \mathbf{z}(\mathbf{x})\hat{\theta}) + \dot{\tilde{\theta}}^T\Gamma\tilde{\theta} \\ &= \mathbf{x}^T\mathbf{S}_a f(\mathbf{x},\theta) + \mathbf{x}^T\mathbf{S}_a\mathbf{g}(\mathbf{x})\mathbf{K}_{LQR}\mathbf{x} + \mathbf{x}^T\mathbf{S}_a\mathbf{g}(\mathbf{x})\mathbf{z}(\mathbf{x})\hat{\theta} + \dot{\tilde{\theta}}^T\Gamma\tilde{\theta}.\end{aligned}$$

Choosing $\dot{\hat{\theta}}^T = -\mathbf{x}^T\mathbf{S}_a\mathbf{g}(\mathbf{x})\mathbf{z}(\mathbf{x})\Gamma^{-1}$ and remembering that $\dot{\tilde{\theta}}^T = \dot{\hat{\theta}}^T$, I have

$$\dot{V} = \mathbf{x}^T\mathbf{S}_a\mathbf{f}(\mathbf{x},\theta) + \mathbf{x}^T\mathbf{S}_a\mathbf{g}(\mathbf{x})\mathbf{K}_{LQR}\mathbf{x} + \mathbf{x}^T\mathbf{S}_a\mathbf{g}(\mathbf{x})\mathbf{z}(\mathbf{x})\theta \leq 0.$$

An inspection of $\dot{V}$ reveals that SOS verifcation will produce an adaptation law which can be derived even when θ enters the dynamics nonlinearly. Unfortunately, as before, while useful for generating an adaptive control law, the LQR-Tree approach suffers because $\hat{\theta}$ and θ are part of the Lyapunov function, making evaluating the funnels at real-time impossible. Because I expect the nominal wind speeds to vary significantly from one flight to another, it would be impossible to know at run time which controller in the tree to choose, since in the adaptive approach (i.e., no wind sensor), the nominal wind speed would not be known apriori.

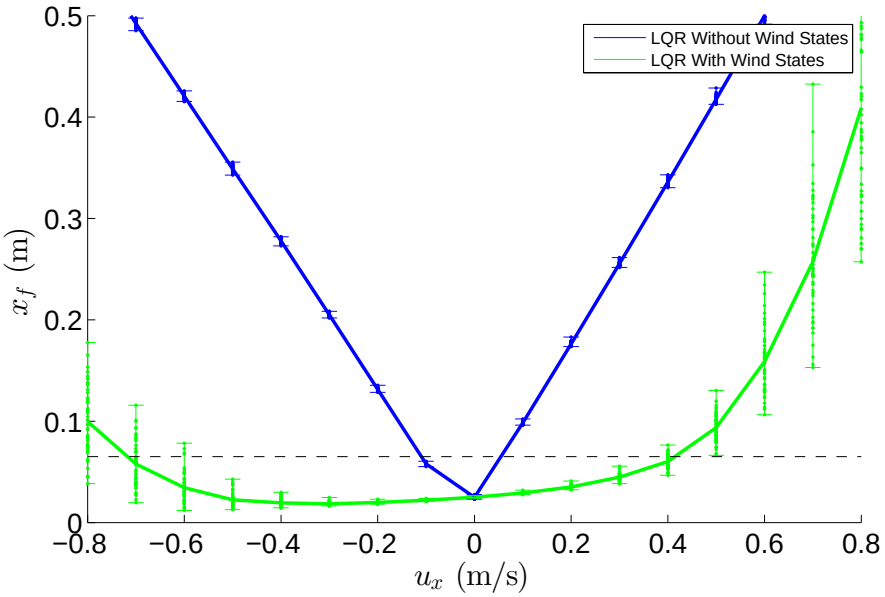


Figure 10-4: Performance of TVLQR with (green) and without (blue) wind state feedback for initial U wind velocity perturbation. The error bars indicate the variation with final cost due to random initial conditions and input uncertainty. These simulations use a second-order wind model.

10.4 Impact of Wind Model on Control

Although it is possible to achieve robust verification by bounding the magnitude of the input disturbance to the wind gust model, this approach still does not address other sources of model uncertainty, such as higher order wind dynamics.

10.4.1 Impact of Wind Model Order on Control

Many of the conventional wind gust models described in 9.3 are actually second-order low-pass filters rather than simple first-order systems. To test the impact of model order on the controller, I designed the TVLQR controller using a first-order model, but I *simulated* the controller's response using a second-order wind model with a unity gain and approximately the same cut-off frequency. The results can be seen in Figures 10-4, 10-5, and 10-9. As is clearly illustrated, there is little difference between simulating the system with a first-order and second-order wind model.

