Supermaneuverable Perching

by

Rick E. Cory

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2010

Author .....
Department of Electrical Engineering and Computer Science
May 21, 2010

Certified by .....
Russell L. Tedrake
Associate Professor, Department of Electrical Engineering
and Computer Science
Thesis Supervisor

Accepted by .....
Terry P. Orlando
Chairman, Department Committee on Graduate Theses

Supermaneuverable Perching

by

Rick E. Cory

Submitted to the Department of Electrical Engineering and Computer Science
on May 21, 2010, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Birds have the impressive ability to gracefully 'swim' through the air while executing aerobatic maneuvers that routinely defy modern aeronautical and control engineering, consistently reminding us that the skies are truly their playground. These animals are masters at inducing and exploiting post-stall aerodynamics to quickly execute maneuvers with unprecedented precision, with nowhere near the sustained propulsive power found in modern state-of-the-art aircraft. This amazing ability to manipulate the air is commonly attributed to the intricate morphology of the wings, tail, feathers and overall sensory motor system of the animal.

In this thesis we demonstrate, on real hardware, that using only an approximate model of the post-stall aerodynamics in combination with principled and novel tools in optimal control, even a simple fixed-wing foam glider (no propeller) made out of rigid flat plates, with a single actuator at the tail, is capable of executing a highly dynamic bird-like perching maneuver to land on a power-line by exploiting pressure drag on its stalled wings and tail. We present a feedback controller capable of stabilizing the maneuver over a wide range of flight speeds and quantify its robustness to wind-gust disturbances. In order to better characterize the aerodynamics during perching, we performed smoke-visualization in a low-speed free-flight wind-tunnel, where we were able to capture real images of the dominant vortex wake dynamics. We describe the application of these results to the synthesis of higher fidelity aerodynamic models. We also demonstrate our initial perching experiments with flapping-wings, using a flapping-wing version of our glider as well as our fully computerized two-meter wingspan robotic bird, Phoenix.

Thesis Supervisor:

Russell L. Tedrake, Associate Professor

Thesis Committee:

Leslie Kaelbling
Professor, Deptartment of Electrical Engineering and Computer Science, MIT

David Lentink
Assistant Professor, Deptartment of Experimental Zoology, Wageningen University

Acknowledgments

First and foremost, I'd like to express my deep and heartfelt gratitude to my research advisor Russ Tedrake, who has been an incredible source of inspiration, support, knowledge, and excitement from day one. The moment he suggested a research project on robotic bird flight I knew I was in for an incredible ride. And now, after almost five years, I have to say that this ride has been nothing short of amazing. I seriously doubt I could have found another advisor who would have been willing to let a computer science student work on robot planes and birds. But most importantly, I doubt I could have found another advisor who would be willing to devote so much time, care, and genuine concern for his students, not only as a mentor, but as a friend. I could write an endless list describing everything I've learned from him, and aside from all the technical chops he instilled in me and allowing me the freedom to explore, which I owe him a debt of gratitude for, two of the most important philosophies I will take with me is to never be afraid to dream big and always push yourself to learn more. Thank You.

I would also like to extend a humble thank you to my thesis committee advisor David Lentink, whom from the first time I met has shared a tremendous enthusiasm for the project and has always made himself readily available to discuss all my rookie aerodynamics questions. From the moment we met he has always shown a genuine interested in wanting to help develop my skills as a serious researcher. I'd also like to thank my thesis committee advisor Leslie Kaelbling, for her willingness to help and whose constructive suggestions have directly helped shape this thesis.

There are so many staff and students who directly contributed endless hours to the work presented in this thesis. Ron Wiken personally dedicated hours upon hours to helping me build a free-flight wind tunnel for flow-visualization experiments and I owe him a debt of gratitude for that. The dedication Ron has for CSAIL and its students is exemplary, and fortunately I was one of the lucky ones who got to experience that dedication time and time again. There have been a countless number of occasions where he personally set aside time specifically to help me solve a problem, whether it was finding the right screw, teaching me how to do woodwork, or drive me around town picking up building materials, all the meanwhile having a big smile on his face. He was a tremendous resource for a lot of the work presented in this thesis. Jason Dorfman was responsible for taking the beautiful flow visualization and robotic bird flight pictures contained in this thesis. I big thank you goes to him for always willing to lend a hand, even at very short notice. I'd also like to express my gratitude to Steve Proulx, who on countless occasions lend a hand to the project and was always willing to help.

During my time as a graduate student I have had the privilege of working with a number of UROP students, whose hard work and dedication directly contributed to the results of this project. Derrick Tan was the first UROP to join the project, lending his expertise in RC airplane aerobatics as well as helping us design the first working ornithopter with Steve Proulx. Andrew Meyer dedicated his time and efforts in a number of areas, ranging from airplane simulations to designing on board cameras for flow-visualization. Zack Jackowski played a huge part in the project, designing and building our first autonomous two-meter wingspan robotic bird, Pheonix, which will serve as a platform for our future work on flapping-wing

perching outdoors. Samvaran Sharma also contributed endless hours to the flow-visualization experiments and has always shown a tremendous dedication to pushing the boundaries of the research.

I'd like to send a special thanks the other graduate students who have been involved in the project at one time or another. Joseph Moore directly worked with me in helping design the communication software for running the real perching experiments. Woody Hoburg and John Roberts both played a huge part in helping everyone think about the problem and always asking the right questions - and their contributions lead to some key results in this thesis. I'd also like to thank everyone in the Robot Locomotion group for providing a stimulating research environment and just being great friends to hang out with.

Finally I'd like to thank my family, without whose constant support none of this would have been possible.

Contents

List of Figures

Chapter 1

Introduction

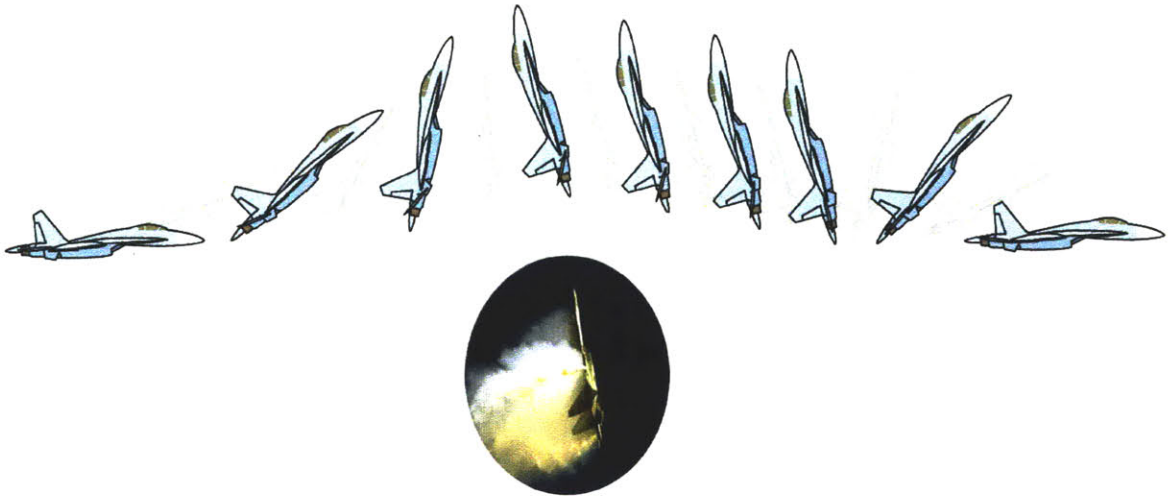


Figure 1-1: Pugachev's Cobra maneuver [Wikipedia, 2009]. Similar to a perching maneuver, the pilot induces a high acceleration pitch up transition to a very high angle of attack and then allows the passive stability of the aircraft to pitch the nose back down for recovery.

1.1 Supermaneuverability

Supermenauverability is a term that describes the capability of an aircraft to operate beyond the conventional flight-envelope limits in a controlled way. Although no strict guidelines exist for categorizing an aircraft as supermaneuverable, there is a qualitative consensus that these aircraft must exceed a threshold of maneuverability beyond the limit of that afforded by conventional aerodynamic control, i.e., the control capabilities afforded by standard control surfaces such as elevator, rudder, etc. This notion of supermaneuverability requires us to assume a qualitative threshold limiting the control capabilities of standard aerodynamic

Figure 1-2: An aircraft carrier landing requires extreme precision and is executed at extremely conservative angles-of-attack to avoid airflow separation, unlike an avian landing maneuver.

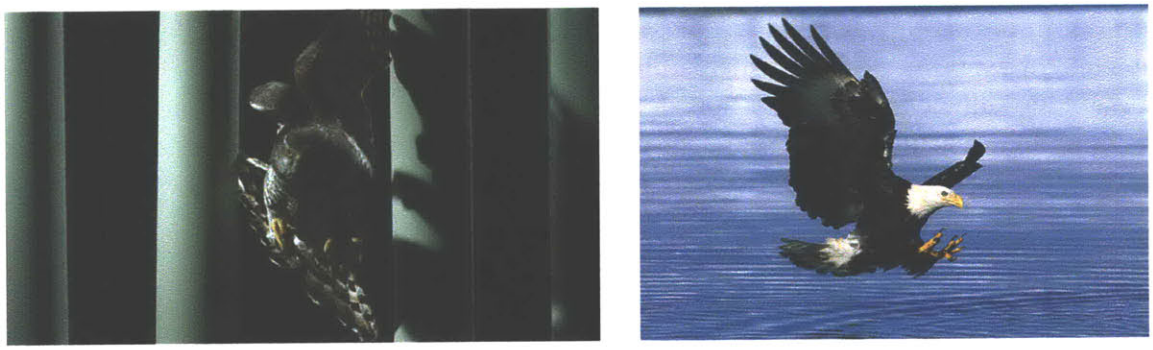
control surfaces. However, given the field's limited understanding of control surface influence at high angles of attack (where unsteady aerodynamics dominate), this classification becomes somewhat nebulous.

Practically, the ability to execute well-known aerobatic maneuvers help categorize an aircraft as supermaneuverable. For example, Pugachev's Cobra, shown in Figure 1-1, is a maneuver that requires a dynamic transition to an extreme post-stall configuration. This causes airflow separation all along the aircraft and, similar to a perching maneuver, causes the aircraft to quickly decelerate, all while maintaining an extremely high angle of attack configuration. These types of impressive aerobatics take the aircraft well beyond normal operating conditions and like the Pugachev's Cobra, are typically characterized by dynamic post-stall flight.

1.2 The Accuracy Challenge

Although there doesn't yet exist a quantifiable metric (and thus an upper limit) for the supermaneuverability of an aircraft, we can point to examples within the operating envelope of our state-of-the-art aircraft that support the existence of a trade-off between executing maneuvers that are highly dynamic, like the Pugachev's Cobra, vs. executing maneuvers that are highly accurate, such as landing on an aircraft carrier. During a carrier landing, a jet will maintain a fairly conservative 8.1° angle of attack approach [Prickett and Parkes, 2001]. This keeps the flight dynamics within the traditional (linear) attached flow regime of the wings and control surfaces. But even keeping the aircraft within this tight and conservative operating envelope, a carrier landing is a daunting task that can easily have fatal consequences [Wilson, 2008]. Moreover, attempting to execute a post-stall maneuver for increased stopping performance would leave the survivability of the pilot completely up to the passive dynamics of the post-stall maneuver. Different than a Pugachev's Cobra which is routinely executed at high altitudes and within sufficiently open airspace for post-stall recovery and with virtually no demands on positional accuracy, high accuracy maneuvers, such as a carrier landing, are much more sensitive to the trajectory of the aircraft.

One reason is that dynamic, high-precision flight maneuvers at high angles of attack,



(a) (b)

Figure 1-3: (a)A Goshawk dynamically weaving through vertical poles morphing its wings to fit within the tight space [Ruben and Svitil, 2007]. (b) A Bald Eagle slows down to pick a fish out of water.

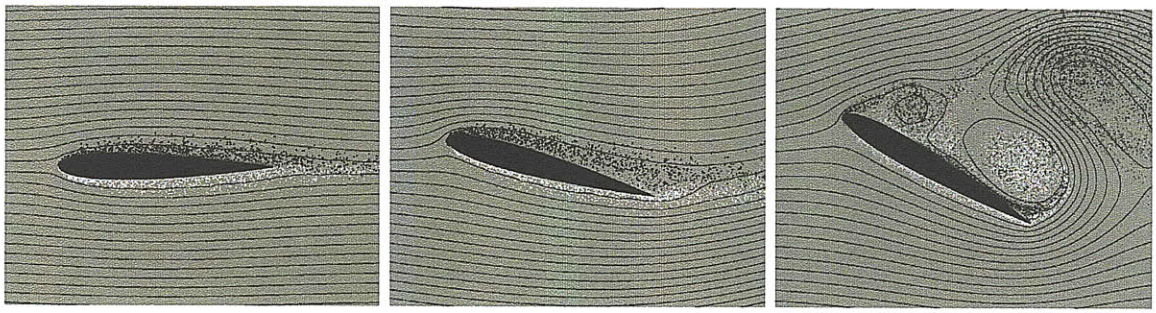
where post-stall aerodydynamics dominate, are not properly addressed by even the most advanced aircraft control systems in the world. Modern supermaneuverable aircraft have the ability to execute maneuvers that can be classified as either highly dynamic or highly accurate, but never both - always trading off one demand for the other. Moreover, these aircraft rely on powerful thrusters that can be energetically expensive to maintain during dynamic maneuvering. Birds, on the other hand, are masters at exploiting post-stall aerodynamics despite low energy budgets and can execute highly dynamic maneuvers with unprecedented accuracy.

1.3 Flying Like a Bird

Birds have the impressive ability to gracefully 'swim' through the air while executing maneuvers that routinely defy modern aeronautical and control engineering, consistently reminding us that the skies are truly their playground. The peregrine falcon can pull out of a 200 mph vertical dive experiencing a force up to 18 times its own weight (18 G's) and is accurate enough to pick a sparrow out of the air in the middle of the maneuver [Tucker, 1998]. Birds are known to dynamically weave through dense forests at high speeds while coming within inches of a fatal crash at fairly alarming rates. They do this by either flaring or tucking in their wings, depending on their need to slow down, speed up, or simply maneuver through tight spaces (Figure 1-3). Eagles, for example, routinely flare their wings at a high angle of attack to quickly slow down and pick a fish right out of the water (Figure 1-3).

Birds can execute these extreme aerobatics with unprecedented precision with nowhere near the sustained propulsive power found in modern aircraft. Although some birds are capable of producing large amounts of thrust compared to their body weight, they can only generate these forces intermittently (i.e. through flapping) and therefore must reason about how to best exploit that capability. However, this does not impede their ability to execute dynamically precise post-stall aerobatics with apparent ease. This carefree execution

of aerial stunts certainly challenges our traditionally conservative approach to autonomous flight control systems design, which unlike their biological counterparts, rely on attached airflow under sufficiently high air speeds. Although nature's flyers by no means represent a limit on agility, learning to fly like a bird does provide a good initial milestone as we move forward with the design of supermaneuverable flying robots [Tedrake et al., 2009].



(a) (b) (c)

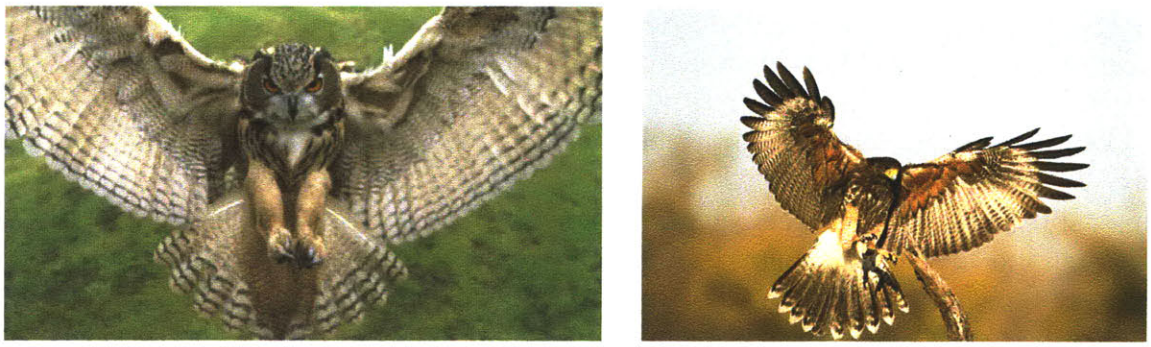
Figure 1-4: This illustrates the flow characteristics of a high angle of attack transition [van Dommelen, 2009]. (a) Zero angle of attack with attached flow over the wing. (b) Moderate angle of attack with attached flow over the wing. (c) High angle of attack stalled wing with separated flow, creating low-pressure pockets behind the wing.

The ability to induce and manipulate post-stall fluid dynamics much like a bird will play a key role in paving the way towards a future where flying robots exhibit a level of control that can rival that of nature's most aerobatic flyers. However, recent trends in state of the art flight control system design (see section 2.1) suggest that supermaneuverability is akin to maintaining a high thrust to weight ratio. If our goal is to match the performance of birds, however, then supermaneuverability should be akin to manipulating and exploiting naturally occurring aerodynamics at low speeds and high angles of attack, without the need to sustain high power propulsion throughout a maneuver.

1.4 Supermaneuverable Perching

Dynamic perching, one of the most common aerobatic maneuvers executed by birds, is representative of a large and important class of aggressive aerial maneuvers that take advantage of unsteady aerodynamics for increased performance. That is, the maneuver exploits post-stall time-varying viscous and pressure drag forces to quickly decelerate the vehicle in an attempt to execute a point landing. Our goal is to compete with birds in metrics of agility, accuracy, and maneuvering power efficiency, i.e., metrics that we believe characterize a large portion of the library of aerobatic maneuvers executed by birds. Successful execution of the maneuver with a glider, the main focus of this thesis, is demanding in terms of all three metrics. This maneuver represents a daunting challenge in controls engineering due to its high degree of underactuation and highly nonlinear aerodynamics at high angles of attack.

During a perching maneuver a bird will flare its wings and tail, mimicking the parachute of a drag racer, and orient its body to a very high angle of attack relative to the direction



(a) (b)

Figure 1-5: (a) A Eurasian Eagle Owl [Turbary, 2010] and (b) a Harris Hawk flaring their wings and tail right before landing on a perch. These birds execute a similarly extreme high angle of attack maneuver in order to quickly stop in a short distance.

of flight (Figure 1-5a). This high angle of attack transition causes the airflow around the wings to separate, i.e. the air fails to smoothly follow the contour of the wing, detaching at the leading edge and creating turbulent, low-pressure pockets of air immediately behind the wings. This effect is not specific to bird wings, but is characteristic of any airfoil at high angles of attack. Figure 1-4 illustrates this separation effect on a standard airfoi in a steady oncoming stream of air. This separation effect induces an incredibly strong pressure-drag, which, in combination with viscous drag forces, effectively creates a powerful set of aerodynamic brakes for the bird. If we were to visualize the airflow around the bird's wings during this maneuver, we would clearly see a dramatic demonstration of post-stall aerodynamics, a regime characterized by turbulence, vortex streets (rotational air currents), and eventual airflow separation right behind the wings as the bird flies in for the point landing (see section 7.1). These flow conditions characterize an aggressive flight envelope that exceeds the limits of any modern autopilot. Our contention, however, is that unlike modern aircraft control systems, birds intentionally induce and manipulate these complicated flow structures to boost their performance and agility; an agility that is as of yet unmatched by even the most advanced autopilots in the world.

In this thesis, I distinguish a supermaneuverable perching maneuver from vertical/static take off and landing (V/STOL) maneuvers by the use of an airflow separation-induced pressure drag to decelerate the vehicle. Executing these dynamic maneuvers allows birds to perch in much less time and with much less energy (scaled) than man-made V/STOL equivalents.

1.5 Contributions and Chapter Organization

The goal of the work presented in this thesis is to design a control system for a flying robot that can compete with birds in metrics of agility, accuracy, and maneuvering power efficiency. Perching, our benchmark task in this thesis, represents a class of aerobatic maneuvers that are demanding in terms of all three of these metrics.

In the next chapter I review the state of the art in both supermaneuverable manned and unmanned aircraft. I highlight their main capabilities and shortcomings and review the most conventional approaches to flight control. I also describe new approaches to the design and control of supermaneuverable flight, and introduce an important control design technique, namely TVLQR, which forms the basis of much of the results presented in this thesis.

In chapter 3, I present a benchmark perching experiment consisting of a real-time data-acquisition and control system architecture interfaced with a motion capture arena. This experimental paradigm has allowed us to study the fundamental longitudinal dynamics and control principles for a supermaneuverable perching maneuver using a conventional fixed-wing glider. Leveraging tools from system identification, I acquire a description of the quasi-steady aerodynamics of a fixed-wing perching glider through data acquisition of real flight trials in our motion capture arena. Through analysis of this flight data, I demonstrate that a fixed-wing glider composed of individual flat plates, as a whole, behaves aerodynamically as a single flat plate, to first order quasi-steady predictions. The flight data used to verify this model captures free-flight aerodynamics through stall and post-stall configurations representative of a dynamic perching maneuver.

I develop an approximate two-dimensional model of our glider based on flat-plate theory quasi-steady aerodynamics and demonstrate on real hardware that by using principled tools in optimal control and a rough approximation of our glider dynamics, even a simple fixed-wing foam glider (no propeller) made out of rigid flat plates, with a single actuator at the tail, is capable of executing a highly dynamic bird-like perching maneuver to land on a perch by exploiting pressure drag on its stalled wings and tail.

Chapter 4 presents the use of novel tools in nonlinear optimal control, namely LQR-Trees, to design a feedback control system in simulation using verified trajectory libraries. The resulting control system shows an improved basin of attraction a factor of four times larger than that of its TVLQR counterpart, with respect to speed disturbances in initial conditions.

In chapter 5 we address the issue of controllability. A measure of the ability of our perching control system to reject wind-gust disturbances is explored and is shown to be greatest at the boundaries of the planned trajectory, in a locally optimal cost-to-go sense. I show that although the instantaneous control authority over pitching moment decreases toward the end of the trajectory (as the speed of the aircraft approaches zero), the sensitivity to wind disturbances also decreases.

In chapter 6, using flat-plate theory aerodynamics, I present a simplified analytical model of a flapping-wing version of our glider. I highlight the capability of increased stopping performance when using flapping-wings, and simulate a simple state-machine open-loop controller that allows our flapping-wing simulation to successfully execute the perching maneuver. Using this analytical model as inspiration, I present the design a flapping-wing version of our flat-plate glider and demonstrate a perching experiment on real hardware that indicates the ability to influence stopping performance using flapping-wings.

I finally conclude with a discussion of our current work on experiments performing smoke-visualization in a low-speed free-flight wind-tunnel, where we were able to capture real images of the dominant vortex wake dynamics for our flat-plate glider. I describe the application of these results to the synthesis of higher fidelity aerodynamic models. I also discuss the

extension of our perching control results to outdoor autonomous perching using a flexible-flapping wing, two-meter wingspan robotic bird, Phoenix.