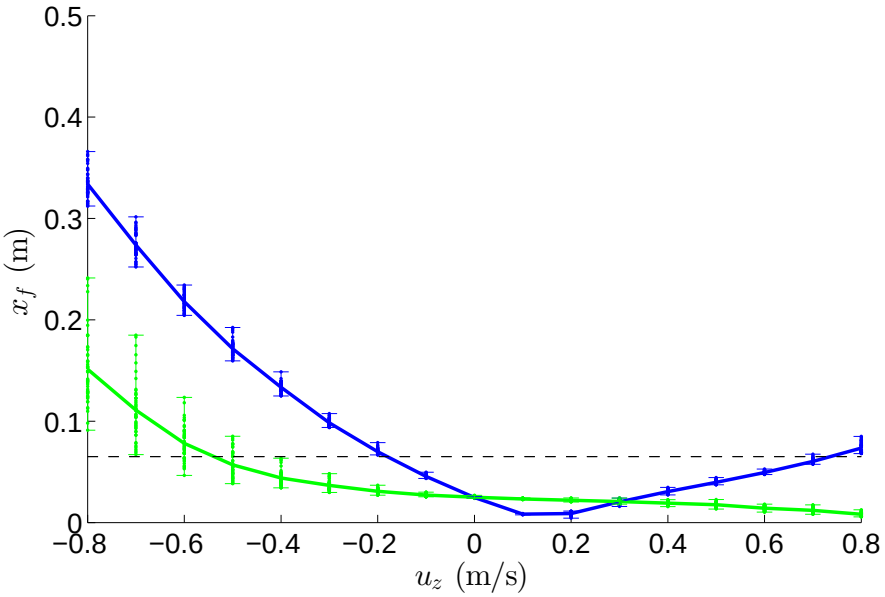


Figure 10-5: Performance of TVLQR with (green) and without (blue) wind state feedback for initial V wind velocity perturbation. The error bars indicate the variation with final cost due to random initial conditions and input uncertainty. These simulations use a second-order wind model.

10.4.2 Impact of Wind Model Cut-off Frequency

As was mentioned in the previous subsections, changing the order of the wind gust model seems to have little effect on the performance of the TVLQR Controller; this is most likely due to the role that cut-off frequency has on the controller performance. The TVLQR approach has the ability to generate a controller for a system which is slowly converging. That is, by the time the horizon time has been reached, not all of the states are required to have converged to the nominal trajectory. Rather, it is only necessary that the cost of the system be decreasing along the nominal trajectory. Therefore, TVLQR works quite well for designing a controller to handle a constant offset in nominal wind speed. As mentioned above, it also is well suited for designing a controller in the situation where there are small stochastic perturbations in that constant wind speed, so long as these perturbations can be measured. For this reason, the closer the cut-off frequency of the wind model is to zero, the better the controller is at compensating for the wind speed variations. This was also noted in [73] for the LTI case.

To demonstrate this, I simulated the TVLQR controller under a range of initial wind

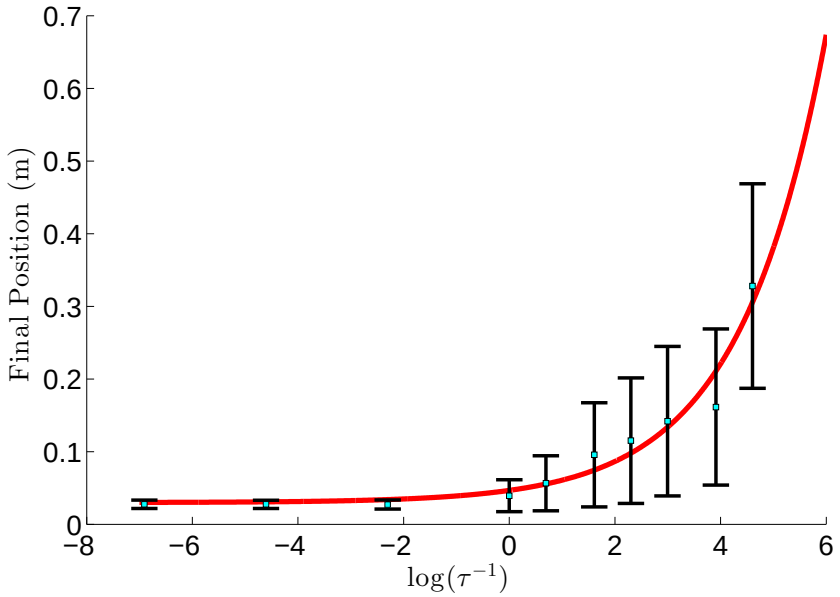


Figure 10-6: Performance of TVLQR with wind state feedback as a function of wind model time constant. Note that as the time constant falls below the length of the maneuver (1s), the controller begins to perform much more poorly.

speed offsets (from -0.3 to 0.3 m/s) as well as wind model time constants (from 1000 s to 0.01 s) and plotted the final position of the glider. As can be seen in Figure 10-6, as the frequency increases, the controller performance worsens even though the gain of the low-pass filter and the magnitude of the input disturbances remain the same.

10.5 LQR-Tree Approach

Unfortunately, these single time-varying controllers are only able to control the aircraft over a small set of wind speeds. As mentioned above, to improve the perching performance of the aircraft over a larger range of wind conditions, an LQR-Tree could be used. In Section 5.5, an LQR-Tree was used to improve perching performance under variable initial conditions. By applying our first-order wind speed model and effectively treating the wind gusts as a slowly varying wind speed offset, the same approach can be used to improve perching performance in wind. Following the work described in Section 5.5, I generated an LQR-Tree for the augmented glider system and attempted to improve coverage over initial wind speed as well as the other state variables. Figure 10-8 compares the performance of