Chapter 2

The State of the Art

2.1 Automatic Control for Fighter Jets

Arguably, the most advanced flight control systems in the world can be found in today's military combat aircraft. High thrust-to-weight ratios, commonly above 1-to-1 and as high as 1.6-to-1, e.g. on the U.S. F22 raptor [US-Airforce, 2009], combined with multi-axis thrust-vectored propulsion systems provide these jets with an impressive degree of maneuverability. These highly maneuverable jets are capable of executing aggressive aerial maneuvers with controlled sideslip that easily exceed stall limits imposed by conventional aerodynamic control (e.g. using standard control surfaces), a characteristic known as supermaneuverability [Gallaway, 1985, Gal-Or, 1990], a term originally coined and studied by Wolfgang Herbst [Herbst, 1983]. The longitudinal stability margin on these jets, defined as the distance from the center-of-pressure (cp) to the center-of-gravity (cg) of the aircraft, is marginally positive (and even negative in some cases), effectively trading off passive stability for increased maneuverability and control sensitivity. This stability tradeoff, known as relaxed static stability (RSS) [Pamadi, 2004], many times results in an inherently unstable aircraft that is difficult (sometimes impossible) to stabilize manually, i.e. without an automatic control system closing the feedback loop, as the disturbances and period of oscillations may fall well outside the control bandwidth of the average human pilot. The X-31A fighter, for example, is longitudinally unstable with an amplitude doubling time of 200ms [Tischler, 1996]. Therefore, automatic (a.k.a. fly-by-wire) control systems are designed to close the low-level feedback loop; that is, interpret the human pilot's desired speeds and orientations into corresponding thrust and control surface command signals. The ability of the automatic control system to augment the pilot commands to enhance flying quality can vary, depending on the maneuvers, but can be measured using standard criteria [Cooper and Jr., 1969] and must meet certain government specifications [MIL-F-8785C, 1980]. These design specifications and control system abstractions allow the human pilot to concentrate on high-level maneuver execution while offloading low-level control reasoning, e.g. critical damping response or non-minimum phase corrections, to the fly-by-wire computer.

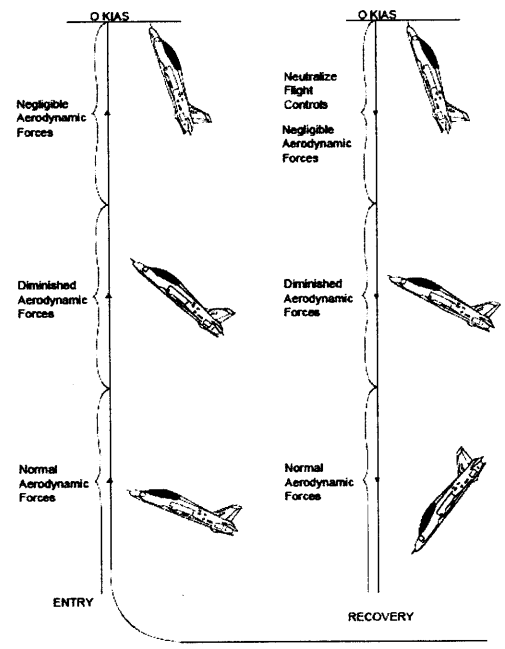
2.1.1 Aerobatic Abilities

Modern fighter pilots are capable of performing extreme aerobatic post-stall combat maneuvers that easily test the limits of their flight control systems. These maneuvers commonly result in a departure of controlled flight where the aircraft's control surfaces have little, if any, effect on the dynamic response of the aircraft due to low airspeeds and separated airflow at high angles of attack, an effect that is alleviated in thrust-vectored jets. The human pilot, in combination with the automatic control system, must go through a set of systematic procedures during a departure recovery [Command, 1998]. Systematic procedures also exist for departure entry, where the pilot intentionally induces a post-stall maneuver, e.g. during air combat. One such maneuver that can be systematically executed (with appropriate thrust-to-weight ratio) is the Pugachev's Cobra (Figure 1-1), which is one of the most demanding and dramatic aerobatic maneuvers performed at airshows around the world. During execution, the pilot induces a pitch-up orientation well beyond 90° while maintaining altitude and dramatically decreasing airspeed, very similar to a perching maneuver. While the aircraft is nosed up (still traveling forward at an extremely high angle-of-attack) the pilot neutralizes the control inputs inducing a nose-down moment on the aircraft (due to passive stability). As the vehicle pitches down and decreases its angle of attack, it picks up speed and recovers from fully stalled to fully controlled flight. The procedures for executing these types of departures are commonly found in standard military flight training manuals [Command, 1998], as shown in Figure 2-1 (taken from the U.S. Naval Air Training Manual). In the case of thrust-vectored jets, execution and recovery are further simplified through added attitude control authority at low airspeeds.

2.1.2 The Machinery

Given that these are arguably the most advanced flight control systems found in any man-made aircraft, the natural question to ask is: Can we harness these state of the art design principles for the development of supermaneuverable autonomous flying robots? A review of the (publicly available) literature reveals that automatic control systems for combat aircraft are overwhelmingly derived using classical linear control techniques, for example root locus, Nichols plots, and time and frequency responses [Tischler, 1996, Harris and Black, 1996], through linearization of the estimated aircraft dynamics. Given the lack of analytically tractable unsteady aerodynamic models at high angles of attack, these dynamics are identified through a set of extensive wind-tunnel tests where static and dynamic stability derivatives are estimated for a variety of flight conditions [Cook, 1997, Alcorn et al., 1996]. Additionally, the dynamics of stalls, departures, and spins for these supermaneuverable aircraft are commonly explored using scaled, free-flying models in specially designed wind-tunnels. NASA Langley, for example, has made use of a 20-foot free-flight vertical spin tunnel for stall/spin experiments that help yield higher fidelity models under these extreme flight conditions [Alcorn et al., 1996]. After a suitable model is identified, MIL-F-8785C [MIL-F-8785C, 1980] and other government standards then serve as guidelines and criteria for control law design, e.g. short period mode damping ratio, pilot-induced oscillation handling, and other dynamic response characteristics.

It turns out, however, that the automatic control system is mostly idle during the execu-



Entry Procedures

a. Start at 250-300 KIAS, no lower than 18,000 ft AGL
b. Commence a 4-g pull to vertical
c. Reduce power to idle as airspeed falls to 150 KIAS
d. Neutralize flight controls as they become ineffective

Recovery Procedures

a. Neutralize Controls

Figure 2-1: Entry and recovery human pilot procedures for a low airspeed 70° nose-up departure, similar to the well-known Pugachev's Cobra maneuver (see Figure 1-1). Taken from the T45-C U.S. Naval Air Training Command Manual [Command, 1998].

tion of low-speed, high angle of attack aerobatics, e.g. while executing the Pugachev's Cobra. In fact, during normal operation, high-alpha and G limiters prevent the aircraft from entering such departures, meaning that the human pilot must manually disengage these safety mechanisms before executing these highly aerobatic maneuvers [Tischler, 1996]. The option to enable/disable thrust vectoring also determines the aircraft's allowable flight envelope. If the autopilot tries to execute a maneuver outside the data-base of the automatic control system while the safety mechanisms are engaged (and thrust vectoring disabled), it will simply ignore the command and maintain heading in order to avoid departures [Tischler, 1996]. Successful execution of the Puchagev's Cobra certainly depends on the pilot's skill level, and passive nose down stability at high angles of attack is critically important during recovery for non-thrust vectored jets, and even in the case of thrust-vectored jets, nose down recovery can still pose a significant challenge [Alcorn et al., 1996], even for a skilled pilot.

2.1.3 Shortcomings

The use of thrust vectoring during high angle of attack, low-speed flight can highly augment the controllability of the system, but these control systems still lack the fidelity to execute high-precision maneuvers under these extreme conditions. Modern fighter control engineers

rigorously address the problem of attitude stabilization, i.e. regulating output variables consisting of roll, pitch, yaw, and their derivatives, using piece-wise linear controllers. However, high-precision maneuvers such as dynamic perching require stabilization over the entire set of state variables, including translational positions and velocities. This demanding level of accuracy will require a finer interaction with the surrounding airflow, especially at low air speeds where unsteady (nonlinear) effects tend to dominate. Moreover, our flying robots should exhibit supermaneuverability by means of non-conventional aerodynamic control at reasonable thrust-to-weight ratios, and have control systems (linear or nonlinear) that exploit the unsteady energy in the air instead of trying to cancel it out. Birds, for example, are masters at exploiting unsteady aerodynamics to execute highly precise aerobatic maneuvers by maintaining thrust levels that are no where near those found in fighter jets. Yet, even with these starved power limitations, they can gracefully execute maneuvers that reveal unprecedented accuracy and agility, e.g. quickly diving in to pick a fish out of water or gracefully landing on the branch of a tree.

Perhaps one of the closest analogies to power-efficient flight can be found within the robotic legged locomotion community, where the use of passive walking dynamics have yielded robots with more natural and more efficient gaits, as compared to their full joint-angle controlled counterparts. Walking robots that strictly control the position of every joint, by canceling out the natural passive dynamics, use at least 10 times the energy (scaled) of a typical human, while passive dynamic walkers can achieve a walking efficiency comparable to that of humans [Collins et al., 2005]. In a similar fashion, by exploiting the naturally occurring aerodynamics during flight, instead of canceling them out with powerful thrusters, we hope to achieve low-power supermaneuverability that will lead to more agile and more efficient flying machines.

2.2 Autonomous Aerobatic Control

Over the last decade, there have been a number of impressive demonstrations of autonomous aerobatics on hobby sized helicopters and airplanes. These small platforms have shown the ability to execute a demanding set of aerobatic maneuvers all under computer control. The most notable of these demonstrations have been on small helicopters capable of autonomously executing maneuvers such as aileron rolls and hammerheads [Gavrilets et al., 2001], inverted hovers [Ng et al., 2004], and other aerobatic maneuvers [Abbeel et al., 2007]. Such maneuvers are facilitated by exploiting a common characteristic of small model aircraft, whose moment of inertia scales down much more rapidly than their size. Helicopters, for example, exhibit a moment of inertia that decreases with the fifth power of the scale factor, while the rotor thrust decreases almost proportionately to the mass of the vehicle [Gavrilets, 2003, Mettler, 2002]. These vehicles, like many aerobatic hobby airplanes, are capable of producing an unusually high thrust to weight ratio, sometimes as large as 3-to-1, essentially allowing the aircraft's propulsion system to dominate the unsteady fluid forces at low speeds and high angles of attack. These capabilities have been routinely exploited in recent fixed-wing aerobatic work [Green and Oh, 2006, Cory and Tedrake, 2007, Frank et al., 2007, Blauwe et al., 2007]. However, even with this high maneuvering propulsive power available, these vehicles are still no match when compared to a common bird

when it comes to combining agility with precision at high angles of attack. Moreover, the ability to increase flight endurance will be highly dependent on the ability to exploit (not cancel) ambient aerodynamic energy during maneuvering.

2.3 Autonomous Perching UAVs

During the last few years, the design and control of perching aircraft has received considerable attention from the aerospace and controls communities because of their important consequences in the future of autonomous flying robots. For example, search and rescue vehicles that can easily recharge by having the ability to perch on a power-line in an energetically efficient way can drastically increase flight range and endurance for a vehicle that would otherwise need to return to base to recharge. To this end, there have been a number of research projects that share our goal to design an autonomous perching aircraft.

Perhaps the most closely related, in terms of aerodynamic performance and agility, is the perching aircraft project at Cornell [Wickenheiser, 2008]. In order to expand their flight envelope, they focus their efforts on redesigning the kinematics of a fixed-wing aircraft rather than the control system. Additional degrees of freedom at the wings and tail allow the vehicle to morph into a configuration that facilitates the execution of a dynamic perching maneuver. This is done by rotating the body/fuselage up into a high angle of attack and keeping the wings within the traditional tight flight envelope characterized by smooth laminar airflow. Similarly, the tail is actuated out of the (unsteady) wake of the body in order to ensure smooth laminar flow. Similar work in [Lukens, 2008, Reich et al., 2009] presents a design for a sectional morphing wing aircraft. The work in this thesis, on the other hand, uses a conventional airplane design that can directly exploit the unsteady aerodynamics of the ambient airflow in order to successfully land on a perch.

Vertical take-off and landing (VTOL) maneuvers have become a popular approach for small autonomous vehicles navigating at slow speeds due to the extremely large thrust to weight ratios available on many commercially available hobby aircraft that are used as platforms for these experiments [Frank et al., 2007]. Using a powerful propeller attached to the nose of these small airplanes, one can essentially hover an airplane (known as a prop-hang) while using the backwash of the propeller to provide the needed control authority for the control surfaces. Although a simple linear control system can navigate this vehicle and execute a (VTOL) perched landing (using proper sensing), it does so at the great expense of speed and energy efficiency. The work presented here uses a glider with no powerplant, i.e. thrust to weight is 0, and has the ability to execute the perching maneuver a factor of ten times faster than it's VTOL counterpart using only the energy required to adjust its control surface.

Work in [Bayrakar and Feron, 2008] demonstrates a small quadrotor that can land on an inclined pad. In the spirit of their earlier control approaches [Gavrilets et al., 2001], a state machine is used to transition between level flight and the open-loop perching maneuver. The landing pad surface is made of a large velcro sheet that can effectively catch the vehicle in mid flight, without it having to make a complete stop. A similar project at Stanford by Cutkosky and his team uses clawed talons on a conventional airplane to effectively catch a wall during an open loop high angle of attack transition [Lussier-Desbiens and Cutkosky, 2009]. Another

project at AFRL similarly uses 'sticky pads' to catch a wall during the open loop maneuver [Anderson et al., 2009]. Our goal, however, is to close the feedback loop on these open-loop control systems in a way that robustly handles wind gusts and disturbances, regardless of the way they latch on to the perch.

2.4 Classical Flight Control

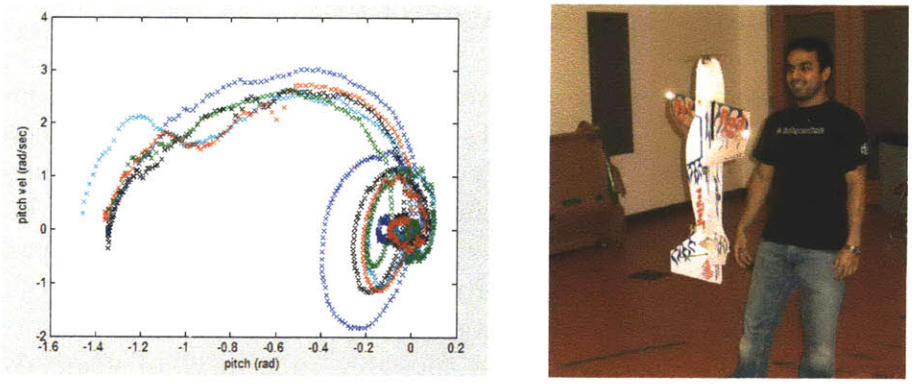
Following the first controlled, manned flight demonstrations by Orville and Wilbur Wright in 1903, the field of aircraft control has seen tremendous progress which has helped pave the way for modern commercial airliners, supermaneuverable combat jets, and space exploration aircraft. Although these achievements have had unquestionable impact on modern transport, scientific research, and world affairs, we have only just begun to scratch the surface of flight control engineering and its applications. Below we highlight the most prominent flight control system architectures to date, some of which form the basis of our work on supermaneuverable perching.

2.4.1 PID Control

By far, the most pervasive approach to UAV flight control system design is the proportional-integral-derivative controller, also known as PID control [Nise, 2003]. This is a linear controller which consists of three gains that multiply position error, velocity error, and an integral over position error. These gains can be calculated through common linear control design methods, e.g. root locus, or many times (for hobby aircraft) through direct hand-tuning if the system model is unavailable. This control approach works well for linear time-inviarant (LTI) flight dynamics, which actually covers a wide range of high-thrust-to-weight ratio hobby aircraft. The large motors attached to these aircraft essentially allow the propellor to dominate all unsteady nonlinear aerodynamic effects at high angles of attack, effectively reducing the dynamical system to a 'flying motor'. Under these operating conditions, one can dramatically violate conventional flight envelopes (e.g. see Figure 2-2), but usually at the cost of efficiency and performance. Moreover, LTI flight controllers will not usually perform outside their operating points, e.g. when thrust does not dominate weight/inertia, as is the case for a dynamic perched landing maneuver.

2.4.2 Linearization and Gain Scheduling

It is quite often the case that although we can accurately model the dynamics of a nonlinear system, the control problem still poses a formidable design challenge, which often results in the control engineer linearizing the system dynamics around a desired trajectory, allowing the application of classical linear control tools. The control problem then becomes that of designing appropriate linear control gains for each of these chosen linearization points, e.g. using root locus, which when interpolated yield a feedback control system that stabilizes the original trajectory. The linearization points are stable equilibriums that correspond to the different set of flight conditions, characterized by flight speed, air pressure, control surface trim, etc. In between these operating points, the control matrices are interpolated according



(a) Samples of BOA

(b) Prop Hang

Figure 2-2: An autonomous prop-hang maneuver using a linear (hand-tuned) PD control. This simple linear controller can stabilize an incredibly large set of initial conditions [Cory and Tedrake, 2007].

to a scheduling variable, which can, for example, correspond to a particular state-variable. During operation, the scheduling variable is measured and the appropriate linear controller is interpolated. A detailed review of this method can be found in [Levine, 1996].

Linearizations can be computed using a first order Taylor series expansion of the nonlinear dynamics about the equilibrium points. Spatial dependencies are not included in the state description of the aircraft, allowing the control engineer to focus specifically on the control of attitude, attitude rates, and other related variables such as side-slip angle.

2.4.3 Feedback Linearization

A more aggressive form of linearization is termed feedback linearization or dynamic model inversion, in which the known (or estimated) system non-linearities are directly compensated for, or fed back, in the control law. Again, spatial dependencies are not included in the dynamics of the aircraft which are assumed to be invariant with respect to position. Feedback linearization differs from gain scheduling in that the linearized controller is valid outside equilibrium flight conditions, so long as the estimated flight dynamics at those flight conditions are well modeled.

A common approach to feedback linearization in aircraft flight control is to separate the dynamics into two time-scales, one involving the fast dynamics (e.g. pitch rate) and the other involving the slow dynamics (e.g side slip) of the aircraft [Snell et al., 1992]. The state variables associated with the fast dynamics are usually the ones directly affected by the available control inputs, e.g. elevator input has a strong influence on pitch rate. The fact that slow state variables have large dependencies on fast state variables allows one to design a control system which exploits this coupling in order to control the variables of interest. This model-order reduction technique is also commonly known as partial feedback linearization, in which only a subset of the state variables can be controlled due to the underactuated nature of the plant.

2.4.4 Linear Quadratic Regulator (LQR)

LQR control theory represents the crown jewel in the field of optimal control due to its mathematical beauty and simplicity, as well as its widespread utility across a spectrum of linear and nonlinear control problems. The theory was pioneered by Rudolf Kalman in the early 1960's [Kalman, 1960], and is commonly referenced as the dual to the well-known Kalman filter. The duality stems from the similarity between the Kalman filter gain and the LQR feedback gain as well as their associated Riccati equations [Simon and Stubeerud, 1970]. In its most general form, the linear quadratic regulator optimizes a quadratic performance index, also known as a cost function, for a (possibly time-varying) linear dynamical system. The elegance of this solution stems from the notion that a control engineer can often summarize the objective of the control system through the optimization of a scalar cost function, which in essence is the underlying theme in the field of optimal control. Moreover, contrary to classical gain scheduling techniques, the linearizations about a trajectory need not be about equilibrium points. In fact, the linearization points themselves may be individually unstabilizable, yet the trajectory itself can be stabilized so long as the linear time-varying (LTV) system meets corresponding controllability conditions over the entire trajectory [Chen, 1998].

For completeness, and to introduce the control principles used in section 3.4.2 and chapter 4, we derive the more general form of this controller here, i.e., the linear-time-varying quadratic regulator.

TVLQR Derivation

Assume we want to design a control policy that minimizes a cost function of the form

$$\begin{aligned} J'(t_0, x_0) &= \phi(x(T), T) + \int_{t_0}^{T} g(x(t), u(t), t)\, dt, && (2.1) \\ \text{s.t.} \quad \dot{x}(t) &= f(x(t), u(t), t), && (2.2) \\ x(0) &= x_0, \end{aligned}$$

where ϕ imposes a final state cost and g imposes an instantaneous cost that is a function of state, input, and time. We can formulate this as an unconstrained optimization problem using the method of Lagrange multipliers to find the minimum of 2.1 given the contraints in 2.2. The augmented objective function then becomes

$$J(t_0, x_0) = \phi(x(T), T) + \int_{t_0}^{T} [g(x(t), u(t), t) + \lambda^T(t)(f(x(t), u(t), t) - \dot{x})]\, dt, \tag{2.3}$$

and is typically written as

$$J(t_0, x_0) = \phi(x(T), T) + \int_{t_0}^{T} [H(x(t), u(t), t) - \lambda^T(t)\dot{x})]\, dt, \tag{2.4}$$

where $\lambda(t)$ represents the Lagrange multiplier at time t and H is the Hamiltonian function defined as

$$H(x(t), u(t), t) = g(x(t), u(t), t) + \lambda^T(t) f(x(t), u(t), t) \tag{2.5}$$

In order to find the minimum we need the differential $dJ = 0$. Using Leibniz's rule for finding derivatives of integrals, this differential can be written (dropping arguments for shorthand notation) as

$$\begin{aligned} dJ &= [\phi_x^T dx + \phi_t dt + (H - \lambda^T \dot{x}) dt]|_T - (H - \lambda^T \dot{x}) dt|_{t_0} \\ &\quad + \int_{t_0}^{T} [H_x^T \delta x + H_u^T \delta u - \lambda^T \delta \dot{x} + (H_\lambda - \dot{x}) \delta \lambda] \, dt, \end{aligned} \tag{2.6}$$

where subscripts represent partial derivatives with respect to those variables, and the vertical bar denotes the expression evaluated at that time. Using integration by parts, 2.6 can be re-written as

$$\begin{aligned} dJ &= (\phi_x - \lambda)^T dx|_T + (\phi_t + H) dt|_T - H dt|_{t_0} + \lambda^T dx|_{t_0} \\ &\quad + \int_{t_0}^{T} [(H_x + \dot{\lambda})^T \delta x + H_u \delta u + (H_\lambda - \dot{x})^T \delta \lambda] \, dt \end{aligned} \tag{2.7}$$

Finding the optima where this differential is equal to zero is the general form of the optimal control problem for a general (even nonlinear) system. Now, if we assume our dynamics are linear and time-varying we can write the system dynamics as

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \tag{2.8}$$

Futhermore, if we define a quadratic cost function of the form

$$\begin{aligned} J(t_0, x_0) &= \frac{1}{2} x(T)^T S(T) x(T) + \frac{1}{2} \int_{t_0}^{T} (x(t)^T Q x(t) + u(t)^T R u(t)) \, dt \\ \text{s.t.} \quad \dot{x(t)} &= f(x(t), u(t), t), \\ x(0) &= x_0, \end{aligned} \tag{2.9}$$

then it turns out we can solve for the optimal feedback control policy that minimizes 2.9. Dropping arguments for notational simplicity, the Hamiltonian from 2.5 is given by

$$H = \frac{1}{2}(x^T Q x + u^T R u) + \lambda^T (Ax + Bu) \tag{2.10}$$

By inspection, in order for 2.7 to be at a stationary point (i.e. $dJ = 0$) we must satisfy

$$\dot{x} = \frac{\partial H}{\partial \lambda} = Ax + Bu, \tag{2.11}$$

$$-\dot{\lambda} = \frac{\partial H}{\partial x} = Qx + A^T\lambda, \tag{2.12}$$

$$0 = \frac{\partial H}{\partial u} = Ru + B^T\lambda. \tag{2.13}$$

From 2.13, the solution for the optimal control satisfies $u(t) = -R^{-1}B(t)^T\lambda(t)$. If we fix the final time, then the other relationship we must satisfy in order for 2.7 to be at a stationary point is

$$\lambda(T) = \frac{\partial \phi}{\partial x}\Big|_T = S(T)x(T) \tag{2.14}$$

Now, in order to find the optimal feedback controller, we need to find $\lambda(t)$. Let us assume that for all times t we satisfy

$$\lambda(t) = S(t)x(t) \tag{2.15}$$

Differentiating 2.15 and substituting in the system dynamics along with 2.12 into the result gives the well known continuous time matrix Riccati equation

$$\dot{S}(t) = -A^T(t)S(t) - S(t)A(t) + S(t)B(t)R^{-1}B^T(t)S(t) - Q \tag{2.16}$$

This equation can be solved backwards in time with final condition $S(T)$ yielding the time-varying linear quadratic regulator (TVLQR) feedback controller

$$u(t) = -R^{-1}B^TS(t)x(t). \tag{2.17}$$

Over the last fifty years, LQR (or the more general TVLQR) control has been the subject of much research and also forms the basis of some of the work presented in this thesis. Although TVQLR requires time-varying linearizations for use on nonlinear systems, new methods that combine tools from motion planning with TVQLR trajectory stabilization are showing promise for nonlinear control of underactuated nonlinear systems [Tedrake, 2009] and is explored in chapter 4.

2.5 Expanding the Flight Envelope

2.5.1 Active Jet Flow Control

A relatively recent technique for manipulating the air at high angles of attack makes use of synthetic jet actuators to tame the unsteady aerodynamic structures present during supermaneuverable flight. The use of jet (and/or suction) actuators can help reduce drag, delay separation and stall, as well as stabilize lift oscillations due to unsteady aerodynamics [el Hak, 2001, Williams et al., 2009, Gursul et al., 2007]. Although the idea of flow control has been around since the early 1900's [Prandtl, 1904], it is only recently with the advent of supermaneuverable aircraft that flow control research has become prominent in field of

aircraft control. Although most practical applications of this approach have been done using open-loop control systems, flow control methods have shown great promise for application on real aircraft. The U.S. X-29 experimental aircraft, for example, successfully implemented vortex flow control and flew 60 successful demonstrations in which air jets were used to manipulate vortices generated at the nose of the aircraft in order to provide added control authority at high angles of attack [NASA, 2009]. Similar work has been reported for UAV control [Amitay and Parekh, 2004]. This enabling technology will certainly continue play a role in the future of supermaneuverable aircraft. In this thesis, however, we focus on biologically inspired flow control through wing actuation, i.e. through use of flapping wings.

2.5.2 Vehicle Morphing Control

With the recent progress of adaptive structures and smart materials and actuators, unconventional wing and body shape changing capabilities, an advantage commonly exploited by birds, have started showing promise for improved aerodynamic performance and efficiency in experimental aircraft. Morphing structures and materials have facilitated the ability to change aspect ratio, wing area, sweep, camber, and overall geometry of the wings and/or body mid-flight [Kosman and Lentink, 2009, Cadogan and Smith, 2004, Ivanco and Scott, 2007, Bowman et al., 2007, de Marmier and Wereley, 2003], and present the possibility of designing aircraft that can choose its kinematic configuration based on the desired flight mode. For example, a long endurance flight mission would be better served through use of minimal sweep, high aspect-ratio wings whereas high speed maneuvering would benefit from a highly swept, low aspect-ratio configuration [Lentink et al., 2007, Kosman and Lentink, 2009]. The focus of our work is to study dynamic morphing for airflow manipulation through use of flapping wings, that is, manipulate the airflow structures through dynamic wing movements.

2.5.3 Machine Learning Control

Modeling the aerodynamics of supermaneuverable flight poses a formidable challenge for control engineering. Although computational fluid dynamics (CFD) methods have matured over the years, their utility in designing feedback control systems for maneuvering flight has been limited due to the complexity, accuracy, and computation time required to produce representative simulation results. Nature's most efficient flyers (and swimmers) routinely exploit unsteady flow structures, such as vortex streets, during maneuvering flight. However, it is unlikely that these aerobatic experts are modeling the full high-dimensional state of the fluid, but instead have acquired and fine-tuned a low-dimensional representation of a control system directly, possibly making use of very crude aerodynamic models to improve their learning performance. Although there is no evidence to suggest this is in fact the case, recent progress in machine learning (a.k.a. approximate optimal control) algorithms have shown the ability to learn a control policy using a compact representation on similarly intractable dynamics [Roberts, 2009, Tedrake, 2004]. In particular, [Roberts, 2009] presents work on a heaving flat plate immersed in water (i.e. a 2D rigid flapping-wing) that was able to learn a control policy that maximizes the horizontal propulsive efficiency in approximately seven minutes, without using a model of the fluid or the flat plate. By comparison, a state

of the art CFD tool will take 36 hours to simulate 30 seconds of flight. Similar machine learning approaches have been successfully used for model-based control policy learning of helicopter aerobatics [Abbeel et al., 2007]. It is highly likely that successful control designs for truly robust and adaptive supermaneuverable perching vehicles will involve elements of both model-free learning and model-based control theoretic approaches.

Chapter 3

Perching with Fixed Wings

In the following sections, I present our results on a simple glider that can land on a perch. We first acquire a surprisingly clean, albeit approximate, model of the aerodynamics up to post-stall by performing system identification on real flight data. We then formulate the perching problem in a way that emphasizes simplicity in design and control, but also rich enough to capture the important characteristic components found in high angle of attack dynamic perching maneuvers. Finally we show that using only an approximate model of the aerodynamics, even this simple glider can exploit pressure drag at high angles of attack to robustly execute the point landing.

3.1 Model Identification for a Perching Glider

3.1.1 Experimental Setup

Our glider design (see Figure 3-1, left) was largely inspired by commercially available aerobatic hobby airplanes. The glider is made of laser-cut 2.8mm thick Dapron foam sheets, carries a GWS four channel micro receiver, an HS-55 Hitech hobby servo, and a Full River 250mah 2-cell lithium polymer battery (see Figure 3-1, right), giving a total combined weight of 77g. For ease of fabrication, the wings are a symmetric foam flat plate (i.e. no camber) with a carbon fiber reinforced leading edge having a 98mm mean chord ($\approx$ 3% thickness-to-chord ratio), a 8:3 aspect ratio, with a maximum lift to drag ratio of approximately 3.5 (at zero elevator angle). The wings are slightly tapered (113mm root chord, 83mm tip chord), and have a 20° dihedral for passive lateral stability. The total surface area of all lifting surfaces (including the fuselage) is 0.1022m^2 for a total wing/surface loading of $0.7534\ \frac{\text{kg}}{\text{m}^2}$. Four 10mm reflective markers placed along the fuselage and four on the elevator control surface allow motion capture reconstruction. The vehicle grabs the perch with a hook mounted just below the center of mass, with a capture window of < 2cm.

The most basic trajectory of a glider landing on a perch can be sufficiently described by the glider's longitudinal dynamics. During this maneuver, these dynamics are mostly controlled by the elevator (controlling pitch) and minor corrections must be made by the ailerons and rudder for deviations outside the longitudinal plane, i.e. roll and yaw. However, given that our glider is unpropelled during flight, the extra drag induced by these control

Figure 3-1: (left) Our experimental glider with a 20° dihedral on the wings for passive lateral stability. Reflective markers are placed along the body and the elevator in order to reconstruct absolute position and orientation using the Vicon motion capture system. (right) Electronics and components: (clockwise from top) hobby servo, motion capture marker, battery, micro receiver. (Quarter shown for scale).

surface deflections creates a non-negligible loss in speed early in the glider's trajectory. A fixed vertical stabilizer and our wing dihedral produce similar lateral corrective forces without the added aerodynamic drag.

We designed an indoor experimental facility which mitigates many of the complications of sensing, computation, and repeatability. Our facility is equipped with a Vicon MX motion capture system consisting of 16 cameras, which use infrared light to track reflective markers on the vehicle. The system provides real-time sub-millimeter tracking of the vehicle and its control surface deflections, as well as the perch location. The basic setup is illustrated in Figure 3-2. Our system has a capture volume of approximately 27 m^3 with the real-time hardware interface communicating new state information at approximately 119 frames per second. To protect the vehicle, we hang a large safety net just above the floor under the entire experiment.

The only sensing equipment onboard the vehicle is a set of small reflective markers attached to the fuselage and elevator. The glider is launched from just outside the capture volume of the Vicon system. Upon entering the volume, the real-time tracking software locks onto the plane and begins relaying position and orientation information over ethernet to the ground control station, a Linux PC running Matlab. The ground control station evaluates the control policy at 45 Hz (every 22 ms) and sends a command by serial port to the buddy-box interface of a hobby radio transmitter. An illustration of the closed feedback loop is shown in Figure 3-3.

This indoor flight environment providing off-board sensing, computation, and control has proven to be an extremely effective experimental paradigm. The protective environment for the vehicle allows us to explore maneuvers well outside the typical flight regime with little risk of damaging the vehicle, and as we report in the following sections, the flight data attained from the motion capture system provides incredibly clean data for system identification.

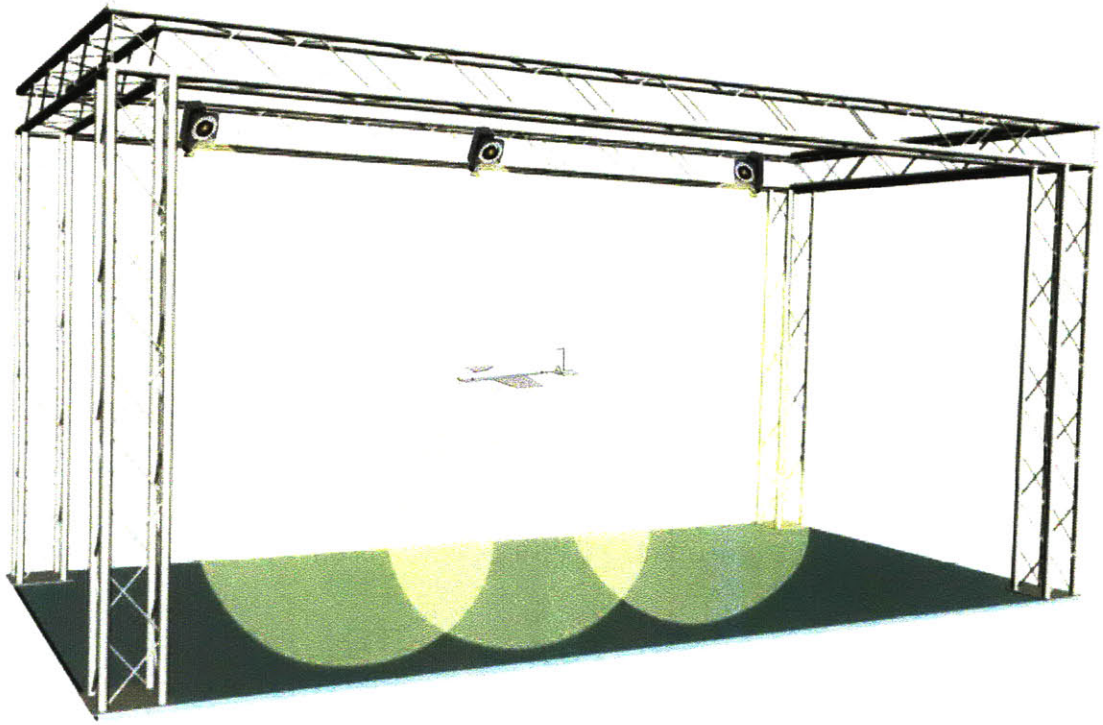


Figure 3-2: Vicon Motion Capture Environment. The Vicon motion capture setup makes use of 16 infrared cameras to track small reflective markers along the body and control surface of the glider. These marker positions allow the system to reconstruct the position and orientation of the glider with high accuracy.

3.1.2 Quasi-Steady Aerodynamic Model Estimation

Regardless of the complexity of the dynamics for a given control problem, it is always to the designer's benefit to extract any and all available information about the structure of the dynamics for use in the control design. This information can include approximate dynamic models, control policy structure based on simulating these models, and/or low-order model representations. Our goal for system identification was to estimate a quasi-steady approximate model of the glider's aerodynamics during a perching maneuver.

Rather than approximate a quasi-steady model in a wind tunnel where representative aerodynamic transients are nearly impossible to duplicate, we obtained our model directly from real flight data. Our primary interest is in the model dynamics in very high angle-of-attack (post-stall) conditions, so we hand-designed a simple feedback policy which brought the nose up and then attempted to bring the nose back down. This was accomplished by defining two deflection planes in the x-axis, the first 2.5m from the perch and the second 1.5m from the perch. Starting with a trimmed elevator angle at launch, the elevator was then deflected up to a specified angle, ψ_1^d, upon crossing the first deflection plane, and down to a different angle, ψ_2^d, upon crossing the second deflection plane. By hand-tuning the parameters, ψ_1^d and ψ_2^d, of this simple controller, we obtained good coverage of the post-

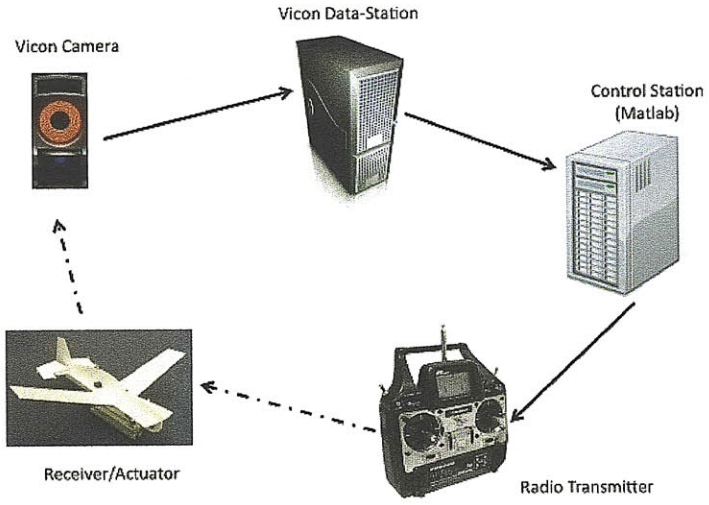


Figure 3-3: Feedback Control Loop.

stall configuration space. These trials typically ended with the plane tumbling towards the ground (e.g., in a falling-leaf mode), to be caught by the safety net.

In total, we recorded 240 flight trajectories for model identification. Data during these trials were logged at the ground station, and included only the elevator commands and the state information from the motion capture system, which provided high-accuracy kinematic information about the trajectory of the plane, including actual elevator deflection, at approximately 119 Hz. The accuracy of each marker estimate is sub-mm, resulting in a sub-mm accuracy in position estimate and < 0.01 rad error in vehicle orientation and elevator angle. We independently estimated the vehicle mass, center of mass position, and moment of inertia (assuming that the mass of the elevator was negligible). The flight data was low-pass filtered (by a 10th order Butterworth filter with a 15Hz cut-off) and differentiated twice. Instantaneous angle-of-attack, $\alpha[n]$, lift and drag forces, $F_L[n]$ and $F_d[n]$, and pitching moments, $M[n]$, were computed directly from the model and estimated accelerations (all at time n). Finally, we estimated the instantaneous coefficients,

$$C_L[n] = \frac{2F_L[n]}{\rho V[n]^2 S} \quad C_D[n] = \frac{2F_D[n]}{\rho V[n]^2 S} \quad C_M[n] = \frac{2M[n]}{\rho V[n]^2 Sc}, \tag{3.1}$$

where C_L, C_D, and C_M represent lift, drag, and moment coefficients, respectively, and c is the mean chord.

The goal of identification here was to generate a predictive model of the flight coefficients that can be used to design the control system. The general trend of the coefficient data is shown in Figure 3-4. Notice that, although data from any individual trial showed time-varying coefficients (potentially) due to the vortex shedding at high angles of attack, the coefficient data averaged over many trials agrees incredibly well with that predicted by flat plate theory [Tangler and Kocurek, 2005], which forms the basis of our simplified flat-plate glider model. In general, the coefficients are a function of the angle of attack of the wings, the

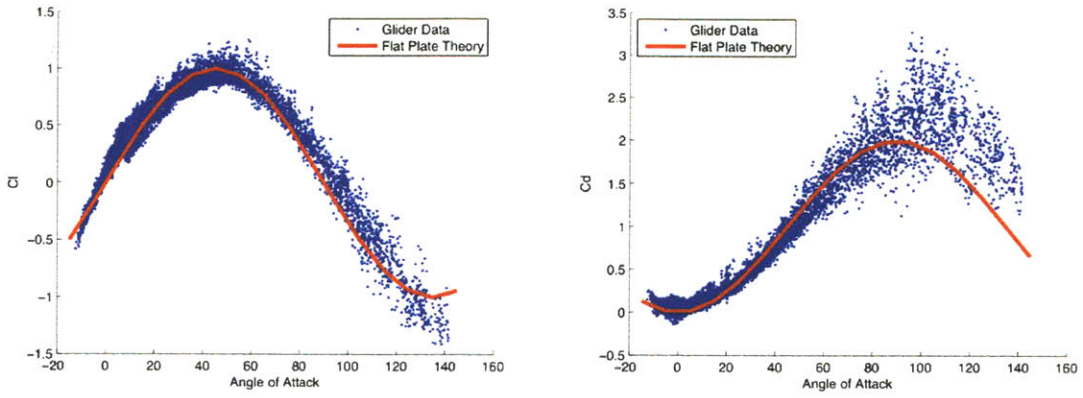


Figure 3-4: Scatter plots showing the general trend of the computed coefficients from flight data. Shown are lift and drag coefficients with their respective flat plate theory predictions.

elevator deflection angle, the Reynolds number, and the Mach number, as well as additional fluid-related terms in the unsteady regime such as pitch rate. In our data, although the Reynolds number (relative to the mean chord) varies characteristically during the trajectory starting around 50,000 at launch and decreasing to around 14,000 at termination, much of the variability can be seen to originate from oscillations, potentially originating from vortex shedding at high angles of attack. A more comprehensive review of our modeling procedures are reported in [Cory, 2008, Cory and Tedrake, 2008].

3.1.3 Flat-Plate Glider Model

Here we present a simplified glider model based on the aerodynamics presented in section 3.1.2. This simple glider model can be described by a single rigid body subject to gravitational and aerodynamic forces. The aerodynamic forces are functions of the angle of attack of both the main wing and the elevator control surface, both of which are modeled as thin flat-plates (see Figure 3-5). As shown in section 3.1.2, thin flat-plate models are in fact a close approximation to the actual wings and control surfaces found on many aerobatic hobby aircraft, whose designs have formed the basis of our experimental platforms. Section 3.1.2 presents the aerodynamic coefficients for one of our flat-plate gliders, which follow closely the coefficients predicted by flat-plate theory. Figure 3-4 illustrates the general trend of the estimated coefficients versus their flat-plate theory predictions. Our data agree closely with their respective theory models, which can be written in closed form as:

$$C_L = 2\sin(\alpha)\cos(\alpha) \tag{3.2}$$

$$C_D = 2\sin^2(\alpha) \tag{3.3}$$

Recent work [Hoburg and Tedrake, 2009] shows that the majority of our glider's aerodynamic behavior can in fact be explained using these same closed form lift and drag coefficient curves.

First, let us define the state of the system as $\mathbf{x} = [x\ y\ \theta\ \phi\ \dot{x}\ \dot{y}\ \dot{\theta}]^T$, and the control input as $\mathbf{u} = \phi_d$. We also define unit vectors normal to the control surfaces (in the directions of

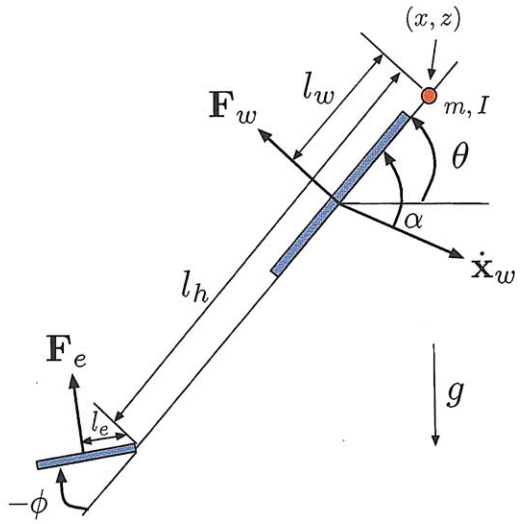


Figure 3-5: Flat plate glider free body diagram.

the illustrated force vectors):

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

where we introduce the notation $s_\theta = \sin\theta$ and $c_\theta = \cos\theta$ which will be used throughout this section. Next, solving for the kinematics of the geometric centroid of aerodynamic surfaces (here equivalent to the the mean aerodynamic chord, which is the center of pressure for flat plate theory) for each aerodynamic surface, we have:

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

$$\dot{\mathbf{x}}_w = \begin{bmatrix} \dot{x} + l_w \dot{\theta} s_\theta \\ \dot{y} - l_w \dot{\theta} c_\theta \end{bmatrix}, \quad \dot{\mathbf{x}}_e = \begin{bmatrix} \dot{x} + l\dot{\theta} s_\theta + l_e(\dot{\theta}+\dot{\phi}) s_{\theta+\phi} \\ \dot{y} - l\dot{\theta} c_\theta - l_e(\dot{\theta}+\dot{\phi}) c_{\theta+\phi} \end{bmatrix}. \tag{3.5}$$

For aerodynamic forces, Equations (3.2) and (3.3) provide expressions for lift and drag coefficients that closely model the observed dynamics. Using these coefficients, the forces on the wing and elevator can be written in closed form. The aerodynamic force acting on an airfoil can be written as components in lift and drag:

$$\mathbf{F}_L = \frac{1}{2}\rho|\dot{\mathbf{x}}_w|^2 C_L S, \quad \mathbf{F}_D = \frac{1}{2}\rho|\dot{\mathbf{x}}_w|^2 C_D S. \tag{3.6}$$

where ρ is the density of air and S is the airfoil surface area. Using C_L and C_D from Eqns.