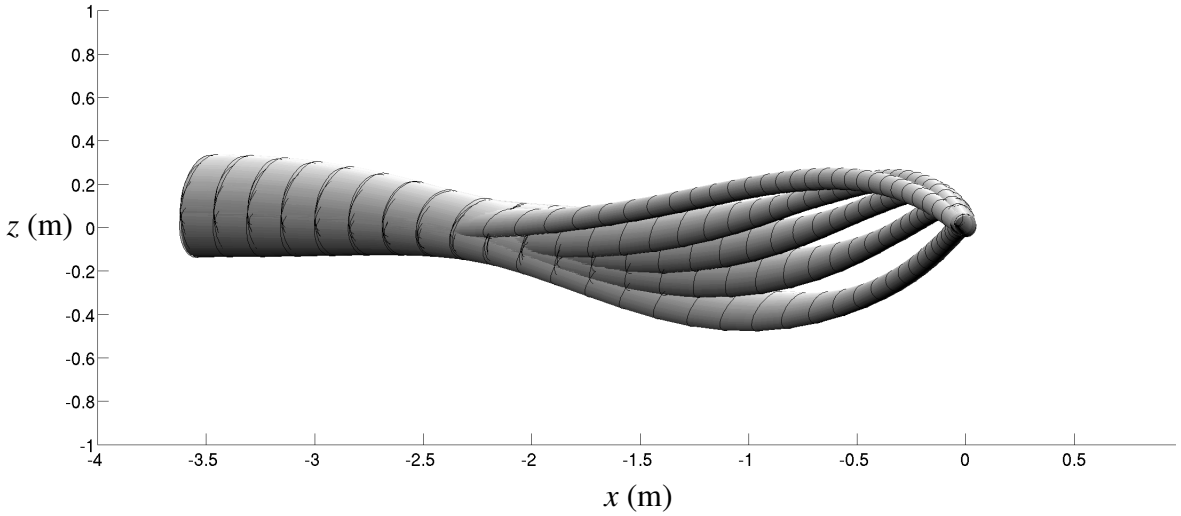


Figure 10-7: Slice of LQR-Tree in x and z dimensions designed for varying horizontal wind speeds. Notice that for larger head winds, the aircraft dips down lower to execute the perching maneuver.

the LQR-Tree with that of a single nominal TVLQR stabilized trajectory which does not account wind gusts. The results show a dramatic improvement in robustness to windgusts compared to that of a single stabilized trajectory. I also applied the control strategy to a higher (second) order wind model and demonstrated that the LQR-Tree controller still performed adequately (see Figure 10-9).

10.6 Outdoor Wind Experiments

The powerline perching hardware presented in Chapter 8.7.4 provides a unique opportunity to test the methods I have proposed for perching outdoors when wind is non-negligible. In this section, I will describe an experimental approach for testing the performance of my control designs in windy environments, and I will summarize some of the preliminary results of these experiments.

10.6.1 Outdoor Experimental Set-up

To test the controller on real hardware, I used the powerline experimental system described in Chapter 8.7.4 and the three-dimensional ultrasonic anemometer described in Chapter 9.3 to acquire the state estimates for the aircraft and the wind. I attempted to

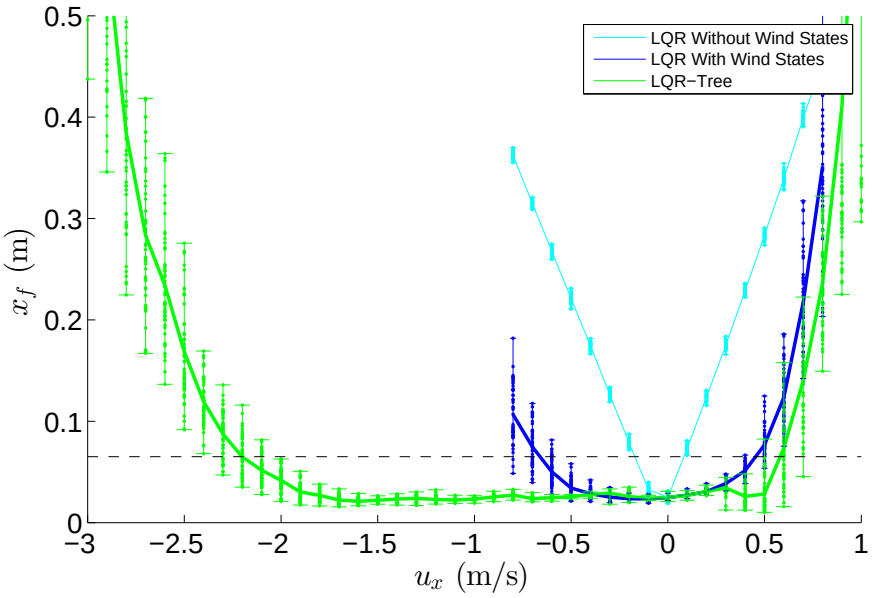


Figure 10-8: Performance of LQR-Tree (green) and single TVLQR trajectory (blue) with wind state feedback for initial U wind velocity perturbation. The error bars indicate the variation with final cost due to random initial conditions and input uncertainty. The performance of TVLQR without wind state feedback (cyan) is provided as a reference. These simulations all use a first-order wind model.

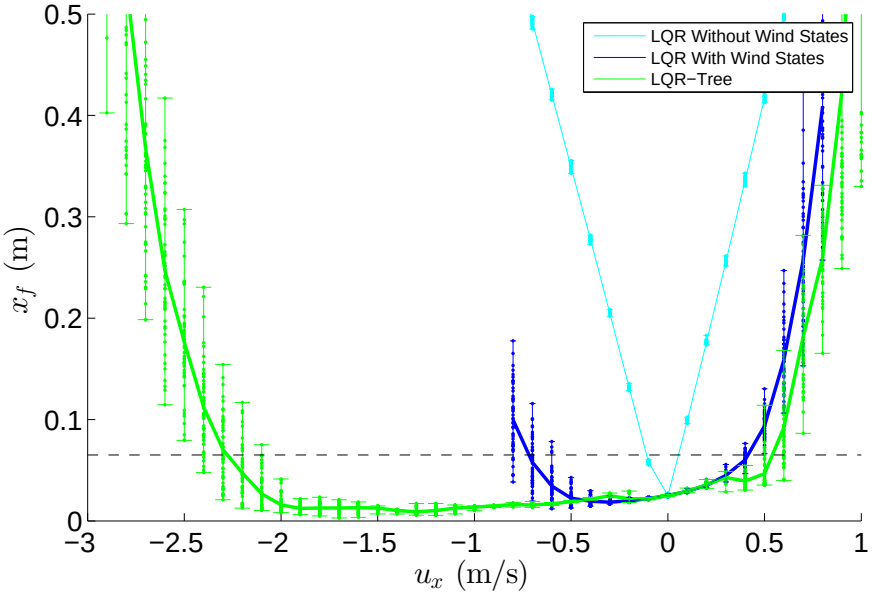


Figure 10-9: Performance of LQR-Tree (green) and single TVLQR trajectory (blue) with wind state feedback for initial U wind velocity perturbation. The error bars indicate the variation with final cost due to random initial conditions and input uncertainty. The performance of TVLQR without wind state feedback (cyan) is provided as a reference. These simulations all use a second-order wind model.