(3.2) and (3.3), these can be written as:

$$\mathbf{F}_L = \rho|\dot{\mathbf{x}}|^2 \cos(\alpha)\sin(\alpha)S \tag{3.7}$$
$$\mathbf{F}_D = \rho|\dot{\mathbf{x}}|^2 \sin^2(\alpha)S. \tag{3.8}$$

where α represents the airfoil's angle-of-attack. Noting that the lift force is perpendicular to velocity in the positive z direction and the drag force is anti-parallel to the velocity, the total aerodynamic forces on the wing and elevator are given by:

$$\alpha_w = \theta - \tan^{-1}(\dot{z}_w, \dot{x}_w),\ \alpha_e = \theta + \phi - \tan^{-1}(\dot{z}_e, \dot{x}_e) \tag{3.9}$$
$$\mathbf{F}_w = \rho S_w|\dot{\mathbf{x}}_w|^2 \sin\alpha_w \mathbf{n}_w = \rho S_w|\dot{\mathbf{x}}_w|(s_\theta \dot{x}_w - c_\theta \dot{z}_w)\mathbf{n_w} = f_w \mathbf{n_w} \tag{3.10}$$
$$\mathbf{F}_e = \rho S_e|\dot{\mathbf{x}}_e|^2 \sin\alpha_e \mathbf{n}_e = \rho S_e|\dot{\mathbf{x}}_e|(s_{\theta+\phi}\dot{x}_e - c_{\theta+\phi}\dot{z}_e)\mathbf{n_e} = f_e \mathbf{n_e} \tag{3.11}$$

where we use the subscripts w and e to denote wing and elevator, respectively. Finally, the glider dynamics are given by:

$$m\ddot{x} = -f_w s_\theta - f_e s_{\theta+\phi} \tag{3.12}$$
$$m\ddot{z} = f_w c_\theta + f_e c_{\theta+\phi} - mg \tag{3.13}$$
$$I\ddot{\theta} = -f_w l_w - f_e(l c_\phi + l_e) \tag{3.14}$$

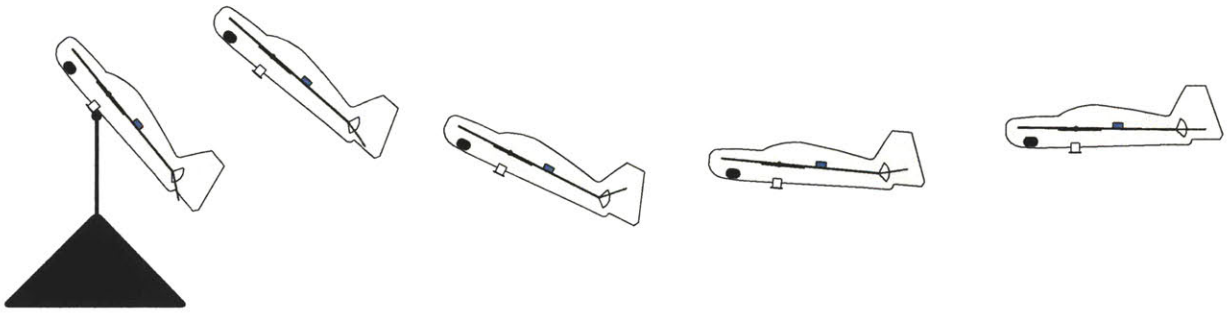


Figure 3-6: Representative perching trajectory in two dimensions.

3.2 Task Formulation

Here we consider the task of making a small autonomously controlled glider land on a laboratory perch. The focus on a glider (no onboard propulsion) helps emphasize the dynamics of the task and avoids control solutions which exploit the large thrust-to-weight ratios available on many small hobby aircraft. These thrust-to-weight ratios are energetically expensive to maintain, and are not necessary for high-performance perching. Specifically, we launch our glider out of a small custom (crossbow) launching device towards a suspended string perch. The vehicle enters our experimental setup at approximately 7 m/s (Re $\approx 50,000$), and about 3.5 m away from the perch. The vehicle must decelerate to almost zero velocity and come to

rest on the perch in less than one second. We 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 surfaces, the task requirements demand a high-precision landing on the string. Figure 3-6 shows a feasible perching trajectory in two dimensions. Note the initial dip in altitude, which compensates for the boost in altitude right before stalling the wings at the end of the trajectory. The cost function that rates the performance of the perching maneuver generally consists of a quadratic penalty on the both the final distance to the perch as well as final speed, as will be shown in the following sections.

The are 4 variables that describe the glider position/angles. These and their derivatives will be used throughout this thesis to describe the state/input of the system:

$$x = \text{horizontal position}$$
$$z = \text{vertical position}$$
$$\theta = \text{pitch}$$
$$\phi = \text{elevator angle}$$

3.3 A Value Iteration Algorithm for Landing on a Perch

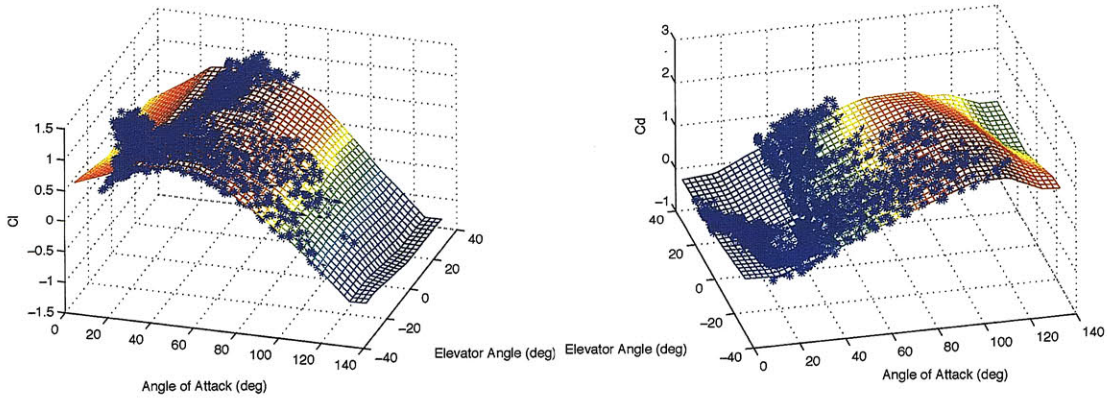


Figure 3-7: A statistical fit of the lift and drag coefficients used for value iteration control design, parameterized by angle of attack and elevator angle. From results presented in [Cory, 2008, Cory and Tedrake, 2008].

Here we describe our initial control design procedure for landing on a perch, which relies on an approximate optimal control value-iteration type algorithm to find a globally optimal feedback policy over the state space of the glider, using a statistical model of the estimated aircraft dynamics. The statistical model of the glider aerodynamics was parameterized as a function of angle of attack and elevator angle, as described in [Cory, 2008, Cory and Tedrake, 2008], also shown in Figure 3-7. Although the glider model presented in section 3.1.3 is a more convenient representation of the dynamics, the flat-plate theory

curves presented there were the result of an analysis that came about after the design and implementation of the control system described here, and were therefore not used in the current control design. Instead, those models formed the basis for design of our later controllers, which are presented in section 3.4.2 and chapter 4.

We formulate the perching problem as an infinite-horizon optimal feedback control problem. Let us define the full-state vector of the system, $\mathbf{x} = [x, z, \theta, \dot{x}, \dot{z}, \dot{\theta}]^T$. The goal of the controller design procedure is to design a full-state feedback policy,

$$u[n] = \pi(\mathbf{x}[n]), \tag{3.15}$$

where u is the desired elevator angle, which is limited by saturation to the (scaled) interval $[-1, 1]$, and $\pi : \Re^6 \rightarrow \Re$. Together with the estimated dynamics, which we summarize with the discrete-time equation,

$$\mathbf{x}[n+1] = f(\mathbf{x}[n], u[n]), \tag{3.16}$$

the feedback policy completely describes the closed-loop dynamics of the system.

In order to steer the vehicle as close as possible to the desired state $\mathbf{x}^d$, i.e., the desired perching configuration, position, and velocity, we define the infinite-horizon cost-to-go,

$$J^\pi(\mathbf{x}) = \min_{n=0,\ldots,\infty} \left[(\mathbf{x}[n] - \mathbf{x}^d)^T \mathbf{Q} (\mathbf{x}[n] - \mathbf{x}^d)\right], \tag{3.17}$$

$$\text{s.t.} \quad \mathbf{x}[0] = \mathbf{x}, \;\; \mathbf{x}[n+1] = f(\mathbf{x}[n], \pi(\mathbf{x}[n])). \tag{3.18}$$

The goal of the numerical optimal control algorithm is to find the feedback policy, π, which minimizes this cost-to-go for all $\mathbf{x}$.

The cost-to-go function described above does not have the standard additive cost structure which is so often exploited in optimal control algorithms[Bertsekas, 2000]. However, a solution to this cost function can be approximated with equally simple algorithms by noting the recurrence relation:

$$J^\pi(\mathbf{x}) = \min \left[(\mathbf{x}[n] - \mathbf{x}^d)^T \mathbf{Q} (\mathbf{x}[n] - \mathbf{x}^d), \; J^\pi(\mathbf{x}')\right], \;\; \mathbf{x}' = f(\mathbf{x}, \pi(\mathbf{x}))). \tag{3.19}$$

Note that the 'min' in this equation returns the minimum value of the two arguments. Similarly, the optimal policy, π^* and optimal cost-to-go, J^*, subscribe to the relation:

$$J^*(\mathbf{x}) = \min \left[(\mathbf{x}[n] - \mathbf{x}^d)^T \mathbf{Q} (\mathbf{x}[n] - \mathbf{x}^d), \; J^*(\mathbf{x}')\right], \;\; \mathbf{x}' = f(\mathbf{x}, \pi^*(\mathbf{x}))). \tag{3.20}$$

By discretizing the state space, this relation provides a recursive algorithm which, for any initial guess $\hat{J}^*(\mathbf{x})$, will converge to the optimal solution $J^*(\mathbf{x})$ of the discretized system. We perform this discretization, and the interpolation back into a continuous full-state policy, using a barycentric function approximator [Munos and Moore, 1998]. In order to be computationally tractable, the discretization in this high-dimensional space must be very coarse; we use 15 bins on x and z, and 8 bins each on θ, $\dot{x}$, $\dot{z}$, and $\dot{\theta}$. We assume that the current state of the elevator, ψ and $\dot{\psi}$ can be directly controlled using on a second order linear actuator model that was estimated using step responses in motion capture.

3.3.1 Coping with Delay

The dynamics of the system are not quite as ideal as described in the control derivation above because of delays in the feedback loop of the experiment. Therefore, our linear actuator model is made to account for the feedback delay in the system. Naively, this delay will increase the number of state variables for our optimal control algorithm. However, we can take advantage of the structure of this pure delay, which happens to be 4 in our discrete-time model, by observing that the delayed dynamics can be written as

$$\mathbf{x}[n+1] = f(\mathbf{x}[n], u[n-4]). \tag{3.21}$$

Therefore, if we solve the optimal control equations above, assuming no delay, we can (through a change of variables) write the optimal feedback policy as

$$u[n] = \pi^*(\mathbf{x}[n+4]). \tag{3.22}$$

We can then execute this policy on the real plane by simulating our state estimate forward in time:

$$\hat{\mathbf{x}}[n+4] = f_4(\mathbf{x}[n], u[n-4], u[n-3], u[n-2], u[n-1]), \tag{3.23}$$

where f_4 is the 4-step dynamics of f.

3.3.2 Results

The success rate of this control scheme, even in simulation, was highly dependent on initial conditions, presumably because of discretization artifacts. Our value iteration control optimization was able to reproduce the perching maneuver in simulation for a limited set of initial conditions. Figure 3-8 shows the translational and pitch trajectories of the glider for a successful simulation run.

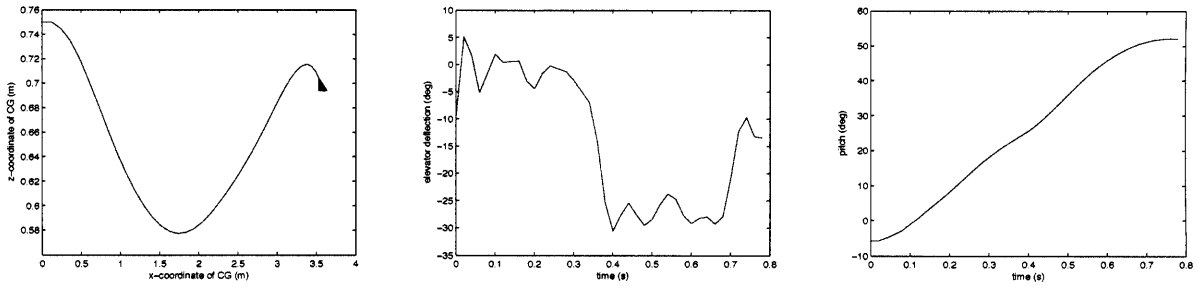


Figure 3-8: (left) The simulated trajectory of the perching maneuver starting at a nominal condition of 6.0m/s 3.5m away from the perch (in red). (middle) The elevator command policy over the perching trajectory. (right) The corresponding simulated pitch trajectory for the perching task.

Figure 3-9 shows frames from a high-speed image sequence from a successful perch, playing out an open loop control trajectory from our optimal control solution. Based on

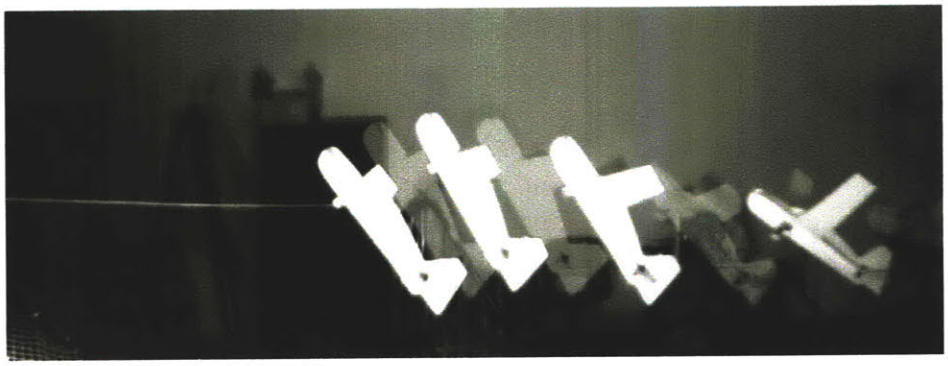


Figure 3-9: Composited stills from a successful perching trajectory using a value iteration control solution.

our initial control design procedure, which used a fairly coarse resolution mesh over the high-dimensional state-space (due to memory constraints), the perching maneuver on the real glider had a relatively small basin of attraction; the launcher put the plane into this basin about 1 in every 5 trials, and only for a few select initial conditions that didn't suffer as much from discretization artifacts (or potential modeling errors).

3.4 Trajectory Design and Local Feedback Control

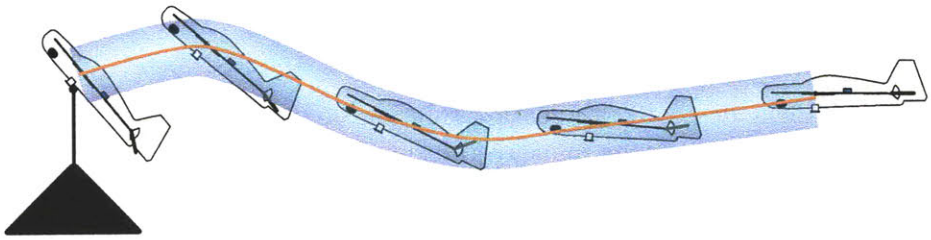


Figure 3-10: A cartoon illustrating the local region of stability around a nominal trajectory.

The full-state feedback non-linear control design presented in section 3.3 is operating at the limits of what one can expect from any type of dynamic programming approach, of which value iteration is a special case, on any high-dimensional state system. Although these algorithms have the capability of yielding a globally optimal full-state feedback solution, it does not scale well beyond 5-6 dimensions (at least for brute-force discretization), and even at these boundaries performance is typically poor. In an effort to circumvent these limitations, we can sacrifice the guarantee of global optimality in favor of a feedback controller that can locally stabilize a pre-computed nominal perching trajectory. Figure 3-10 illustrates our basic goal, that is, we first compute a nominal trajectory (shown in red) using a trajectory

planning algorithm, and then locally stabilize this trajectory by computing a local stabilizing feedback controller. The role of the stabilizing feedback controller is to guarantee that states that are within its basin of attraction (shown in transparent blue) will land on the perch.

In the following sections we take precisely this approach. We first compute a nominal trajectory using convex optimization, and then feedback stabilize this nominal trajectory using a control design based on a time-varying linearization of the dynamics. In the sections that follow, we assume a seven dimensional description of the glider state given by $\mathbf{x} = [x, z, \theta, \phi, \dot{x}, \dot{z}, \dot{\theta}]^T$. The control input is elevator velocity $\dot{\phi}$. All glider variables are relative to the perch coordinate system.

3.4.1 Open-Loop Trajectory Generation

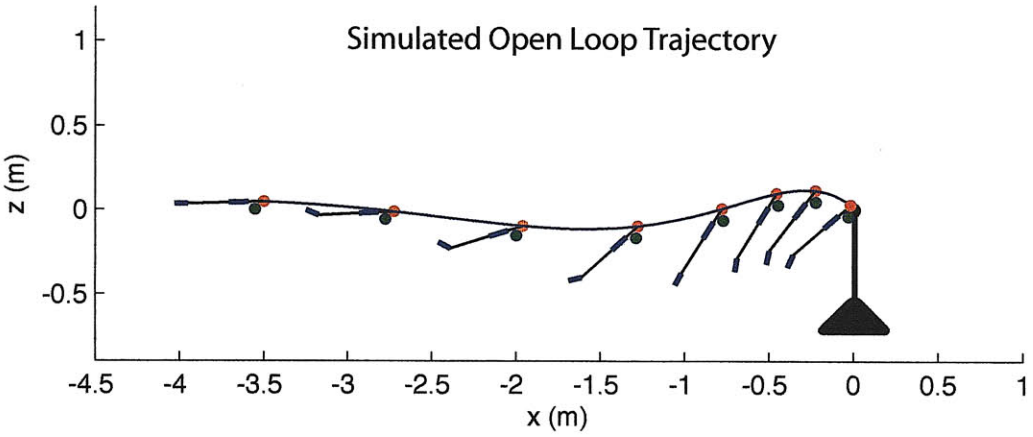


Figure 3-11: A gradient descent optimization solution of the cost in equation 3.24 produces a trajectory which executes the perching maneuver in simulation, as shown here. Shown is the center of mass (red), its corresponding trajectory (solid blue line), and the latching hook (green). This resultant trajectory is the nominal trajectory that will be stabilized in the following section.

We can compute a nominal perching trajectory using standard convex optimization techniques. We first formulate the problem as a finite-horizon open-loop optimal control problem (i.e., the controller is solely a function of time) and the optimization consists of finding the control policy u which minimizes the cost-function

$$J = \sum_{n=0}^{N} \left(u^2[n]R + \mathbf{x}^T[n]Q\mathbf{x}[n] \right) + \mathbf{x}^T[N]Q_f\mathbf{x}[N], \tag{3.24}$$

where $R = R^T > 0$ is cost on actuation, $Q = Q^T \geq 0$ is cost on state error and $Q_f = Q_f^T \geq 0$ is the cost on final state error. For this work all were taken to be diagonal matrices, with R set to $10^{-6} \cdot I$ (where I as the identity matrix), Q set to the zero matrix, while Q_f's was a diagonal matrix set to $\text{diag}(Q_f) = [2000\ 2000\ 100\ 0\ 20\ 10\ 0]$, corresponding to gains of 2000

on x and y, 100 on θ, 20 on $\dot{x}$, and 10 on $\dot{z}$.

The optimization was performed using gradient descent on the cost-to-go function w.r.t. the open loop control inputs $u[n]$, using the flat-plate glider model presented in section 3.1.3, for a single nominal initial condition of $\mathbf{x} = [-3.5, .1, 0, 0, 7.0, 0, 0]^T$. Typically non-convex problems, such as the minimization of 3.24, suffer from local minima when solving using standard convex-optimization tools. However, in our case, variations in initial control parameters resulted in the same open loop control solution. We note that there are a number of ways one can choose to compute a nominal perching trajectory, e.g. using collocation and SQP optimization tools [Roberts et al., 2009], randomized motion planning [LaValle, 2006], or even using human pilot data [Abbeel et al., 2007, Gavrilets, 2003, Frazzoli et al., 2002]. In our case, and for the perching maneuver studied here, standard gradient descent on the cost-to-go function produced satisfactory results. The resulting trajectory (in position and orientations) is shown in Figure 3-11. This nominal trajectory serves as a basis for trajectory stabilization.

3.4.2 TVLQR Trajectory Stabilization

Our goal in designing a local feedback controller is two-fold. First, we want to reject disturbances, due to wind-gusts or other perturbations, as we fly along the trajectory. Second, given that our aerodynamic model is only approximate, there will be a discrepancy between the trajectory that the model predicts using the open loop controller designed in the previous section, and the actual trajectory that will be followed by the real glider, which is of course subject to additional dynamics that we didn't account for, e.g., vortex shedding, dynamic-stall, etc.

In the feedback control design procedure we describe here, we don't explicitly account for model errors. However, if we have reasonable bounds on the model error (and potentially its structure), tools exist that explicitly account for these errors in the control design (e.g. robust control), and are the subject of much research. Our hope here is to understand the limitations of not accounting for these errors explicitly in the control design procedure. In the future, as we obtain better descriptions of the unsteady aerodynamics at play (see section 7.1), we hope to include this information in our control design procedure.

The full non-linear system is converted to a linear time-varying (LTV) system by linearizing the nonlinear dynamics around the nominal trajectory. The behavior of the system along the trajectory can then be written as:

$$\mathbf{x}[n+1] - \mathbf{x}_t[n+1] = A[n](\mathbf{x}[n] - \mathbf{x}_t[n]) + B[n](\mathbf{u}[n] - \mathbf{u}_t[n]), \tag{3.25}$$

where A and B are the linearizations of the dynamics at time t of the trajectory, while $\mathbf{x}_t$ and $\mathbf{u}_t$ are the nominal trajectory state and action vectors, respectively.

A time-varying linear quadratic regulator (TVLQR) controller was computed around the optimized nominal trajectory resulting in a linearized feedback control policy. This was obtained by defining a quadratic cost function along the nominal trajectory with the same structure as 3.24, and then (see section 2.4.4) computing the discrete time Riccati equation

backwards in time according to:

$$S[n-1] = A^T[n]S[n]A[n] - (A^T[n]S[n]B[n])(B^T[n]S[n]B[n]+R)^{-1}(B^T[n]S[n]A[n]) + Q, \tag{3.26}$$

using a large but finite final cost. The matrix S can then be used to derive the optimal TVLQR controller of the form [Kwakernaak and Sivan, 1972]:

$$\begin{aligned} \mathbf{u}[n] &= -(B^T[n]S[n]B[n]+R)^{-1}(B^T[n]S[n]A[n])(\mathbf{x}[n]-\mathbf{x}_t[n]), & (3.27)\\ &= -K[n]\bar{\mathbf{x}}[n] & (3.28) \end{aligned}$$

where $\mathbf{x}_t[n]$ is the nominal trajectory of the system. The terminal value $S[N]$ for this integration is then a final cost equivalent to the Q_f of equation 3.24 . While the cost function structure used to develop the controller was the same as that used in trajectory generation, the gains were changed. For this TVLQR controller, $R = 2.5 \times 10^{-4}$, Q the zero matrix, and $Q_f = [100, 100, 0.1, 0, 0.25, 0.25, 0]^T$.

3.4.3 Hardware Implementation

After designing the nominal trajectory described in section 3.4.1 we implemented and ran this open-loop controller on our real glider using our real time control architecture, matching the initial conditions for which the controller was designed. As expected, because of dynamics that were not accounted for in the open-loop design, the maneuver quickly diverges from the nominal solution and misses the goal. Figure 3-12 shows a time-lapse comparison between our simulated vs. real results, using both open loop and TVLQR control. The real trajectories were recorded using our off-board control station and were used to draw the results shown. The figure clearly illustrates the effects of the unmodeled dynamics on the performance of the open loop controller. Notice that the vehicle quickly pitches up to a nearly vertical orientation, eventually missing the perch by approximately 60cm. The performance of this open-loop controller is not surprising, given that there is no feedback that can compensate for errors and the model used to design the controller was only approximate. Figure 3-13 shows a sequence of stills from a high speed video taken for the open-loop control experiment.

The feedback stabilizing controller on our glider hardware uses the feedback state information from the motion capture system to servo the glider's position relative to the desired nominal trajectory in order to compute the quantity $(\mathbf{x}[n] - \mathbf{x}_t[n])$ in equation 3.27. The feedback gain vector K in equation 3.28 summarizes everything about the TVLQR controller that was designed in the previous section. Therefore, computing this stabilizing controller on-line merely requires indexing the value of K at each time step n and performing a simple multiplication in order to obtain the corresponding control input at the corresponding time-step. In essence, the TVLQR controller knows it's state at each time step (obtained from motion capture) and knows where it wants to be (based on the nominal solution) and uses this difference to drive the glider back toward the nominal trajectory in order to reach the goal at the final time step. Although no explicit reasoning about modeling errors was involved in the control law design, modeling errors will show up as trajectory errors (as was seen in the open loop case) and so long as the estimated local gradients of the system are

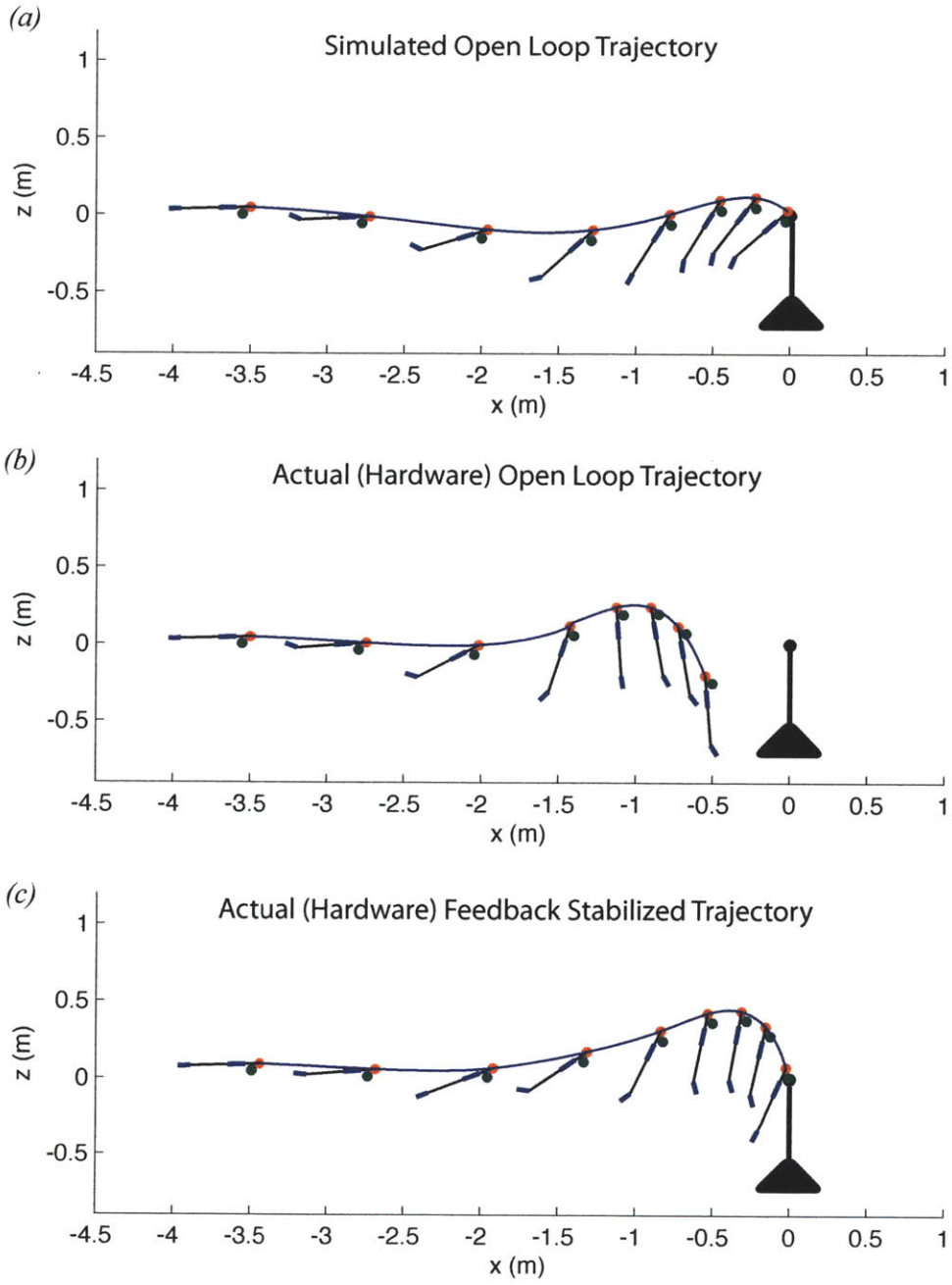


Figure 3-12: Trajectory comparisons for open-loop vs. TVLQR feedback in simulation and on real hardware. (a) The nominal trajectory obtained through the open loop control design procedure described in section 3.4.1. (b) The trajectory on the real glider when running this same open loop controller with the same initial conditions from (a). Because of unmodeled dynamics, the actual trajectory deviates from the predicted trajectory. (c) The TVLQR stabilized trajectory on the real glider. Using TVLQR feedback and the approximate model of the glider dynamics, the controller is able to successfully stabilize the trajectory.

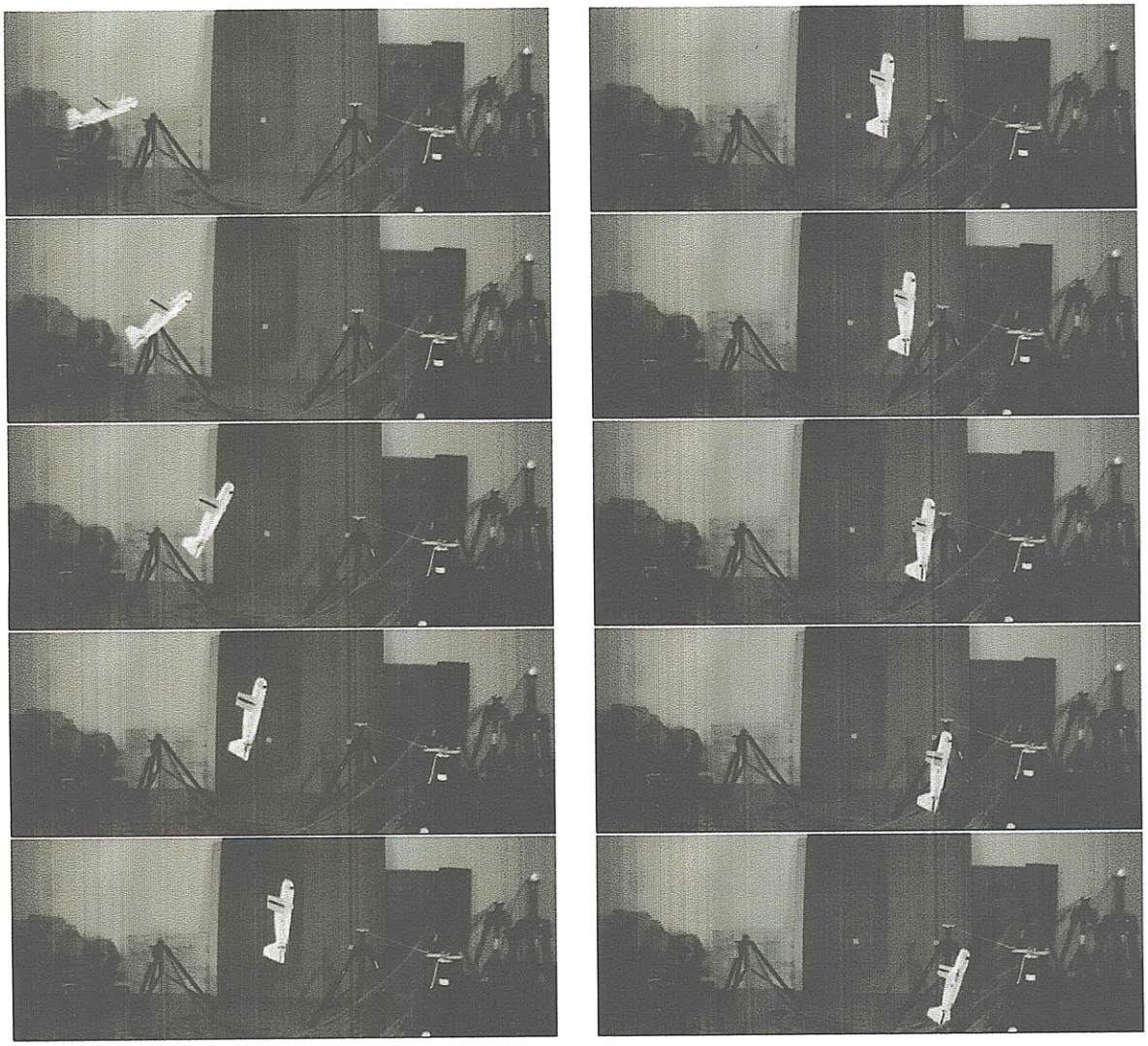


Figure 3-13: Stills from a failed open-loop trajectory played out on the real glider. Although initial conditions for the nominal condition used to solve the open-loop control were matched, the glider missed the perch by ~60cm and finishes with a very large pitch orientation.

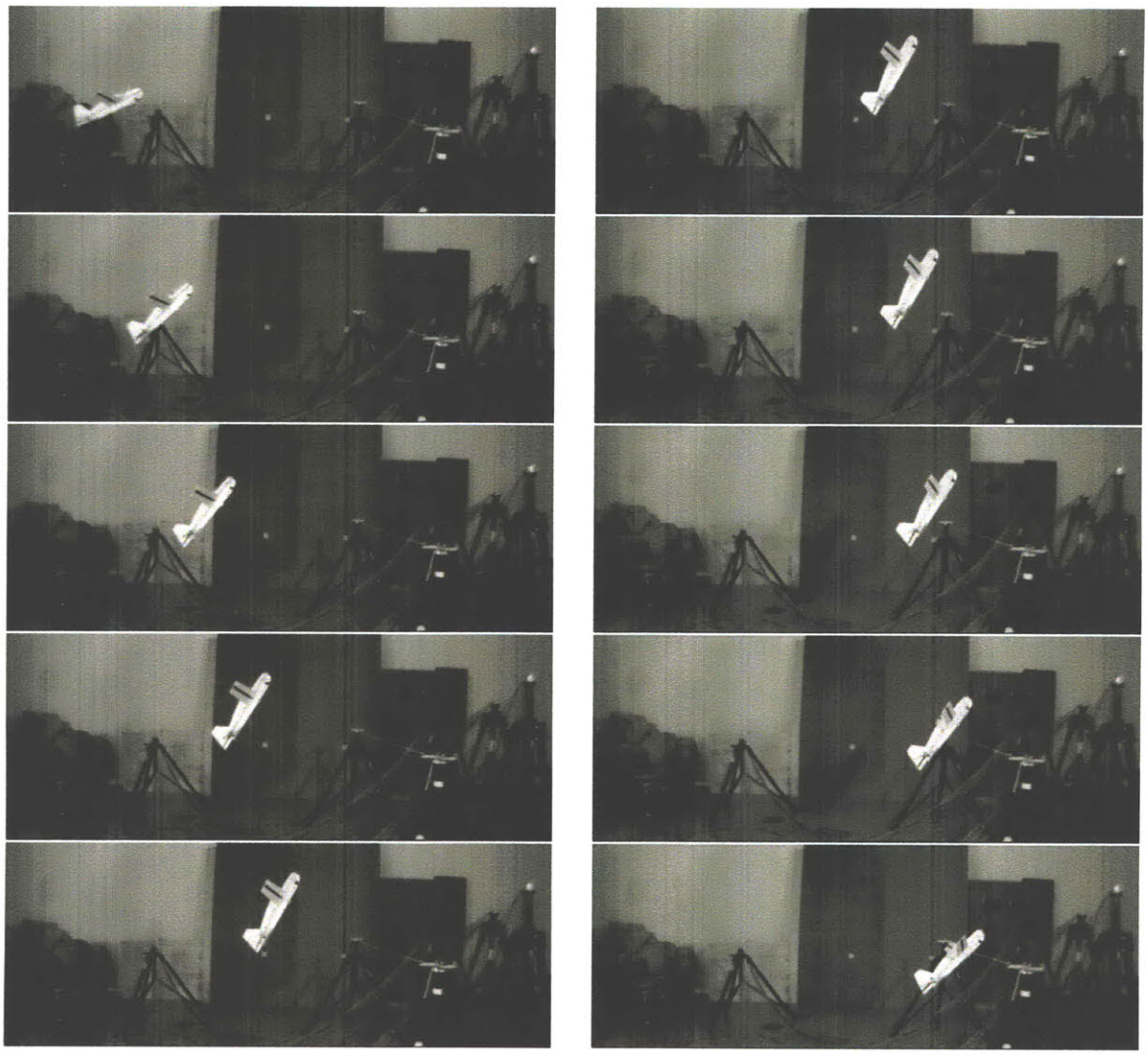


Figure 3-14: Stills from a successful perching maneuver using a stabilizing TVLQR feedback controller. The glider hook comes within <2cm from the goal and achieves a pitch angle that closely resembles the nominal solution toward the end of the maneuver.

approximately correct around the nominal trajectory, we can hope to cancel these errors in the same way we attempt to cancel disturbances.

As is shown in Figure 3-12(c), the implementation of the TVLQR control law on our real hardware was able to stabilize the perching maneuver and successfully make the point landing. Figure 3-14 shows a sequence of high speed video stills taken during the execution of a feedback stabilized (TVLQR) perching maneuver. The sequences shown in Figure 3-12 shows that the stabilized maneuver pitches much less than its open-loop counterpart, allowing it to generate less pressure drag to travel the extra distance needed to land on the string.

3.4.4 Robustness to Initial Speed

Here we investigate the robustness to initial launching speed of the TVLQR control system described in section 3.4.2. We perform this quantification in simulation mapping initial launching speed to final distance to the perch at the end of the trajectory, i.e. at the final time step of the time-horizon specified in equation 3.24. The goal is to understand the set of initial conditions that can be stabilized to the goal, given a perfect model of the glider (the model is perfect in simulation). Note that because the controller is linear and only locally valid around the nominal trajectory, only a bounded set of initial launching speeds can be stabilized to the goal (within a certain tolerance).

We investigate this robustness by launching the glider (in simulation) from initial speeds that range from 6 m/s to 9 m/s. We do this in two steps. The first step runs the open-loop controller (i.e. no feedback) designed for the nominal speed of 7 m/s and plots the final distance to the perch at the end of the last time step. Figure 3-15 shows (in blue) the mapping between initial launching speed and final error to the perch for this open loop control. We see there that as the initial launching speed is decreased or increased, the open-loop control solution begins to incur a final error in terms of distance to the perch. We similarly plot the performance of the TVLQR feedback controller. Contrary to the open-loop controller, feedback allows the TVLQR controller to stabilize disturbances in initial speed within the vicinity of the nominal initial speed (shown in Figure 3-15 (red)). Initial speeds falling within the interval of approximately 6.8 m/s - 7.4 m/s are shown to be successfully stabilized to the goal.

In order to characterize the performance of our simulation vs. the real glider, we did a similar initial speed robustness analysis using real hardware. Figure 3-16 shows the results. In total we ran 45 open loop control trials and 54 feedback control trials. 6 trials were discarded because of missing packets in the real time stream of the motion capture data. The glider was launched using our custom cross-bow launching device using a remote controlled trigger to minimize variability due to human launching. For a given launch, the variability in initial speed would typically fall within the range of ± 0.1 m/s. 4-6 trials were executed for a given trigger point on the launcher. And this trigger was adjusted so that initial speeds ranged from approximately 6 m/s to 8.5 m/s, which was a physical speed limitation of our launching device. The variance in the open loop data is relatively low, suggesting that the launching conditions were fairly repeatable. Because of unmodeled dynamics the open-loop controller performs very poorly around the initial speed for which it was designed, i.e., 7 m/s,

causing the open loop performance curve (blue) to shift to the right, relative to its simulated prediction (Figure 3-15). The TVLQR feedback controller is shown to have a dramatic effect on performance within the sampled range of launching speeds, consistently coming within ~5 cm of the goal. We note that the feedback data exhibits significantly higher variance that the open loop data. This is potentially due to the feedback controller relying on noisy motion capture data for feedback during flight. Also, discrepancies between the sampled control clock of the motor and our motion capture stream may contribute to this variability. Even with this increased variance, however, the region of initial speeds capable of being stabilized within a reasonable distance from the goal is much higher than that predicted from simulation.

Given the promising performance of our TVLQR implementation, the natural question that arises is: What advantage could we gain in practice if we could stabilize in simulation the entire set of initial conditions shown in Figure 3-15? Another words, we'd like to ideally flatten out the red curve down to the zero error axis. We can already see that if we were to take more than one nominal trajectory and corresponding feedback controllers, we would be able to stabilize different intervals within the set of initial speeds. The question then becomes: how do we 'stitch' these feedback controllers together so as to fully bring all the red data points in Figure 3-15 down to zero error and secondly, what effect would this have on the real performance of our glider. As we will see in chapter 4, new tools in feedback motion planning and trajectory verification will allow us to explore this possiblity.

3.5 Discussion

Our initial results using a value iteration type algorithm to obtain an optimal full-state feedback nonlinear policy reinforced the well-known fact that function approximation plays an important role in the performance of the computed controller, even in simulation. In our case, an algorithm using an extremely coarse representation of the value function (in six dimensions) was able to converge to a solution, but was extremely sensitive to changes in initial conditions, presumably because of the rather coarse meshing. We note that although more advanced methods of value function approximation for high-dimensional state spaces exists, e.g., algorithms such as fitted value iteration [Szepesvary and Munos, 2005] which rely on finding compact parametric models of the value function, our rather quick success in designing locally stabilized feedback trajectories in simulation inspired our transition to explore these types of local methods. Moreover, we felt that the perching task in particular exhibits characteristics that inherently lend themselves to local control methods. Namely, in the case of our perching glider there is virtually no ability to add energy to the system (assuming the elevator cannot act as a propellor). Therefore a major partition of the seven dimensional state space is not relevant when searching for a feedback controller. That is, certain states just don't have the energy to get to the goal, for example, being 3.5 meters away from the perch traveling at 0.5 m/s. Moreover, the difference between trajectories over the range of initial speeds for which we were able to compute open loop controllers (6.5-9 m/s) fell within a relatively small neighborhood of the state-space.

Given our success using local trajectory stabilizers, we have also started to explore the possibility of recomputing, in real time, a new nominal trajectory every control time step.

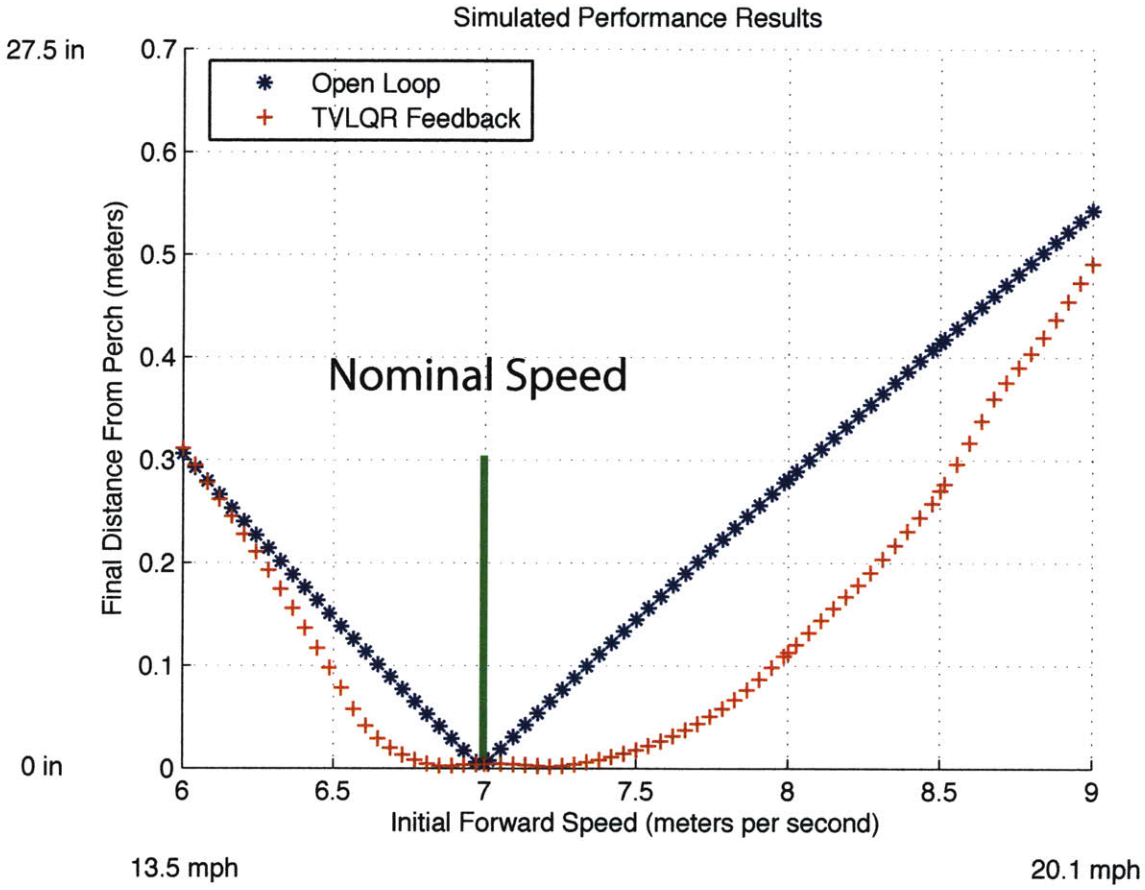


Figure 3-15: The robustness to variations in initial launching speed for the open-loop controller (blue) vs. TVLQR feedback controller (red). In order to avoid potentially large discontinuities in the the final error (an artifact of defining positive and negative distances to the perch) we use the absolute distance to the perch. Around the nominal launching speed (7 m/s) the error of the open-loop controller is zero, as expected. Moving away from the nominal initial speed results in a final distance error to the goal which monotonically increases as one moves away from the nominal initial speed. In contrast, the feedback TVLQR controller can compensate for variations in initial speed to successfully reach the goal with zero error (within tolerance). This occurs for initial speeds that fall approximately within the range of 6.8 m/s - 7.4 m/s.