Figure 10-10: Outdoor experimental setup.

position the outdoor experimental set up in an area with low cross winds and place the ultrasonic anemometer close to the perch but well away from the flightpath of the vehicle. Sensor data was sent from the ultrasonic anemometer to the base station via ethernet and then from the base station to the aircraft via Wi-Fi. These wind speed measurements were then included as part of the estimated UAV state and therefore acted as an input to the control law. The base station also monitored the wind speed in the vicinity of the perch, sounding an alarm when the wind was within a reasonable range ($<$ +/- 1 m/s in all directions). It was the goal of these experiments to demonstrate some improvement in perching response when the wind gusts were incorporated in to the control design. A photo of this outdoor experimental set-up can be seen in Figure 10-10.

10.6.2 Control Approach

To improve the perching performance of the UAV in wind gusts, I followed the work carried out in the previous section and designed a small LQR-Tree which attempted to more adequately cover the initial wind velocities encountered by the glider. In the end, the LQR-Tree consisted of five stabilized trajectories, the nominal, two trajectories designed for a head wind of 0.5 m/s and 1.0 m/s, and two trajectories designed for up and down drafts of 0.5 m/s. At run time, the glider sensed the wind speed and selected the controller

to maximize its perching performance. Some preliminary perching results using wind for feedback can be seen in Figure 10-11. In this figure, the glider is perturbed by a head wind, as can be seen in the vehicle's sudden change in yaw. However, by using the wind measurements, the controller is able to intelligently reason about this headwind and ensure that the glider still lands on the perch.

Figure 10-11: Outdoor perching results with wind feedback.

Chapter 11

Conclusion

The work presented in this thesis demonstrates on real hardware the value of SOS verification techniques in generating a library of local nonlinear controllers which, when combined, can produce extremely robust performance for highly underactuated systems. Not only has this thesis addressed improving robustness to variable initial conditions, but it has also presented methods for handling dynamic and parametric model uncertainty, as well as methods for handling external environmental disturbances. However, there are several issues with the approach presented in this thesis which need to be explored further.

Three-dimensional effects are one set of phenomena which have not been addressed in this thesis, neither for the control nor the estimation problem. For the estimation problem, an effort has been made in [70], to address them, but I will summarize the main points of that discussion here. Interestingly enough, it is moving away from the experimental set-up and into the field that solves the three-dimensional estimation problem. In a real world scenario, all powerlines will possess significant sag due to the weight of the cable. This sag produces magnetic field components in the lateral directions, and if the amount of sag in the wire is known, this provides the required information to estimate the aircraft in all three dimensions. The sag of the wire, while not known a priori, can be estimated online by adding a second sensor on to the vehicle and incorporating the constant wire sag parameter into the state of the Extend Kalman Filter. Both current and wire spacing can also be estimated similarly, as is shown in [70].

The control approach extends quite naturally to three dimensions, though some care

may be required to reduce the number of states for verification. Both the nonlinear optimization method and the time-varying linear feedback can be applied directly to a three dimensional model of the aircraft. For system identification, a three dimensional flat-plate model can still be used for a baseline, while radial basis functions can be still be added to the three-dimensional aerodynamic coefficients to improve model accuracy. With the increase in state, however, SOS verification will become much more costly. If the same flat-plate baseline model is used, the total number of states for the three-dimensional aircraft will be 15 - the very limit for what can be handled by most SOS optimization software. Even if three states are removed, and direct control over control surface position is assumed, the aircraft will still possess 12 states. It should be possible to generate funnels for such a high dimensional system with the existing software tools. Nevertheless, there will be very little room, if any, for adding variables to compensate for uncertainty or adaptation. For this reason, for these three dimensional systems, it will be necessary to explore other SOS techniques for verification. Recently, a few new approaches for verifying high dimensional systems have emerged [60]. These approaches, known as DSOS and SDSOS, restrict the set of polynomials to a specific subclass which can be searched over more efficiently using Linear Programming or Second Order Cone Programming instead of a Semidefinite Programming. In [60], the authors demonstrate their approach on a system with 30 states and 14 inputs! This incredible feat lends credibility to the claim that the methods presented in this thesis will be able to scale effectively not only to three dimensions, but also to many more complex, real-world robotic systems.

In fact, these new DSOS and SDSOS approaches open a wide range of possibilities for additional studies with the perching glider system. Because of state size limitations, no effort was made in this thesis to try to verify the entire powerline perching system, that is, the system which comprises both the controller and the estimator. For the purposes of my experiments, I assumed perfect state information when executing the LQR-Tree controller. However, the state estimator for the powerline perching UAV will undoubtedly have its own dynamics and noise that will impact size of the backwards reachable set. One could imagine including the estimator dynamics and the sensor noise in to the SOS verification step to get a more accurate representation of the true backwards reachable set. Similarly,

the adaptive control design approach presented in this thesis suddenly becomes much more tractable. In Section 6.4, I mention that two SOS variables are added for every unknown parameter. This could easily yield unwieldy SOS programs for systems of reasonable size, like the glider. With an unknown constant offset on each acceleration, suddenly, the number of SOS variables jumps from 7 in the nominal case to 13 in the adaptive controller case. With DSOS or SDSOS, however, such an increase in state size will not be as detrimental