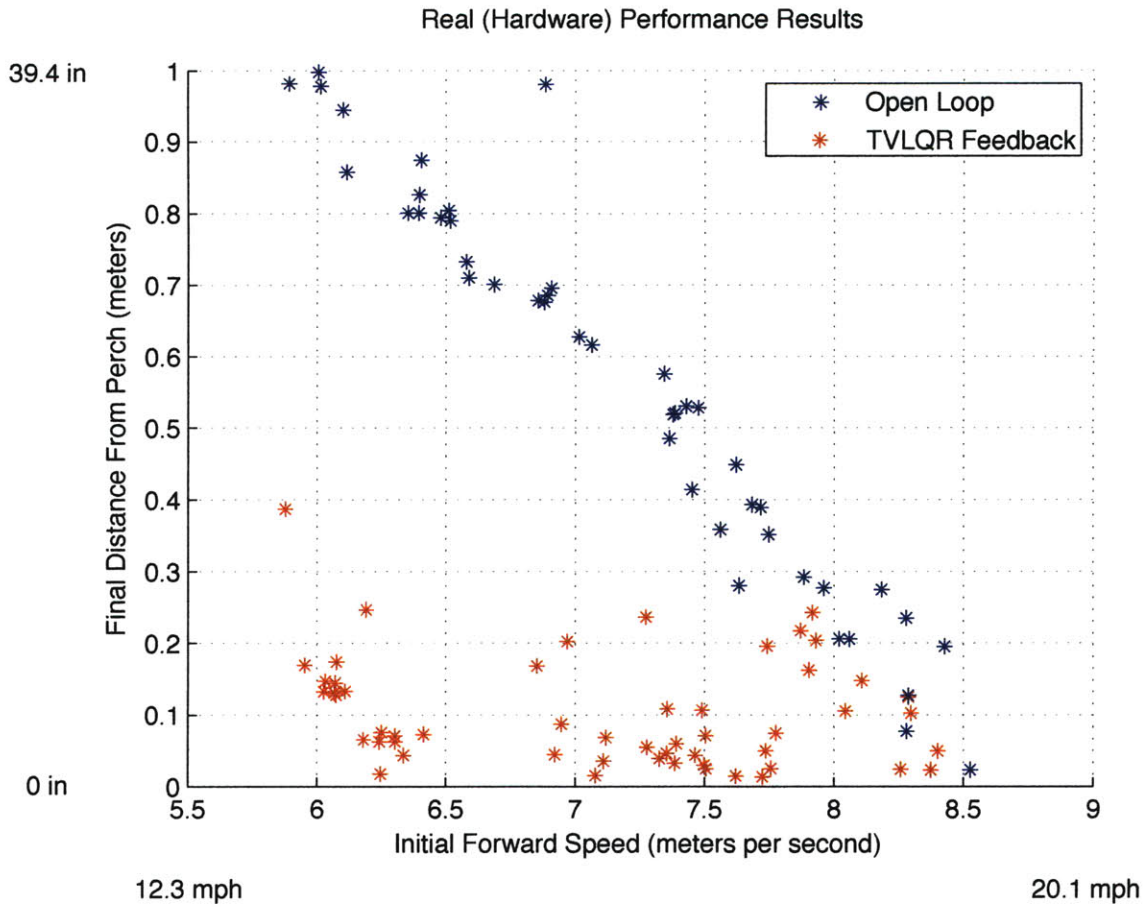


Figure 3-16: Real glider experimental results for the robustness to variations in initial launching speed for the open loop controller (blue) vs. TVLQR feedback controller (red). In total we ran 45 open loop control trials and 54 feedback control trials. 6 trials were discarded because of missing packets in the real time stream of the motion capture data. The glider was launched using our custom cross-bow launching device using a remote controlled trigger to minimize variability due to human launching. For a given launch, the variability in initial speed would typically fall within the range of ± 0.1 m/s. 4-6 trials were executed for a given trigger point on the launcher. And this trigger was adjusted so that initial speeds ranged from approximately 6 m/s to 8.5 m/s, which was a physical speed limitation of our launching device. The variance in the open loop data is relatively low, suggesting that the launching conditions were fairly repeatable. Because of unmodeled dynamics the open-loop controller performs very poorly around the initial speed for which it was designed, i.e., 7 m/s, causing the open loop performance curve (blue) to shift to the right, relative to its simulated prediction (Figure 3-15). The TVLQR feedback controller is shown to have a dramatic effect on performance within the sampled range of launching speeds, consistently coming within $\sim$5 cm of the goal. We note that the feedback data exhibits significantly higher variance that the open loop data. This is potentially due to the feedback controller relying on noisy motion capture data for feedback during flight. Also, discrepancies between the sampled control clock of the motor and our motion capture stream may contribute to this variability. Even with this increased variance, however, the region of initial speeds capable of being stabilized within a reasonable distance from the goal is much higher than that predicted from simulation.

Using this approach, we could potentially have the ability to expand the operating envelope of our local controllers, given that the trajectory design process reasons about the non-linearities of the dynamics. Our initial results in simulation suggest that this approach is in fact able to stabilize disturbances in initial conditions, and re-plan a nominal trajectory every control time-step. As of the writing of this thesis, we are preparing the real-time software to test this approach on the real glider in motion capture.

Although we have not yet formally quantified the robustness to other initial condition variables, initial simulation tests have shown that our feedback controller can stabilize appreciable disturbances in initial pitch ($> 22°$) as well as disturbances in initial height ($> 0.4m$), and initial horizontal position ($> 0.4m$). However, the sensitivity of these variables to disturbances is not symmetric about their nominal value. For example, a disturbance in horizontal position that results in a shorter initial distance to the perch is significantly more sensitive than that which results in a longer initial distance to the perch. Presumably this is because, since the control system wants to be at a certain horizontal position at a specific time, it is significantly easier for the system to slow down (i.e., loose energy through drag) than it is for it to catch up (i.e., add energy). Ideally the controller would be independent of time, and intuitively, choose the state that is closest to its current state and re-index into the controller, taking time out of the equation. A well known solution to this problem is that of using a transverse linearization around the nominal trajectory [Shiriaev et al., 2009]. In effect, this type of linearization allows the controller to index the control through the dynamics that are transverse to the local trajectory, using an appropriate indexing variable, e.g., one that is monotonically decreasing over the trajectory. This is the subject of some of our future work.

Chapter 4

Feedback Control Using Trajectory Libraries

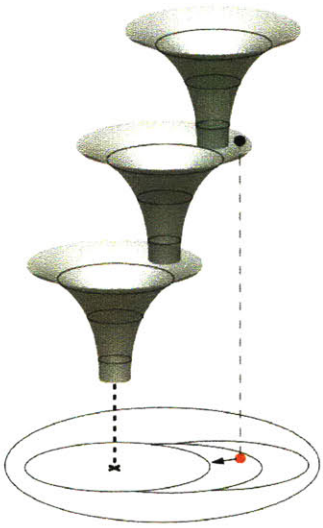


Figure 4-1: Geometric interpretation of feedback control with funnels adapted from [Burridge et al., 1999]. By stitching together local stabilizing controllers, represented by individual funnels in this illustration, we can guarantee that a larger portion of the operating region can be stabilized to the goal.

We saw in the previous chapter how a local controller based on a feedback stabilized nominal trajectory was able to compensate for model errors as well as disturbances in initial speed. Although this type of local control approach tends to sacrifice global optimality, the performance on real hardware yielded impressive results. Here we take this approach a step further and design a controller based on these local methods using new design tools for feedback motion planning algorithms, namely LQR-Trees [Tedrake, 2009, Reist and Tedrake, 2010]. We design a sparse library of feedback stabilized trajectories that will enable us to widen the

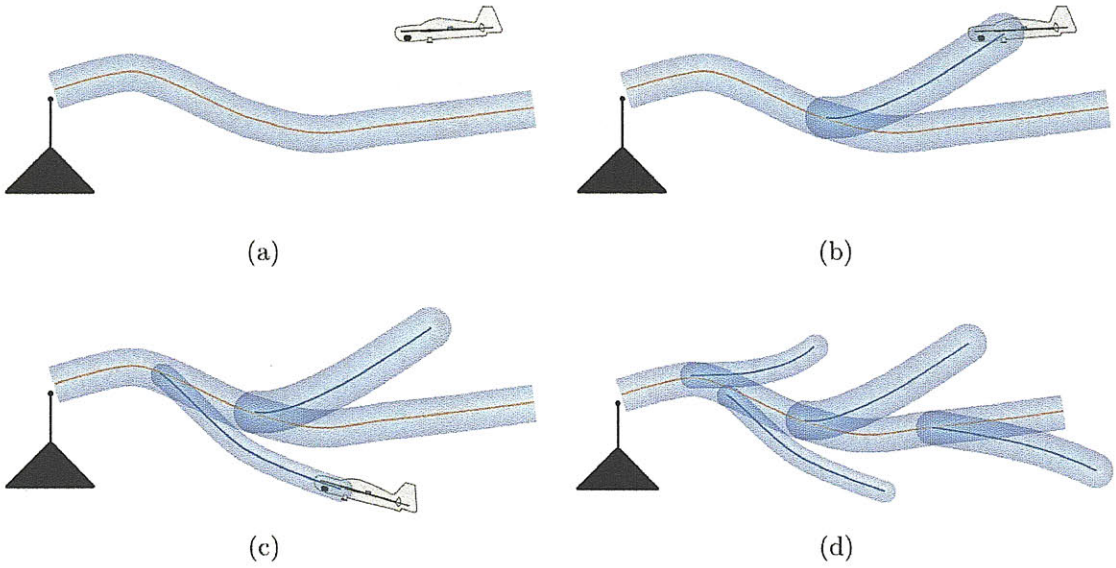


Figure 4-2: Feedback motion planning with LQR-Trees [Tedrake, 2009, Reist and Tedrake, 2010]. This cartoon illustrates the design evolution concept of the algorithm over a two dimensional slice of the state-space, i.e., (x, z) positions. As we explore parts of the state space that are not guaranteed to stabilize to the goal (verified through simulation using a candidate local controller) we can add new feedback stabilized nominal trajectories with their own basins of attractions (blue). This is done until all states capable of getting to the goal fall within some basin. The result is a sparse library of stabilized trajectories that guarantee that any state that *can* get to the goal *will* get to the goal.

basin of attraction for our perching control system. One of the main motivations behind the LQR-Trees control design algorithm is scalability and planning in continuous space, which, as we shall demonstrate, will enable the design of a control policy that is compact as well as consistent with the continuous dynamics of the aircraft, unlike the approach explored in section 3.3.

At the highest level, we can imagine stitching together a library of our TVLQR stabilized trajectories in order to cover as much of the relevant state space as possible in order to guarantee that any given state that *can* get to the goal *will* get to the goal. In order to efficiently cover the relevant space, in terms of the number of trajectories necessary, one would require a geometric description of the basins of attraction for each stabilized trajectory. If this information were available, one could imagine covering the relevant space with as few trajectories as possible, reasoning about their ability to stabilize neighboring states. The LQR-Trees algorithm allows us to do just that. Figure 4-2 shows a cartoon of how this process might evolve, beginning with a single stabilized nominal trajectory. We subsequently sample states outside of the nominal controller's basin of attraction and grow a trajectory back to the nominal solution [1]. As this process evolves, a tree structure will fill the relevant space.

[1] In general, we grow the sampled state to the closest node in the existing tree using a distance metric.

4.1 Trajectory Libraries and Controller Composition

Inspired by recent tools in randomized motion planning control algorithms [LaValle, 2006], the goal of the LQR-Trees approach is to design a library of local trajectories that are feedback stabilized with certifiable bounds on their regions of stability (a.k.a. basins of attraction). The algorithm uses these bounds in a design procedure, which yields a feedback controller that (probabilistically) covers a bounded region of state space, even for nonlinear systems. This is done by composing these local controllers in a way that reasons about their individual basins of attraction. The ability to verify and bound basin of attraction guarantees for local feedback controllers allows the control design process to explicitly and formally reason about the limitations of using linear controllers on a linearization of an inherently non-linear dynamical system, as is the case with our perching glider.

A beautiful geometric interpretation for the sequential composition of controllers is presented in [Burridge et al., 1999]. There, an analogy is made to a series of stacked funnels that, if arranged properly, effectively widen the capture region of the original funnel, as shown in Figure 4-1. Using this interpretation, any given state in the domain of the controller that is capable of being driven to the goal, represented by the x-y plane in Figure 4-1, when projected up to the funnel surface will always evolve downwards due to the closed-loop dynamics. When this downward state evolution on the funnel surface is projected back onto the domain (i.e., the state space), it will always follow a trajectory that converges to the goal.

The LQR-Trees algorithm explicitly and formally reasons about the placement and geometry of these local feedback controllers in order to obtain a control solution which guarantees that any state within a bounded region of state space capable of getting to the goal will stabilize to the goal. As we shall see, this procedure will produce a controller with a much larger basin of attraction when compared to that of our TVLQR solution.

4.2 Simulation Based LQR-Trees

Here we briefly describe simulation based LQR-Trees in the context of perching control design. The differences between simulation based LQR-Trees [Reist and Tedrake, 2010] and sum-of-squares (SOS) based LQR-Trees [Tedrake, 2009] lies in the basin verification procedure. [Tedrake, 2009] uses an algebraic (SOS) method to formally verify basins and provides proper stability guarantees, whereas [Reist and Tedrake, 2010] uses a non-conservative approximation providing probabilistic guarantees. The simulation based approach, however, is simpler to implement in practice. For instance, it can easily check state constraints and enforce input constraints in simulation, whereas the formal approach requires a more subtle treatment of these.

The goal here is to design a library of TVLQR stabilized trajectories that probabilistically cover the region of state space relevant to the perching task. One way to define a bound over the set of states relevant to our perching task is through elimination of states based on minimum and maximum energy requirements. For instance, a large part of the state space will never be able to reach the goal because of insufficient speed, e.g., when the glider is 3.5 m away from the perch with a speed of 1 m/s. For now we assume that this bounded

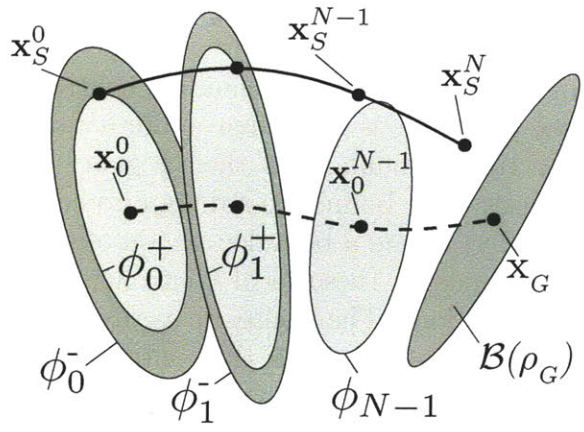


Figure 4-3: An illustration of the falsification procedure which takes place once a sampled state has failed to reach the goal using the stabilizing controller of a nearby nominal trajectory (figure adapted from [Reist and Tedrake, 2010]). State superscripts and funnel parameter subscripts indicate adjusted time indexing, i.e., $\mathbf{x}_0^0 = \mathbf{x}_0[n]$ and $\mathbf{x}_S^0 = \mathbf{x}_S[k-n]$. We see that the sample state $\mathbf{x}_S^0$ when simulated forward does not reach the goal. The falsification procedure consists of shrinking the ellipsoids corresponding the each nominal state. Here the initial ellipsoids are shown in dark grey, and the falsification procedure results in the ellipsoids indicated in light grey.

region of state-space is known and defines a set of state constraints on our control problem. The general procedure for designing a simulation based LQR-Trees controller begins with defining a goal state whose neighboring dynamics are approximated using a time-invariant linearization about the goal. As described in [Reist and Tedrake, 2010], the system dynamics must be zero at the goal (i.e., a system equilibrium) in order to compute a controller that locally stabilizes the goal for all of time, i.e., we require that $\mathbf{f}(\mathbf{x}_G, \mathbf{u}_G) = 0$. In the case of our perching task, however, we do not model the instantaneous collision events that define a successful perch. Therefore, by simply using our glider dynamics without a collision model, there exists no equilibriums with respect to all state variables, i.e., it is always the case that $\mathbf{f}(\mathbf{x}_G, \mathbf{u}_G) \neq 0$. In order to mitigate this limitation, we manually define a continuous set of goal states

$$\mathcal{B}(\rho_G) = \left\{ \mathbf{x} \mid (\mathbf{x} - \mathbf{x}_G)^T Q_f (\mathbf{x} - \mathbf{x}_G) \leq \rho_G \right\}, \tag{4.1}$$

where ρ_G is a manually chosen threshold and Q_f is a positive definite cost matrix. Here ρ_G is chosen such that $\mathcal{B}\rho_G$ contains states that are within a few centimeters away from the perch with a sufficiently small velocity, on the order of 1 m/s. Equation 4.1 defines a continuous set of states that are considered successful perched states. The problem then becomes that of designing a controller that regulates all states within our bounded region of interest to the set $\mathcal{B}(\rho_G)$. Here we give a high-level sketch of the design procedure. The reader is referred to [Reist and Tedrake, 2010] for a more complete and formal treatment.

The design process is iterative, beginning with a single trajectory and growing into a 'tree' of feedback stabilized trajectories that (probabilistically) guarantee that all states within our bounded set capable of reaching the goal will reach the goal. Given an initial (arbitrary) TVLQR stabilized trajectory obtained through the same design procedure describe in section

3.4 we assign each nominal state in that trajectory $\mathbf{x}_0[n]$ a corresponding funnel parameter $\phi[n] \in \Re$, which describes a hyper-ellipsoid around each of these nominal states. A state $\mathbf{x}$ is inside this ellipsoid if

$$(\mathbf{x} - \mathbf{x}_0[n])^T S[n](\mathbf{x} - \mathbf{x}_0[n]) < \phi[n], \tag{4.2}$$

where the matrix S defines the optimal cost to go for the linearized state vector $\bar{\mathbf{x}}$, both at time n

$$J^*[\bar{\mathbf{x}}, n] = \bar{\mathbf{x}}^T[n] S[n] \bar{\mathbf{x}}[n]. \tag{4.3}$$

At the beginning of this process we don't know the correct magnitude of the $\phi[n]$, i.e., the width of the funnel. Therefore for each new added trajectory (beginning with the first) we set $\phi[n] \rightarrow \infty$, indicating that the funnel initially covers the entire state space. We next randomly sample a state $\mathbf{x}_S$ and must decide whether it is in the basin of the existing controller(s) or whether a new trajectory to the goal is required [2]. If the sampled state $\mathbf{x}_S$ lies within the ellipsoid of an existing nominal state $\mathbf{x}_0[n]$, then we simulate the system from that point using the corresponding TVLQR controller for the nominal trajectory beginning at $\mathbf{x}_0[n]$ in order to verify that it is in fact within that trajectory's funnel. If the simulation succeeds we simply continue on to sample another state. If the simulation fails, then the ellipsoid for the existing nominal state was an over-estimate and a falsification procedure takes place, where every funnel parameter along the nominal trajectory, starting with $\phi[n]$, is decreased to

$$\phi[k] = (\mathbf{x}_S[k-n] - \mathbf{x}_0[k])^T S[k](\mathbf{x}_S[k-n] - \mathbf{x}_0[k]), \quad k = \{n, \cdots, N-1\}, \tag{4.4}$$

where N is the number of nodes in the nominal trajectory starting with $\mathbf{x}_0[n]$. Figure 4-3 illustrates a single iteration of this falsification procedure. Note that the funnel parameters are only decreased and never increased. We subsequently compute a new trajectory from our sample state $\mathbf{x}_S$ back to the goal as well as compute its corresponding stabilizing feedback controller. The algorithm terminates when we encounter a fixed number of samples that successfully reach the goal using the existing tree.

4.3 LQR-Trees for Perching Control

We implemented simulation based LQR-Trees in simulation using the basic procedure outlined in the previous section. In our perching implementation, we restricted state sampling to the set of initial conditions over speed and all new trajectories were taken directly to the goal, rather than growing back to a neighboring trajectory. Given our state sampling was done over a restricted set of initial conditions, the number of trajectories required to cover this set was typically on the order of 4-5, and therefore planning back to the goal for this small set of trajectories was relatively inexpensive. The free parameters in equation 4.1 were chosen to be $Q_f = [2000, 2000, 100, 0, 20, 10, 0]^T$, i.e., the same final cost matrix as was chosen for the trajectory optimization, and $\rho = 40$ was chosen as the cost threshold where

[2] The general algorithm attempts to grow a trajectory toward the nearest point on the existing tree using an appropriate distance metric. Special treatment is given to sampled states that fail to connect to the nearest node. In the implementation presented here, we simply grow the trajectory back to the goal.

all states having a cost lower than this value were deemed successful perched states. These cost gains and threshold defining successful perched states were chosen through a process of manual trial and error tuning in simulation until the resultant trajectory balanced a tradeoff between high accuracy and minimum speed at the goal, characteristic of an avian landing maneuver. Our restricted set of initial conditions were chosen such that all but one of the state variables were equal, i.e., only the initial forward speed of the glider varied over the sampled states. The sampled state was chosen as $[-3.5, .1, 0, 0, \dot{x}, 0, 0]^T$, where $\dot{x}$ was sampled in the interval 6.0 m/s $\leq \dot{x} \leq$ 9 m/s. The unvaried initial state variables were chosen to be equal to those of the nominal trajectory computed in section 3.4.2. As was explained there, the range of initial speeds and distance to the perch were chosen based on the limitations on the bandwidth and capture volume of our motion capture system. The purpose of the 0.1 radian initial pitch was to provide the vehicle with a non-zero angle of attack at launch.The algorithm terminated after 20 consecutive samples were successfully driven to the goal using the existing tree. A typical run of the algorithm would terminate successfully after approximately 7-10 consecutive samples. Therefore 20 was chosen as a conservative termination threshold.

In order to use the tree policy, nodes (over initial conditions) were ordered based on a confidence metric [Tedrake, 2009, Reist and Tedrake, 2010]. Each candidate node was evaluated in terms of its estimated cost to go, making sure the sampled state would fall within its basin of attraction and choosing the node with the largest confidence margin. Specifically, node n was chosen if its associated cost to go satisfied the following conditions

$$\rho_n[0] - J_n^*(0, \bar{\mathbf{x}}) = \rho_n[0] - (\mathbf{x}_S - \mathbf{x}_n[0])^T S_n[0](\mathbf{x}_S - \mathbf{x}_0[0]) > 0 \quad (4.5)$$

$$\rho_n[0] - J_n^*(0, \bar{\mathbf{x}}) = \max(\rho_k[0] - J_k^*(0, \bar{\mathbf{x}})), \forall k \quad (4.6)$$

Figure 4-4 illustrates the basin of attraction of our original TVLQR controller (section 3.4.2) v.s. that of the newly computed LQR-Trees controller. The LQR-Trees controller expands the region of attraction by approximately a factor of four. Most of the initial speeds are successfully brought to within tolerance of a successful perched state, with the exception of those falling below approximately 6.3 m/s, where the glider simply doesn't have enough energy to be driven to the goal.

4.4 Discussion

The most general form of the LQR-Trees algorithm samples over a bounded region of the full state space, which includes guarantees for states falling outside the set of initial conditions and can address disturbances that occur after maneuver entry. For this task, given our large seven dimensional state space, sampling and covering this entire space was computationally inefficient due to weak bounds on our sample space. As explained in the text, one approach would be to narrow the search space based on energy. This is the subject of future work. Our LQR-Trees algorithm implementation sampled over initial speeds. This one dimensional parameterization was sufficient to stabilize a relatively large range of initial speeds over their entire trajectory (as compared to TVLQR in simulation), using only relatively few (4-5) nominal stabilized trajectories. In order to increase the number of initial states that can be

guaranteed to get to the goal (if they are in fact capable of reaching the goal), one could sample over n-dimensional sets of initial conditions. Adding extra dimensions to the initial condition parameterization could also mean better performance during maneuver execution, potentially adding extra trajectories that would otherwise be omitted when sampling over one dimension. The sum-of-squares (SOS) version of LQR-Trees [Tedrake, 2009] could also help mitigate some of the problems associated with planning in high dimensional state spaces, as is the case for our glider. Direct algebraic computation of basins for a given trajectory doesn't require any forward simulations, and instead this verification is offloaded to an efficient algorithm that verifies the postive-definiteness of a polynomial representation of the dynamics and the cost-to-go. Recent progress on the inclusion of actuator limits in the SOS version of the algorithm is showing promise for the application to perching control.

A natural extension of this algorithm would be to the domain of robust control, i.e., taking into account model errors or stochastic disturbance models. If one had a class of disturbance models available (or similarly a class of model errors), the verification would simulate forward by first uniformly sampling over these models. As the number of samples approaches ∞, one would obtain a conservative 'worst-case' robust control solution.

Our initial experiments on the real glider suggest that some initial condition trajectories have smaller basins of attraction than predicted. It is possible that the estimation of basins for some trajectories are more sensitive to modeling errors than others. As was seen for the performance plot of the TVLQR controller on the real glider, a single nominal trajectory covered a wide range of initial speeds, much larger than predicted by simulation, albeit with relatively high variance. In order to mitigate some of the difficulties associated with approximating basins for the real glider in spite of modeling errors, we could modify the algorithm to add new trajectories and adjust basins using experiments on the real glider. Instead of simulating, a failed trajectory on the real glider could be used as an indicator of where a new nominal trajectory is needed. If done enough times, the basin 'gaps' would essentially be filled in using the information from trials on the real glider. This online control design procedure is also the focus of future work.

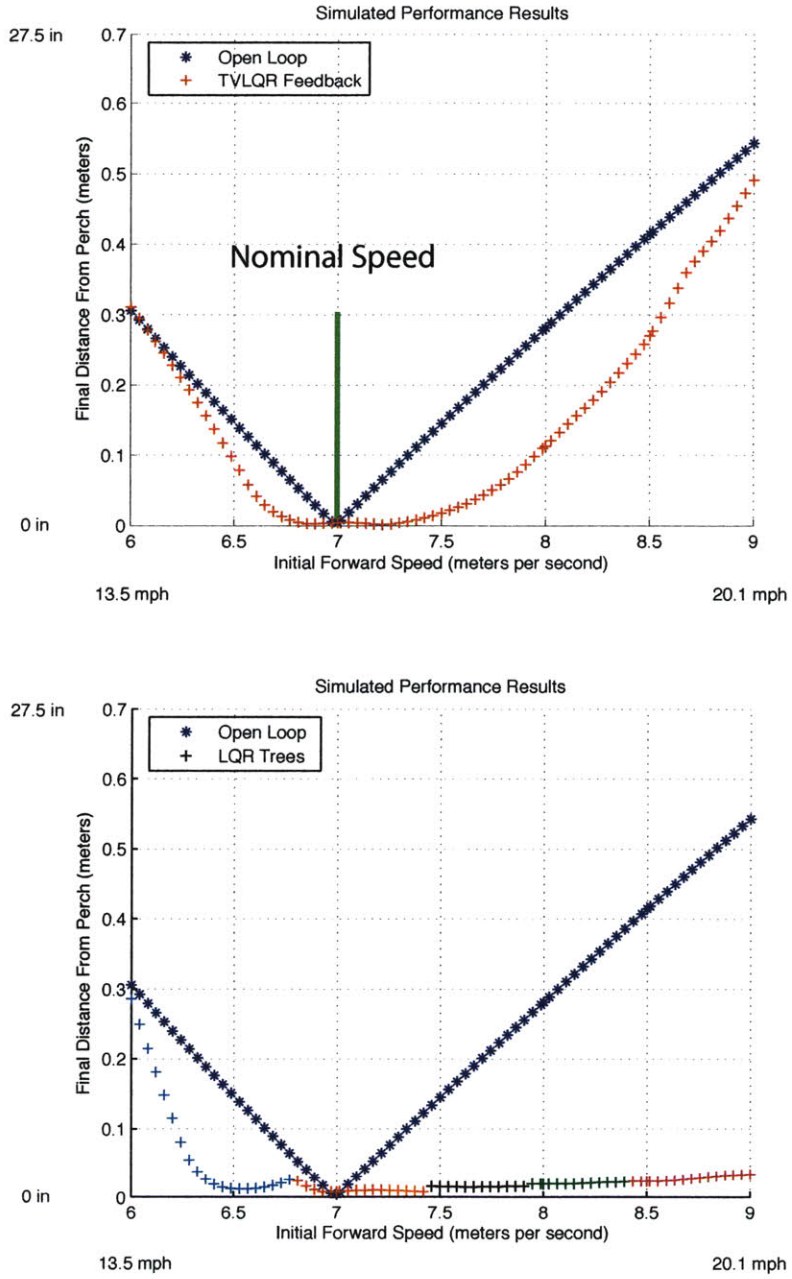


Figure 4-4: Comparison between the TVLQR basin of attraction (top) vs. that for LQR-Trees (bottom). In order to avoid potentially large discontinuities in the the final error (an artifact of defining positive and negative distances to the perch) we use the absolute distance to the perch. Using a library of feedback stabilized trajectories, the region of attraction over initial speeds increases approximately by a factor of four. The LQR-Tree curve shows a color-coded indication of when a new trajectory branch is chosen for a given initial speed. We note that below approximately 6.3 m/s, the glider doesn't have the initial speed/energy necessary to reach the goal, which causes a divergence from the horizontal zero error axis.

Chapter 5

Robustness of Fixed Wing Perching

5.1 Controllability

It is well known that the requirements for controllability can be reformulated as a *finite-horizon, minimum-energy, fixed final-state, linear-quadratic (LQ) optimal control problem* [Lewis, 1992]. More formally, our discrete-time linear time-varying system

$$\bar{\mathbf{x}}[n+1] = A[n]\bar{\mathbf{x}}[n] + B[n]\bar{\mathbf{u}}[n], \quad \bar{\mathbf{x}}[0] = \mathbf{x}_0, \tag{5.1}$$

is said to be controllable over $[n_0, N]$ if we can find a finite energy control input sequence $\bar{\mathbf{u}}[n]$ which drives the final state to zero, i.e.,

$$\bar{\mathbf{x}}[N] = \mathbf{0}. \tag{5.2}$$

It can be shown that in order for 5.1 to be controllable, the symmetric positive semi-definite controllability Gramian given by

$$G_c[n_0, N] = \sum_{n=n_0}^{N} \Phi[n_0, n]B[n]R^{-1}B^T[n]\Phi^T[n_0, n] \tag{5.3}$$

must be nonsingular, where $\Phi[\,\cdot\,,\cdot\,]$ is the state-transition matrix of $A[\,\cdot\,]$ and is commonly used to express the general solution for LTV dynamics [Chen, 1998]. The expression given by 5.3 isn't practical for numerical evaluation since it requires knowledge of the state transition matrix Φ which isn't readily available. However, a well known property of the controllability Gramian is that it satisfies

$$G_c[n-1, N] = A[n]G_c[n, N]A^T + B[n]R^{-1}B^T[n], \; G_c[N, N] = 0 \tag{5.4}$$

which can be computed backwards in time in order to yield the controllability at each time step. Using the LTV dynamics and solving equation 5.4, we can verify that the perching trajectory obtained in section 3.4.1 is in fact controllable during most of the maneuver. Figure 5-1 shows the rank evolution of the controllability Gramian over the perching trajectory computed in section 3.4.1. This controllability Gramian was computed for our discrete-time

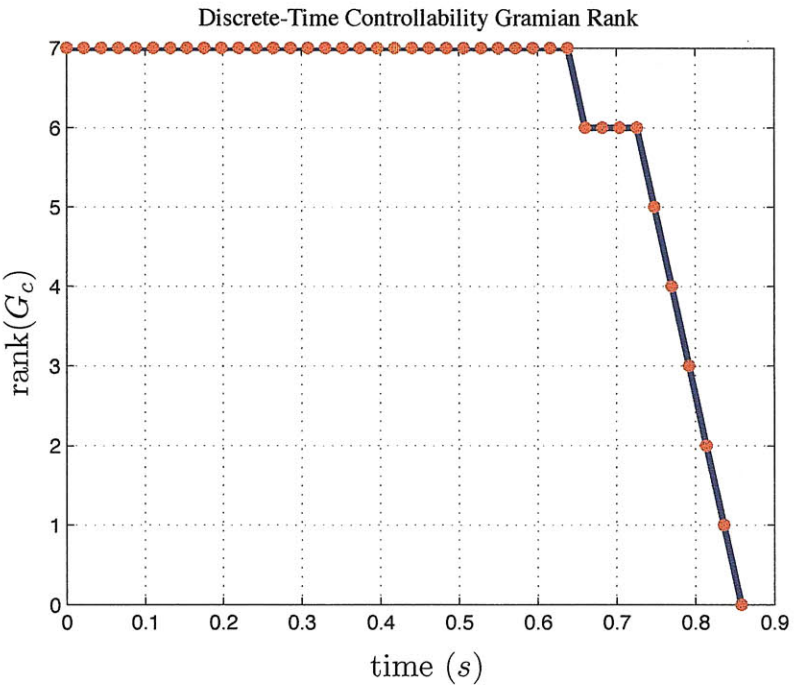


Figure 5-1: Rank evolution of the controllability Gramian over the perching trajectory computed in section 3.4.1. This controllability Gramian was computed for our discrete-time LTV dynamics with a control input period of 22ms, i.e., a control loop running at approximately 45Hz, simulating the control input frequency of our actual hardware implementation. As shown here, the LTV dynamics are controllable for most of the trajectory. The rank deficiency during the last seven time steps correspond to the evolution of our single input to the glider's seven state variables (it takes seven steps for our single input to influence all state variables). A singular value below the threshold of Matlab's `rank` command causes the Gramian to be characterized as singular for more than seven time steps.

LTV dynamics with a control input period of 22ms, i.e., a control loop running at approximately 45Hz, simulating the control input frequency of our actual hardware implementation. The LTV dynamics are controllable for most of the trajectory until the last 10 time steps, where the Gramian becomes rank-deficient. The rank deficiency during the last seven time steps correspond to the evolution of our single input to the glider's seven state variables (it takes seven steps for our single input to influence all state variables). A singular value below the threshold of Matlab's `rank` command causes the Gramian to be characterized as singular for more than seven time steps. Note that in general, the transition to a rank-deficient Gramian for a discretized continuous-time system (as is the case for our glider) will depend on the control input rate, potentially pushing the transition point arbitrarily close to the final time step. For our particular hardware setup, we have a strict limit on the control-loop frequency, limited by the bandwidth of our offboard communication.

Here we have shown that the linearized glider dynamics are controllable. However, the notion of controllability presented here does not bound the input sequence $\bar{\mathbf{u}}[n]$. This implies

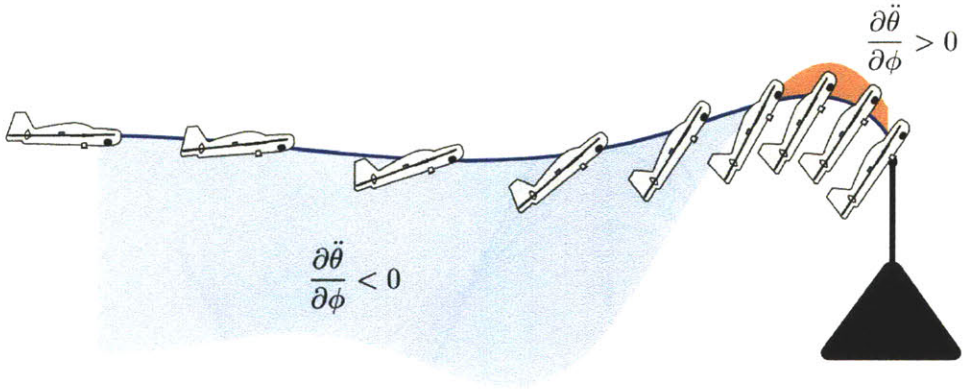


Figure 5-2: Instantaneous control authority over pitching moment.

that although there exist 'some' bounded control input that can drive the system to the goal for any given initial state $\mathbf{x}_0$ (indicated by the Gramian rank), it may in fact be beyond the physical limitations of our actuator. Hence our ability to regulate the LTV dynamics to the goal may greatly depend on how close we are to hitting actuator limits. Moreover, we'd like to characterize our ability to regulate disturbances in practice given that the glider is a highly underactuated system, i.e., we have one control input controlling seven state variables. Additionally, as we will show in the following section, there is weak coupling between the input and state variables as the air speed decreases near the goal, as well as state dependent sing changes in the relationship between the elevator input and the system response. Therefore, in order to quantify the 'degree' to which we can regulate our system to the goal in practice given these limitations, we will need a richer (i.e., non-binary) quantifier. This is explored in the next section.

5.2 Instantaneous Control Authority

Feedback linearization (a.k.a. inverse dynamics control) is a common flight control framework that relies on the separation of time-scales between state variables [Levine, 1996]. The dynamics are separated into two time-scales: fast-dynamics corresponding to those state variables that are directly affected by the control inputs and evolve faster (e.g., pitch rate, roll rate, yaw rate) and slow-dynamics corresponding to variables which are not directly influenced by the inputs and evolve slower (e.g., angle of attack, climb rate, etc.) [Menon et al., 1987, Snell et al., 1992]. This framework relies on the controller's ability to closely track the desired set of fast time-scale variables through direct control surface deflections. The slow time-scale control problem is then solved separately assuming the fast variables as control inputs. This separation of time-scales is a well-studied phenomena in inverse dynamics control and relies on the fact that, *independent of the control solution*, standard control surface inputs contribute relatively small forces to the aircraft dynamics, and instead contribute mostly to moments. The more general form of this type of control, also known as partial feedback linearization, is also a well known and understood tool in

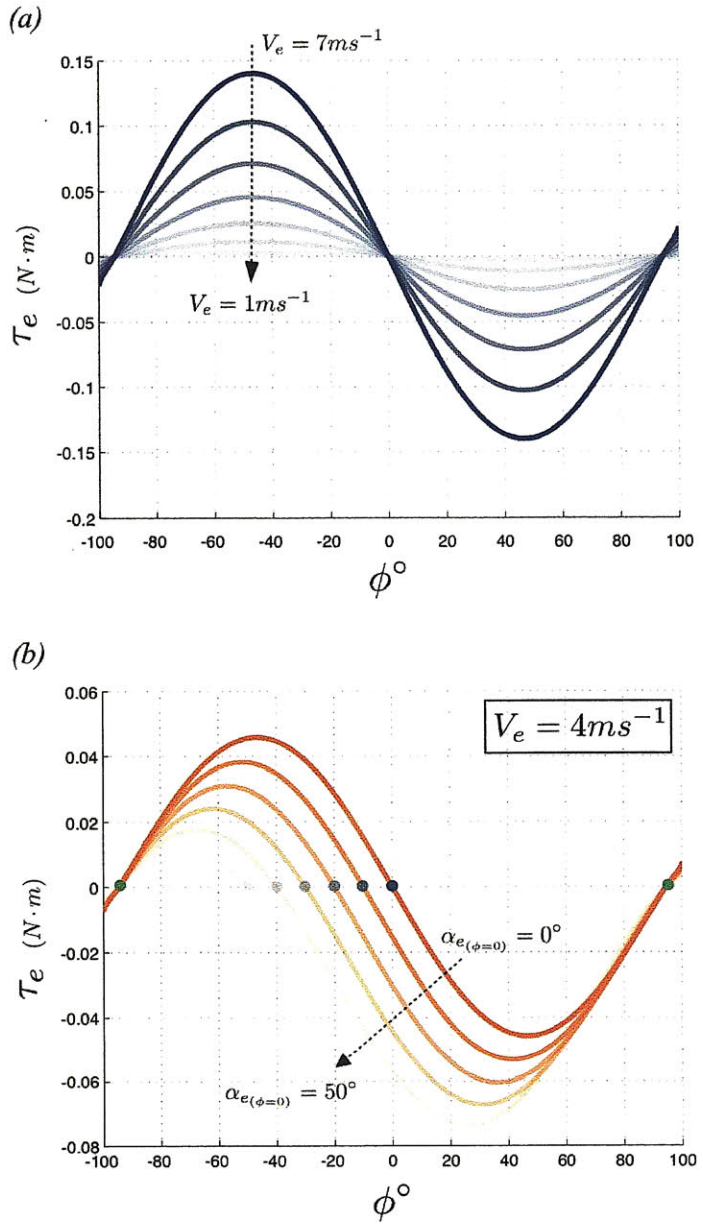


Figure 5-3: (a) The moment produced as a function of elevator angle for airspeeds ranging from 7 m/s down to 1 m/s. Note that as airspeed approaches zero the magnitude of the available moments diminish, following an inverse square relationship to the velocity. Notice also that there are three elevator zero moment contributions from the elevator, two of which correspond to the elevator surface being perpendicular to its moment arm with respect to the glider's center of gravity, and the third being when the angle of attack of the elevator is zero. (b) As the angle of attack on the elevator at zero degrees changes, the corresponding moment curve shifts diagonally, while maintaining it's two extreme zero-moment points fixed. The zero-moment point corresponding to zero angle of attack shifts along the horizontal axis.

robotics.

In the case of our perching glider, separation of time-scales results in a single fast time-scale variable, $\dot{\theta}$. This, in effect, is the proxy variable through which we can affect the rest of the glider dynamics during the execution of the perching maneuver. An important question to ask is: How does our ability to influence $\dot{\theta}$ (through $\ddot{\theta}$) change over the course of the trajectory, particularly as the airspeed over the control surface approaches zero? Figure 5-2 illustrates the relative level of instantaneous control authority over pitch acceleration (or equally pitching moment). We see that as the trajectory evolves and the speed of the aircraft decreases, the magnitude of this gradient decreases as expected. Perhaps more interesting is the fact that this gradient becomes zero before completely changing signs toward the end of the trajectory. In order to understand this behavior we can analyze the dynamics of our glider during these transients.

First we notice that there exists three values of ϕ that result in a zero moment contribution from the elevator. The first of these occurs when the elevator angle of attack is zero, causing the force on the elevator to be zero, and the other two, which are independent of glider state, occur when the position vector of the elevator with respect to the center of gravity is parallel to the force acting on the elevator, causing the moment arm to be zero. These three points are given by

$$\phi_0 = \left\{ \tan^{-1}(\dot{z}_e, \dot{x}_e) - \theta,\ \pm\cos^{-1}\left(\frac{l_e}{l_h}\right) \right\}. \tag{5.5}$$

Figure 5-3 (left) shows the elevator moments as a function of elevator airspeed and angle ϕ, for the case where the angle of attack of the elevator at ($\phi = 0$) is zero, i.e., ($\alpha_{e_{(\phi=0)}} = 0$). The fixed elevator zero moment points occur at ($\phi = \pm 94.67°$) and the elevator zero moment point associated with zero angle of attack occurs at $\phi = 0$. We also see the intuitive result that as the elevator airspeed decreases the maximum torque output over the entire range of ϕ diminishes, following an inverse square relationship to the velocity. This phenomena is also responsible for the diminishing magnitude of $\partial\ddot{\theta}/\partial\phi$ in Figure 5-2.

The figure also illustrates how a negative elevator deflection, i.e. ($\phi < 0$), can have both positive and negative gradients depending on the magnitude of ϕ, similarly for $\phi > 0$. Intuitively this means that there is a limit to how far the elevator can deflect before a further increase (or decrease) will start to generate the opposite effect. This gradient sign switch is what we see toward the end of the trajectory in Figure 5-2. The right of Figure 5-3 illustrates how changing the angle of attack relative to $\phi = 0$ diagonally shifts the moment curve and horizontally shifts the elevator zero moment point due to angle of attack. This was computed for the (arbitrary) mean trajectory velocity of 4 m/s. Computing this curve for a larger/smaller speed would simply scale the curve respectively about the zero torque axis, keeping the elevator zero moment points fixed.

In the next section we will see that although our instantaneous control authority over pitching moment diminishes as the glider approaches the perch as shown above, the controller's sensitivity to wind-gusts also diminishes.

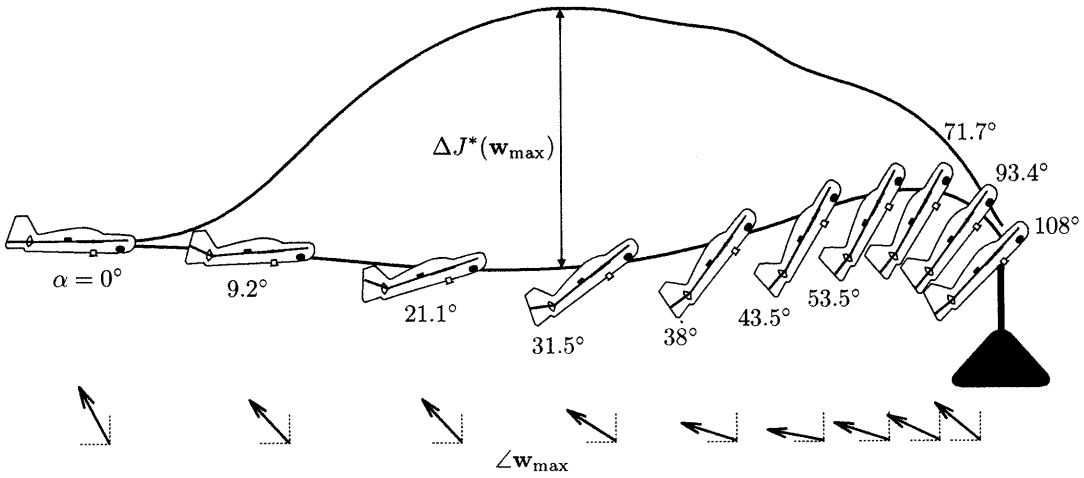


Figure 5-4: The relative cost of a unit wind-gust disturbance applied in the worst possible direction. The worst direction is found through a line search in simulation, shown in the bottom arrows. We see that the controller is least sensitive at the beginning and end of the trajectory and most sensitive in the middle.