A third area of potential investigation involves trying to reduce the conservatism of the funnels that are generated using SOS. Because a quadratic form was chosen for the time-varying Lyapunov function, this inevitably restricts the tightness of the backwards reachable set. For instance, because the true backwards reachable set is not symmetric about the nominal trajectory, only searching over the set of symmetric forms may stunt funnel growth when it reaches the boundary of the set in one direction but not the other. This is especially true for the robust verification case, where, as seen in Figure 6-1, the backwards reachable set can take on all sorts of non-symmetric, non-convex shapes. One could imagine that this restricted Lyapunov function parametrization could, in some cases, severely limit the range of uncertainty that can be verified. One method to improve on this would be to allow the Lyapunov function to be represented by a higher degree polynomial, though this has the potential to drastically increase optimization run time. Another means of addressing this issue would be to also optimize over the quadratic form's center as well. Because the methods presented here only check the boundary of the funnel without guaranteeing convergence to the nominal trajectory, this could be a feasible means of reducing some of the conservatism due to symmetry.

Overall, however, the results presented in this thesis provide convincing evidence that verification techniques can be useful for controller synthesis and improved system performance. I am able to show that an incredibly underactuated system is able to perch robustly from a wide range of initial conditions, even when faced with sensor noise and external disturbances. The work also highlights the effectiveness of model predictive control strategies in situations where exploiting system dynamics is essential. The author hopes that the reader will be encouraged by the promising results presented in this thesis, so much so, that the reader tries to apply some of the methods described here to whatever challenging,

underactuated control problem he or she may be faced with. For too long, many in the controls community have viewed controller verification with skepticism, since much of its treatment has been theoretical and simulation based in nature. It is my hope that this thesis provides clear and convincing evidence that controller verification and feedback motion planning can be a useful and practical tool for control system design.

Bibliography

[1] Richard J. Adams, James M. Buffington, and Siva S. Banda. Design of nonlinear control laws for high-angle-of-attack flight. *Journal of Guidance, Navigation, and Control*, 17(4):737–746, July-August 1994.

[2] Charles W. Alcorn, Mark A. Croom, and Michael S. Francis. The X-31 experience: Aerodynamic impediments to post-stall agility. In *Proceedings of the 33rd Aerospace Sciences Meeting and Exhibit*. AIAA 95-0362, January 9-12 1995.

[3] K. Alexis, G. Nikolakopoulos, and A. Tzes. Constrained optimal attitude control of a quadrotor helicopter subject to wind-gusts: Experimental studies. *Proceedings of the American Control Conference*, pages 4451–4455, July 2010.

[4] Michael J. Allen. Guidance and control of an autonomous soaring UAV. *NASA/TM-200-214611*, February 2007.

[5] Angela M. Berg and Andrew A. Biewener. Kinematics and power requirements of ascending and descending flight in the pigeon (columba livia). *The Journal of Experimental Biology*, (211):1120–1130, February 2008.

[6] Angela M. Berg and Andrew A. Biewener. Wing and body kinematics of takeoff and landing flight in the pigeon (columba livia). *The Journal of Experimental Biology*, (213):1651–1658, January 2010.

[7] John T. Betts. Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207, 1998.

[8] Hugo De Blauwe, Selcuk Bayraktar, Eric Feron, and Faith Lokumcu. Flight modeling and experimental autonomous hover control of a fixed wing mini-UAV at high angle of attack. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2007.

[9] Mark B.E. Boslough. Autonomous dynamic soaring platform for distributed mobile sensor arrays. Technical Report SAND2002-1896, Sandia National Laboratories, 2002.

[10] James Buffington, Richard Adams, and Siva Banda. Robust, nonlinear, high angle-of-attack control design for a supermaneuverable vehicle. *Proceeings of the AIAA Guidance, Navigation, and Control Conference*, pages 690–700, 1993.

[11] Daniel J. Bugajski and Dale F. Enns. Nonlinear control law with application to high angle of attack. *Journal of Guidance, Control, and Dynamics*, 1992.

[12] Jeff B Burl. *Linear Optimal Control: H (2) and H (Infinity) Methods*. Addison-Wesley Longman Publishing Co., Inc., 1998.

[13] A.C. Carruthers, A.L.R. Thomas, S. M. Walker, and G. K. Taylor. Mechanics and aerodynamics of perching manoeuvres in a large bird of prey. *The Aeronautical Journal*, 114(1161):673–680, November 2010.

[14] Anna C. Carruthers, Adrian L.R. Thomas, and Graham K. Taylor. Automatic aeroelastic devices in the wings of a steppe eagle Aquila nipalensis. *Journal of Experimental Biology*, 210(23):4136–4149, 2007.

[15] Anjan Chakrabarty and Jack W. Langelaan. Flight path planning for uav atmospheric energy harvesting using heuristic search. *AIAA Guidance, Navigation, and Control Conference*, August 2010.

[16] Gary T. Chapman and Murray Tobak. *Observations, Theoretical Ideas, and Modeling of Turbulent FlowsPast, Present, and Future*. Springer-Verlag, 1985.

[17] B. Chavanne. Small uavs may recharge on powerlines. *Aviation Week DTI*, 2007.

[18] T. Cheviron, F. Plestan, and A. Chriette. A robust guidance and control scheme of an autonomous scale helicopter in presence of wind gusts. *International Journal of Control*, 82(12):2206–2220, December 2009.

[19] S. Cho and K. R. Cho. Nonlinear adaptive control for high angle of attack flight. *KSME Journal*, 9(2):147–155, 1995.

[20] Rick Cory. *Supermaneuverable Perching*. PhD thesis, Massachusetts Institute of Technology, June 2010.

[21] Rick Cory and Russ Tedrake. On the controllability of agile fixed-wing flight. In *Proceedings of the 2007 Symposium on Flying Insects and Robots (FIR)*, August 2007.

[22] Rick Cory and Russ Tedrake. Experiments in fixed-wing UAV perching. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, pages 1–12. AIAA, 2008.

[23] Thao Dang, Alexandre Donze, Oded Maler, and Noa Shalev. Sensitive state-space exploration. *IEEE Conference on Decision and Control*, pages 4049–4054, December 2008.

[24] Theodore De Karman and Leslie Howarth. On the statistical theory of isotropic turbulence. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 164(917):192–215, 1938.

[25] Alexis Lussier Desbiens, Alan Asbeck, and Mark Cutkosky. Hybrid aerial and scansorial robotics. *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 72–77, May 2010.

[26] Alexis Lussier Desbiens, Alan T Asbeck, and Mark R Cutkosky. Landing, perching and taking off from vertical surfaces. *International Journal of Robotics Research*, January 2011.

[27] Franklin W Diederich and Joseph A Drischler. *Effect of spanwise variations in gust intensity on the lift due to atmospheric turbulence*. National Advisory Committee for Aeronautics, 1957.

[28] Alexandre Donze and Oded Maler. Systematic simulations using sensitivity analysis. *Hybrid Systems: Computation and Control*, 2007.

[29] C. E. Doyle, J. J. Bird, T. A. Isom, C. J. Johnson, J. C. Kallman, J. A. Simpson, R. J. King, J. J. Abbott, and M. A. Minor. Avian-inspired passive perching mechanism for robotic rotorcraft. *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pages 4975–4980, 2011.

[30] Hugh L Dryden. Turbulence investigations at the national bureau of standards. In *Proceedings of the Fifth International Congress of Applied Mechanics*, pages 362–368, 1938.

[31] Mu Huang; Bin Xian; Chen Diao; Kaiyan Yang; Yu Feng. Adaptive tracking control of underactuated quadrotor unmanned aerial vehicles via backstepping. *American Control Conference*, June 2010.

[32] Adrian Frank, James S. McGrew, Mario Valenti, Daniel Levine, and Jonathhan How. Hover, transition, and level flight control design for a single-propeller indoor airplane. In *AIAA Guidance, Navigation and Control Conference and Exhibit*. AIAA, 2007.

[33] Jyotirmay Gadewadikar, Frank L. Lewis, Kamesh Subbarao, Kemao Peng, and Ben M. Chen. H-infinity static output-feedback control for rotorcraft. *Journal of Intelligent Robotic Systems*, 2009.

[34] Philip E. Gill, Walter Murray, and Michael A. Saunders. *User's Guide for SNOPT Version 7: Software for Large -Scale Nonlinear Programming*, February 12 2006.

[35] Patrick R. Green and Peter Cheng. Variation in kinematics and dynamics of the landing flights of pigeons on a novel perch. *Journal of Experimental Biology*, 201:33093316, 1998.

[36] William E. Green and Paul Y. Oh. A fixed-wing aircraft for hovering in caves, tunnels, and buildings. June 2006.

[37] Frederic M Hoblit. *Gust loads on aircraft: concepts and applications*. American Institute of Aeronautics and Astronautics, 1988.

[38] Warren Hoburg and Russ Tedrake. System identification of post stall aerodynamics for UAV perching. In *Proceedings of the AIAA Infotech@Aerospace Conference*, Seattle, WA, April 2009. AIAA.

[39] W. E. Holley and A. E. Bryson Jr. Wind modeling and lateral control for automatic landing. *AIAA Guidance and Control Conference*, August 1975.

[40] John C. Houbolt, Roy Steiner, and Kermit G. Pratt. Dynamic response of airplanes to atmospheric turbulence including flight data on input and response. *NASA Technical Report TR R-199*, June 1964.

[41] Chien Y. Huang and Gareth J. Knowles. Application of nonlinear control strategies to aircraft at high angle of attack. *Proceedings of the 28th Conference on Decision and Control*, December 1990.

[42] S. Jung and J.T. Wen. Nonlinear model predictive control for the swing-up of a rotary inverted pendulum. *ASME J. on Dynamics, Measurements, and Control*, 126:666–673, September 2004.

[43] Andrey Nikolaevich Kolmogorov. The local structure of turbulence in incompressible viscous fluid for very large reynolds numbers. In *Dokl. Akad. Nauk SSSR*, volume 30, pages 299–303, 1941.

[44] Mirko Kovac, Jurg Germann, Christoph Hurzeler, Roland Y. Siegwart, and Dario Floreano. A perching mechanism for micro aerial vehicles. *Journal of Mirco-Nano Mechatronics*, 5(3-4):77–91, January 2010.

[45] Miroslav Krstic and Petar V. Kokotovic. Control lyapunov functions for adaptive nonlinear stabilization. *System and Control Letters*, 26, December 1995.

[46] Jack W. Langelaan. Biologically inspired flight techniques for small and micro unmanned aerial vehicles. *Guidance, Navigation and Control Conference*, August 2008.

[47] Jack W. Langelaan. Tree-based trajectory planning to exploit atmospheric energy. *American Control Conference*, June 2008.

[48] Jack W. Langelaan, Nicholas Alley, and James Neidhoefer. Wind field estimation for small unmanned aerial vehicles. *AIAA Guidance, Navigation and Control Conference*, 2010.

[49] Jack W. Langelaan and Gotz Bramesfeld. Gust energy extraction for mini- and micro- uninhabited aerial vehicles. In *Proceedings of the 46th Aerosciences Conference*, page 15, 2008.

[50] Jack W. Langelaan, John Spletzer, Corey Montella, and Joachim Grenestedt. Wind field estimation for autonomous dynamic soaring. 2012 IEEE International Conference on Robotics and Automation, May 2012.

[51] Nicholas R. J. Lawrance and Salah Sukkarieh. Autonomous exploration of a wind field with a gliding aircraft. *Journal of Guidance, Control, and Dynamics*, 34(3), May 2011.

[52] Nicholas R.J. Lawrance and Salah Sukkarieh. A guidance and control strategy for dynamic soaring with a gliding uav. *2009 IEEE International Conference on Robotics and Automation*, 2009.

[53] H. W. Liepmann. Extension of the statistical approach to buffeting and gust response of wings of finite span. *Journal of the Aeronautical Sciences*, March 1955.

[54] H.W. Liepmann. On the application of statistical concepts to the buffeting problem. *Journal of the Aeronautical Sciences*, 19(12), December 1952.

[55] Lizarralde, F.; Wen, and J.T.; Liu Hsu. A new model predictive control strategy for affine nonlinear control systems. In *Proceedings of the 1999 American Control Conference*, volume 6, pages 4263–4267, San Diego, California, June 1999. IEEE.

[56] W. Lohmiller and J.J.E. Slotine. Control system design for mechanical systems using contraction theory. *IEEE Transactions on Automatic Control*, 45(5):984 – 989, May 2000.

[57] Lev Gerasimovich Loitsianskii. Some basic laws of isotropic turbulent flow, 1945.

[58] Jennifer M. Lukens, Gregory W. Reich, and Brian Sanders. Wing mechanism design and analysis for a perching micro air vehicle. In *Proceedings of the 49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. AIAA, April 2008.

[59] Anirudha Majumdar, Amir Ali Ahmadi, and Russ Tedrake. Control design along trajectories with sums of squares programming. *arXiv preprint arXiv:1210.0888*, 2012.

[60] Anirudha Majumdar, Amir Ali Ahmadi, and Russ Tedrake. Control and verification of high-dimensional systems with DSOS and SDSOS programming. In *Under Review*, 2014.

[61] Anirudha Majumdar, Mark Tobenkin, and Russ Tedrake. Algebraic verification for parameterized motion planning libraries. In *Proceedings of the 2012 American Control Conference (ACC)*, page 8, Montreal, CA, June 2012.

[62] Adnan Martini, Franois Lonard, and Gabriel Abba. Dynamic modelling and stability analysis of model-scale helicopters under wind gust. *Journal of Intelligent Robotic Systems*, 2009.

[63] J. M. McDonough. Introductory lectures on turbulence. *Departments of Mechanical Engineering and Mathematics, University of Kentucky*, 2007.

[64] D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2520–2525. IEEE, 2011.

[65] D. Mellinger, N. Michael, and V. Kumar. Trajectory generation and control for precise aggressive maneuvers with quadrotors. In *Proceedings of the 12th International Symposium on Experimental Robotics (ISER 2010)*, 2010.

[66] Daniel Mellinger, Nathan Michael, Michael Shomin, and Vijay Kumar. Recent advances in quadrotor capabilities. *2011 IEEE International Conference on Robotics and Automation*, May 2011.

[67] Daniel Mellinger, Michael Shomin, and Vijay Kumar. Control of quadrotors for robust perching and landing. *International Powered Lift Conference*, October 5-7 2010.

[68] Darryl E. Metzger and J. Karl Hedrick. Optimal flight paths for soaring flight. *Journal of Aircraft*, 12(11), 1975.

[69] I.M. Mitchell, A.M. Bayen, and C.J. Tomlin. A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on Automatic Control*, 50(7):947–957, July 2005.

[70] Joseph Moore. Powerline perching with a fixed-wing UAV. Master's thesis, Massachusetts Institute of Technology, May 2011.

[71] Joseph Moore and Russ Tedrake. Powerline perching with a fixed-wing UAV. In *Proceedings of the AIAA Infotech@Aerospace Conference*, Seattle, WA, April 2009. AIAA.

[72] Joseph Moore and Russ Tedrake. Control synthesis and verification for a perching UAV using LQR-trees. In *Proceedings of the IEEE Conference on Decision and Control*, page 8, Maui, Hawaii, December 2012.

[73] Laurence H. Mutuel and Randal K. Douglas. Controller design for low speed flight in turbulence. *American Institute of Aeronautics and Astronautics*, January 1997.

[74] Michiel J. Van Nieuwstadt and Richard M. Murray. Real-time trajectory generation for differentially flat systems. *International Journal of Robust and Nonlinear Control*, 8:995–1020, 1998.

[75] A.V. Oppenheim, A.S. Willsky, and S. Hamid. *Signals and Systems*. Prentice Hall, 2nd edition, August 1996.

[76] A. Papachristodoulou and S. Prajna. Robust stability analysis of nonlinear hybrid systems. *Automatic Control, IEEE Transactions on*, 54(5):1035 –1041, may 2009.

[77] Pablo A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Institute of Technology, May 18 2000.

[78] Chinmay K. Patel and Ilan Kroo. Control law design for improving UAV performance using wind turbulence. In *Proceedings of the 44th AIAA Aerospace Sciences Meeting and Exhibit*. AIAA, January 2006.

[79] Alan Poole. The birds of north america online. *Cornell Laboratory of Ornithology, Ithaca, NY.*, 2005.

[80] Prajna, S., Parrilo, P.A., Rantzer, and A. Nonlinear control synthesis by convex optimization. *IEEE Transactions on Automatic Control*, 49(2):310 – 314, feb. 2004.

[81] S. Prajna, A. Jadbabaie, and GJ Pappas. Stochastic safety verification using barrier certificates. *43rd IEEE Conference on Decision and Control*, pages 929–934, 2004.

[82] S. Prajna, A. Jadbabaie, and G.J. Pappas. A framework for worst-case and stochastic safety verification using barrier certificates. *IEEE Transactions on Automatic Control*, 52(8):1415 –1428, Aug 2007.

[83] A.L. Prickett and C.J. Parkes. Flight testing of the f/a-18e/f automatic carrier landing system. *Aerospace Conference, 2001, IEEE Proceedings.*, 5:2593–2612 vol.5, 2001.

[84] G Reich, O Wojnar, and Roberto Albertani. Aerodynamic performance of a notional perching mav design. In *47 th AIAA Aerospace Sciences Meeting*, 2009.

[85] Reynolds and Osborne. On the dynamical theory of incompressible viscous fluids and the determination of the criterion. *Philosophical Transactions of the Royal Society of London.*, 186:123–164.

[86] John W. Roberts, Rick Cory, and Russ Tedrake. On the controllability of fixed-wing perching. In *Proceedings of the American Control Conference (ACC)*, 2009.

[87] Ian Charles Rust. Control of a nonlinear underactuated system with adaptation, numerical stability verification, and the use of the LQR-Trees algorithm. *S.B. Thesis*, May 2010.

[88] Santhakumar, M., Asokan, and T. Non-linear adaptive control system for an underactuated autonomous underwater vehicle using dynamic state feedback. *International Journal of Recent Trends in Engineering*, 2(5), November 2009.

[89] P. Seiler and G.J. Balas. Quasiconvex sum-of-squares programming. *Proceedings of the IEEE Conference on Decision and Control*, pages 3337 – 3342, 2010.

[90] S. Snell, Dale Enns, and William Garrard. Nonlinear control of a supermaneuverable aircraft. *Proceeings of the AIAA Guidance, Navigation, and Control*, 1989.

[91] S. Antony Snell, Dale F. Enns, and William L. Garrard Jr. Nonlinear inversion flight control for a supermaneuverable aircraft. *Journal of Guidance, Control, and Dynamics*, 1992.

[92] E. Sontag. Feedback stabilization of nonlinear systems, 1989.

[93] Jacob Steinhardt and Russ Tedrake. Finite-time regional verification of stochastic nonlinear systems. In *Proceedings of Robotics: Science and Systems (RSS) 2011*, January 17 2011.

[94] W. Tan and A. Packard. Searching for control lyapunov functions using sums of squares programming. *Allerton conference on communication, control and computing*, pages 210–219, 2004.

[95] J. Tangler and J. David Kocurek. Wind turbine post-stall airfoil performance characteristics guidelines for blade-element momentum methods. In *43rd AIAA Aerospace Sciences Meeting and Exhibit*. AIAA, 2005.

[96] Geoffrey Ingram Taylor. Statistical theory of turbulence. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 151(873):421–444, 1935.

[97] GI Taylor. Statistical theory of turbulence. ii. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 151(873):444–454, 1935.

[98] Russ Tedrake. *Underactuated Robotics: Learning, Planning, and Control for Efficient and Agile Machines: Course Notes for MIT 6.832*. Working draft edition, 2012.

[99] Russ Tedrake, Ian R. Manchester, Mark M. Tobenkin, and John W. Roberts. LQR-Trees: Feedback motion planning via sums of squares verification. *International Journal of Robotics Research*, 29:1038–1052, July 2010.

[100] Ajay Thukral and Mario Innocenti. A sliding mode missile pitch autopilot synthesis for high angle of attack maneuvering. *IEEE Transactions on Control Systems Technology*, 6(3):359–371, 1998.

[101] Bernd Tibken and Kamil F. Dilaver. Robust stability of nonlinear polynomial systems by lmi. *International Conference on Control Applications*, September 2004.

[102] Mark M. Tobenkin, Ian R. Manchester, and Russ Tedrake. Invariant funnels around trajectories using sum-of-squares programming. *Proceedings of the 18th IFAC World Congress, extended version available online: arXiv:1010.3013 [math.DS]*, 2011.

[103] Claire J. Tomlin, Ian M. Mitchell, Alexandre M. Bayen, and Meeko K. M. Oishi. Computational techniques for the verification and control of hybrid systems. In *Multidisciplinary Methods for Analysis Optimization and Control of Complex Systems*, Mathematics in Industry, pages 151–175. Springer Berlin Heidelberg, 2005.

[104] U. Topcu, A.K. Packard, P. Seiler, and G.J. Balas. Robust region-of-attraction estimation. *IEEE Transactions on Automatic Control*, 55(1):137 –142, Jan 2010.

[105] Ufuk Topcu and Andrew Packard. Local robust performance analysis for nonlinear dynamical systems. *2009 American Control Conference*, June 2009.

[106] Ufuk Topcu and Andrew Packard. Local stability analysis for uncertain nonlinear systems. *IEEE Transactions on Automatic Control*, 54(5):1042–1047, May 2009.

[107] Oskar von Stryk. Users guide for dircol: A direct collocation method for the numerical solution of optimal control problems, November 1999.

[108] John M. Wharington. Heuristic control of dynamic soaring. *5th Asian Control Conference*, 2005.

[109] Adam M. Wickenheiser and Ephrahim Garcia. Longitudinal dynamics of a perching aircraft. *Journal of Aircraft*, 43(5):1386–1392, 2006.

[110] Adam M. Wickenheiser and Ephrahim Garcia. Optimization of perching maneuvers through vehicle morphing. *Journal of Guidance, Control, and Dynamics*, 31(4):815–824, July-August 2008.

[111] Fen Wu and Stephen Prajna. A new solution approach to polynomial lpv system analysis and synthesis. *Proceeding of the 2004 American Control Conference*, June 2004.

[112] Yiyuan J. Zhao. Optimal patterns of glider dynamic soaring. *Optimal Control Applications and Methods*, 25:67–89, 2004.