5.3 Robustness to Wind-Gusts

There are a variety of possible disturbances that can introduce non-negligible trajectory perturbations during the execution of a perching maneuver. These can range from aerodynamic disturbances such as wind-gusts, to disturbances originating from damages to the vehicle. These will affect the trajectory of the glider in different ways, e.g. force magnitudes and time-scales can vary significantly depending on the type of disturbance; one may deliver an impulse force, while the other may introduce a constant long-term force or moment on the vehicle. If a description of the disturbance types and their likelihood were available, then one could potentially use this information to quantify the robustness of a given controller against the most probable perturbations. Here we investigate the most natural type of disturbance for our task: disturbances originating from wind-gusts.

Our goal here is to quantify the ability of our controller to reject wind-gust disturbances in a (locally) optimal cost-to-go sense. Another words, we attempt to evaluate the 'degree of controllability' along a perching trajectory (given our physical actuator limitations) by performing a sensitivity analysis of the optimal cost-to-go function, which can be compactly written as $\mathbf{x}^T[n]S[n]\mathbf{x}[n]$, for a given 'worst-case' wind-gust disturbance. This is done independently for each time-step of the controller, which allows us to compare relative costs for different wind-gust magnitudes and time-scales along the trajectory.

For simplicity we assume that only translational (i.e., non-rotational) wind-gusts perturb our dynamics. We first perturb the dynamics at each control time step by adding a unit

wind-gust directly to the computation of lift and drag forces to both the wing and elevator.

$$\mathbf{V}_w = \mathbf{V}_w^0 + \mathbf{w} \tag{5.6}$$
$$\mathbf{V}_e = \mathbf{V}_e^0 + \mathbf{w}, \tag{5.7}$$

where $\mathbf{w}$ was chosen to be a unit gust disturbance lasting one control time step, i.e., 22ms. The worst case unit wind-gust is found by performing a line search over all possible angles of attack for the added gust with respect to the optimal cost-to-go value at the next time step of the discrete controller.

$$\Delta J^*(\mathbf{w}_{\max}, n) = (\mathbf{x}^w[n+1] - \mathbf{x}^0[n+1])^T S[n+1](\mathbf{x}^w[n+1] - \mathbf{x}^0[n+1]) \tag{5.8}$$

where the $\mathbf{x}^w$ indicates the wind disturbed state and $\mathbf{x}^0$ indicates the undisturbed state (i.e., the nominal state). Figure 5-4 shows the relative magnitude of the optimal cost-to-go after perturbing the dynamics with a worst-case unit wind-gust for a single time-step. We can see that the controller is relatively insensitive to gusts at the beginning and end of the trajectory. Intuition tells us that a gust disturbance at the beginning of the trajectory allows sufficient time for the controller to effectively mitigate its effects on the final outcome. The end of the trajectory, however, is less sensitive because of the relatively long time-scale necessary for a wind-velocity disturbance to integrate into a sufficiently large position disturbance (which have by far the largest final costs). This time-scale difference causes most of the disturbances to affect pitch in the short term, with relatively little effect on position. In fact, it is this same time-scale difference that is exploited in feedback linearization controllers described in section 5.2. This disturbance integration time-scale is in its most critical point during the middle of the trajectory, where wind-disturbances have enough time to integrate into substantial position errors and the control system hasn't sufficient time to effectively mitigate its effects on the final outcome.

5.4 Discussion

The cost function used to design the linear feedback controller directly influences the shape of the curve shown in Figure 5-4. In our case, position errors were penalized a factor of 100 times more than velocities, resulting in a curve that emphasizes sensitivity to position. The correct cost function for perching can certainly be the focus of much research in itself, e.g., in determining the right trade-off between accuracy, final velocity, final pitch, etc. In our case, the cost was chosen based on manual tuning through trial and error, attempting to qualitatively satisfy a tradeoff between goal accuracy (within a few centimeters) and minimizing landing speed. For the case of our glider, which has no propulsion, stopping short of the goal means catastrophic failure, therefore, accuracy is emphasized much more than landing speed. In practice, however, the addition of an onboard power plant will also play a role in the choice of the cost-function, having to balance energy use, landing speed, in addition to landing accuracy. Birds are almost certainly evaluating this tradeoff constantly, taking into account wind conditions, etc. Our goal here is to begin to understand the consequences of this trade-off in a relatively simple system.

The results presented here are limited to planar, unit magnitude, short duration (one control time-step) analysis. In practice, as we transition to outdoor flight, these analyses will necessarily need to include rotational as well as non-planar disturbances, with varying magnitudes and durations. The robustness evaluation procedure presented here easily lends itself to these more complicated disturbance regimes. For example, given a realistic class of gust disturbance models with varying time scales, we can modify the procedure to simulate the full nonlinear dynamics over the whole time horizon, allowing us to exactly evaluate the performance of the control system without using an estimate of the linearized dynamics.

Chapter 6

Perching with Flapping Wings

The results presented in chapter 5.1 show that one of the advantages of a thrust capable plane is its ability to demonstrate more control authority over a larger portion of the perching trajectory. Here we investigate the dynamics of flapping wings, which have the capability of generating thrust while providing an opportunity to manipulate the air at a finer level through dynamic wing movements, potentially recapturing some of the energy lost in the unsteady wake of the wing. Birds also have the ability to maintain control authority as they approach a perch by flapping their wings. This is done to maintain airspeed over their wings and tail which allows them to maneuver as they come to a stop. Moreover, during perching, flapping wings provide the ability to slow the vehicle down much faster than would otherwise be possible using fixed-wings. Because the drag force is proportional to velocity squared, the forward flap generates an increase in braking force whose magnitude is much larger than the decrease in force during the backward flap, effectively creating a stronger average braking drag force throughout the maneuver. In these next sections, I present a hardware design for a simple flapping-wing version of our glider, describe a simple flapping-wing model for perching using flat-plate theory aerodynamics, and show our initial success in perching with flapping wings on real hardware.

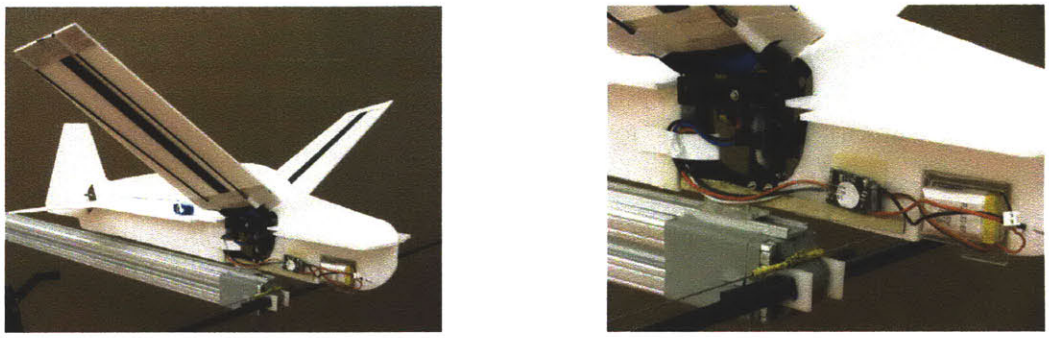


Figure 6-1: A flapping-wing airplane based on our perching glider design, weighing in at just under 170g. The addition of a uni-directional degree of freedom allows speed-control of the flapping frequency, which can exceed 7 Hz.

6.1 A Simple Flapping-Wing Airplane

Using our fixed-wing glider (Figure 3-1) as a basis for design, we constructed a flapping-wing capable aircraft with a single degree of freedom at the wing-joints in order to mimick one of the most fundamental capabilities of birds. Our design is shown in Figure 6-1. This flapping-wing airplane can flap its wings at a maximum frequency that can exceed 7 Hz and uses a transmission that is powered by a small 300 mah lithium polymer battery. The geometry of the airplane is the same as that for the perching glider and the only capability that was added was a uni-directional degree of freedom that allows speed control of the flapping frequency. This flapping-wing airplane weighs in at just under 170g. It was designed with simplicity in mind, that is, the goal was to take a small step in the direction of biomimetic flight design while keeping the problem as tractable as possible, since our strong belief is that the simplicity and tractability of our perching glider design played a large role in allowing us to acquire a deep understanding about it's dynamics and control.

6.2 Quasi-Steady Flapping Flat-Plate Model

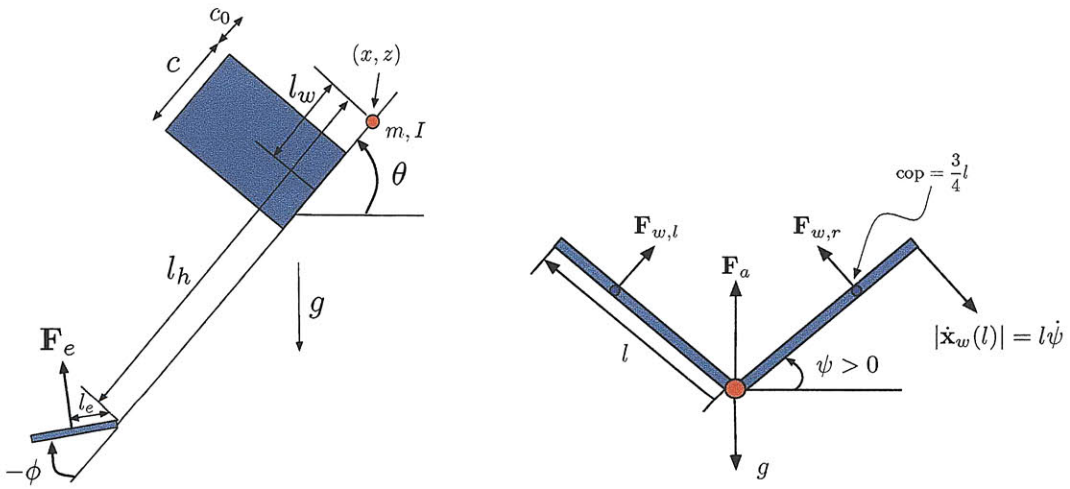


Figure 6-2: Flapping Flat-Plate Model. (Left) Side body view of the vehicle, with positive left wing dihedral. (Right) Front view along the fuselage at the same configuration. Here ψ is positive for the left wing (shown), and negative for the right wing.

Here we present a simplified model of a rigid, flapping-wing airplane based on the flat-plate aerodynamics of our glider (see section 3.1.2). Although this model is based on quasi-steady, thin flat-plate assumptions, it serves as a starting point for control system design and preservers the fundamental structure of the control problem. The aircraft model follows that of the flat-plate glider presented in Section 3.1.3 with an additional degree of freedom for the wing dihedral, i.e., the flapping angle ψ. This additional degree of freedom generates extra velocity terms in our model which we discuss below. The free body diagram of our model is shown in Figure 6-2.

We first derive the equations of motion for a fixed-wing glider with a dihedral on the wings. As we will show, the added dihedral changes our effective angle of attack from that of the fixed-wing model presented in section 3.1.3. Let us determine the normal unit vector of a single wing as a function of glider pitch θ and wing dihedral ψ. Assume the wing's coordinate system is initially aligned with the world coordinate system (roll, pitch, yaw are all zero). The unit vector **j** points toward the leading edge, and **i** points away from the wing tip for the left wing, and towards the wing tip for the right wing. The initial unit normal wing vector is $\mathbf{n}_{w_0} = [0\ 0\ 1]^T$. Following the roll, pitch, yaw rotation convention (Euler fixed X, Y, Z rotation), the normal unit wing vector $\mathbf{n}_w$ can be written in terms of fuselage pitch θ and wing dihedral ψ as:

$$\mathbf{n}_w = \begin{bmatrix} c_\psi & 0 & s_\psi \\ 0 & 1 & 0 \\ -s_\psi & 0 & c_\psi \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\theta & -s_\theta \\ 0 & s_\theta & c_\theta \end{bmatrix} \mathbf{n}_{w_0} = \begin{bmatrix} s_\psi c_\theta \\ -s_\theta \\ c_\psi c_\theta \end{bmatrix}. \tag{6.1}$$

Similarly, the normal unit fuselage vector can be found to be $\mathbf{n}_f = [0\ -s_\theta\ c_\theta]^T$. The angle of attack, defined as the inner angle between the velocity vector and its projection onto the wing, can be written as the complement of the angle between the wing's velocity vector $\dot{\mathbf{x}}_w$ and the negative of the wing's unit normal vector

$$\alpha = \frac{\pi}{2} - \cos^{-1}\left(\frac{-\mathbf{n}_w^T \dot{\mathbf{x}}_w}{||\mathbf{n}_w|| ||\dot{\mathbf{x}}_w||}\right), \tag{6.2}$$

$$= \sin^{-1}\left(\frac{-\mathbf{n}_w^T \dot{\mathbf{x}}_w}{||\dot{\mathbf{x}}_w||}\right). \tag{6.3}$$

In general, the angle of attack along the wing will be a function of the distance along the chord as well as the distance along the span of the wing. For this simplified dihedral model, we will assume that the aerodynamic forces act at mid-chord and mid-span along the wing, i.e., at $(c/2, s/2)$. The distance of this point from the center of gravity is simply $l_w' = \sqrt{(ls_\psi/2)^2 + l_w^2}$. The wing velocity is then given by (see equation 3.5)

$$\dot{\mathbf{x}}_w = \begin{bmatrix} \dot{x} + l_w'\dot{\theta}s_\theta \\ \dot{z} - l_w'\dot{\theta}c_\theta \end{bmatrix}. \tag{6.4}$$

Recalling the expression for the total aerodynamic force on a flat plate (equation 3.10), the aerodynamic force on the wing can then be written as

$$\mathbf{F}_w = \rho S ||\dot{\mathbf{x}}_w||^2 \sin(\alpha) \mathbf{n}_w, \tag{6.5}$$

$$= \rho S ||\dot{\mathbf{x}}_w|| \left(-\mathbf{n}_w^T \dot{\mathbf{x}}_w\right) \mathbf{n}_w, \tag{6.6}$$

$$= f'_w \mathbf{n}_w. \tag{6.7}$$

Since this aerodynamic force is present on both wings, their spanwise vector components

cancel, giving a total aerodynamic force of

$$\begin{aligned} \mathbf{F}_a &= 2f'_w \cos\psi \mathbf{n}_f, & (6.8)\\ &= f_w \mathbf{n}_f. & (6.9) \end{aligned}$$

The glider dynamics are then given by equations 3.12 through 3.14, where the elevator forces are the same as that for the non-dihedral glider.

We now present a model that includes flapping wings. In general, the wing velocity is a function of the distance along the span of the wing, denoted by s, as well as the distance from the center of mass to a point along the chord of the wing, denoted by r. It is given by

$$\begin{aligned} \dot{\mathbf{x}}'_w(s,r) &= \dot{\mathbf{x}}_w + \mathbf{n}_w s\dot{\psi} \quad \text{(left wing)}, & (6.10)\\ \dot{\mathbf{x}}'_w(s,r) &= \dot{\mathbf{x}}_w - \mathbf{n}_w s\dot{\psi} \quad \text{(right wing)}, & (6.11) \end{aligned}$$

where $\dot{\mathbf{x}}_w$ is given by equation 6.4 with $l_w{}' = \sqrt{(s_\psi s)^2 + r^2}$. The general expression for the total aerodynamic force is given by

$$|\mathbf{F}_w| = \rho \int_{r=c_0}^{c_0+c} \int_{s=0}^{l} |\dot{\mathbf{x}}'_w(s,r)| \left(-\mathbf{n}_w^T \dot{\mathbf{x}}'_w(s,r)\right) ds\, dr \qquad (6.12)$$

where $\mathbf{n}_f$ is the unit vector normal to the fuselage and is given by $\mathbf{n}_f = [s_\theta\ 0\ c_\theta]^T$. For the simple model presented here, we will assume that the aerodynamic forces act at mid-chord and mid-span along the wing. The wing velocity is then given by

$$\begin{aligned} \dot{\mathbf{x}}'_w &= \dot{\mathbf{x}}_w + \mathbf{n}_w s\dot{\psi} \quad \text{(left wing)}, & (6.13)\\ \dot{\mathbf{x}}'_w &= \dot{\mathbf{x}}_w - \mathbf{n}_w s\dot{\psi} \quad \text{(right wing)}, & (6.14) \end{aligned}$$

where $\dot{\mathbf{x}}_w$ is given by equation 6.4 with $l_w{}' = \sqrt{(ls_\psi/2)^2 + l_w^2}$. The aerodynamic force on the wing is again given by 6.5, with the total force given by 6.9.

Let us look at the special case where the vehicle is flapping but has no translational velocity with $\theta = 0°$, i.e., we can imagine the flapping vehicle sitting on a load cell (in still air) and the goal is to compute the forces on the vehicle strictly due to flapping. It turns out that in this case we can exactly evaluate the location of the center of pressure for the flapping plate. Given that the vehicle has no translational velocity, the angle of attack on the wings strictly due to flapping will be 90°, i.e., the forces acting on the wings for this simple model will be strictly due to drag. We can then define the total aerodynamic force acting on the left wing as

$$\begin{aligned} \mathbf{F}_w &= \mathbf{n}_w \frac{1}{2}\rho C_D(\alpha) \int_{s=0}^{s=l} |\dot{\mathbf{x}}_w(s)|^2 dA & (6.15)\\ &= \mathbf{n}_w \rho c \int_{s=0}^{s=l} s^2\dot{\psi}^2 ds & (6.16)\\ &= \mathbf{n}_w \frac{1}{3}\rho c \dot{\psi}^2 l^3 & (6.17) \end{aligned}$$

where A represents wing area, s is a length along the wingspan, and we assume for simplicity that the wing chord c is constant over the span of the wing. Since $\alpha = 90°$, and using our simple flat-plate theory aerodynamics [1], we have $C_D(90°) = 2$. Since the aerodynamic force $\mathbf{F}_w$ is present on both wings, the horizontal components of these two forces cancel out and their vertical components sum to give the total aerodynamic force $\mathbf{F}_a$ acting on the center of mass

$$|\mathbf{F}_a| = 2|\mathbf{F}_w| \cos\psi \tag{6.18}$$

$$= \frac{2}{3}\rho c \dot{\psi}^2 l^3 \cos\psi \tag{6.19}$$

Because the air velocity grows linearly along the span of the wing, we would expect the center of pressure to be closer to the wing tip. We can calculate the location of the center of pressure along the wing by integrating the pressure p along the span of the wing s

$$\text{cop} = \frac{\int_s sp(s)ds}{\int_s p(s)ds} \tag{6.20}$$

For a small area dA along the span of the wing, we can write the magnitude of the aerodynamic force as

$$|\mathbf{F}_w^{dA}| = \rho|\dot{\mathbf{x}}_w(s)|^2 dA \tag{6.21}$$

Given $|\dot{\mathbf{x}}_w(s)| = s\dot{\psi}$, the pressure along the area dA is then

$$p(s) = \rho s^2 \dot{\psi}^2 \tag{6.22}$$

Using equation 6.20, the center of pressure location is then given by

$$\text{cop} = \frac{\int_{s=0}^{l} s(\rho s^2 \dot{\psi}^2)ds}{\int_{s=0}^{l} (\rho s^2 \dot{\psi}^2)ds} \tag{6.23}$$

$$= \frac{3}{4}l \tag{6.24}$$

6.3 Open Loop Control

Our first attempts at landing on a perch with our flapping-wing airplane have been with hand-tuned open-loop controllers, both in simulation and on real hardware. Figure 6-3 shows a successful perching maneuver using a hand-tuned flapping controller in simulation. The control parameterization, for both simulation and real hardware, is based on two deflections of the elevator with fixed flapping frequency. Figure 6-4 shows a sequence of stills from a high speed video of a successful perching maneuver attempted with one of our hand tuned controllers that was obtained by defining two elevator deflection points along the trajectory: one where the elevator attempts to pitch the aircraft nose-up, and a second deflection that

[1] $C_D = 2$ assumes only flat-plate theory aerodynamics. In reality, other effects such as leading edge vortices will generate much higher force coefficients on flapping wings.

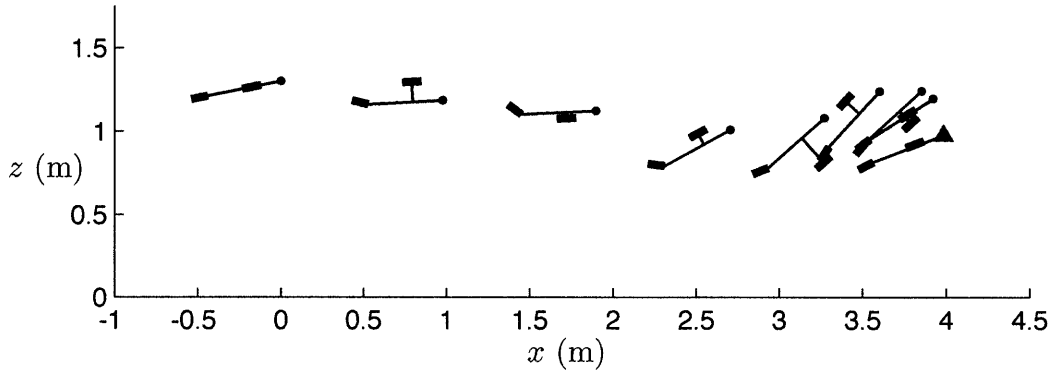


Figure 6-3: A simulation sequence of an open-loop trajectory for our flapping wing airplane. As the vehicle approaches the perch, the last forward flap significantly slows down the vehicle.

attempts to pitch the aircraft nose-down as it prepares to land. The wing angles were reset to zero after each attempted trial, and the flapping frequency was fixed at approximately 4Hz and was started as soon as the aircraft entered the motion capture space. The video for the successful test run shown in Figure 6-4 suggests an appreciable braking capability provided from the forward flap of the wings right before landing. Experimentally, we found that even though our initial conditions were kept fairly constant and our control sequence kept unchanged, the behavior of the trajectory varied from trial to trial, presumably because small deviations in initial conditions were amplified by the complex dynamics of flapping-wings and/or the added dynamics of a flapping transmission. Our hope is to use this new experiment as a platform for testing hypothesis on avian landing maneuvers as well as push the envelope on landing with enhanced stopping performance and precision through the added control-authority provided by flapping-wings.

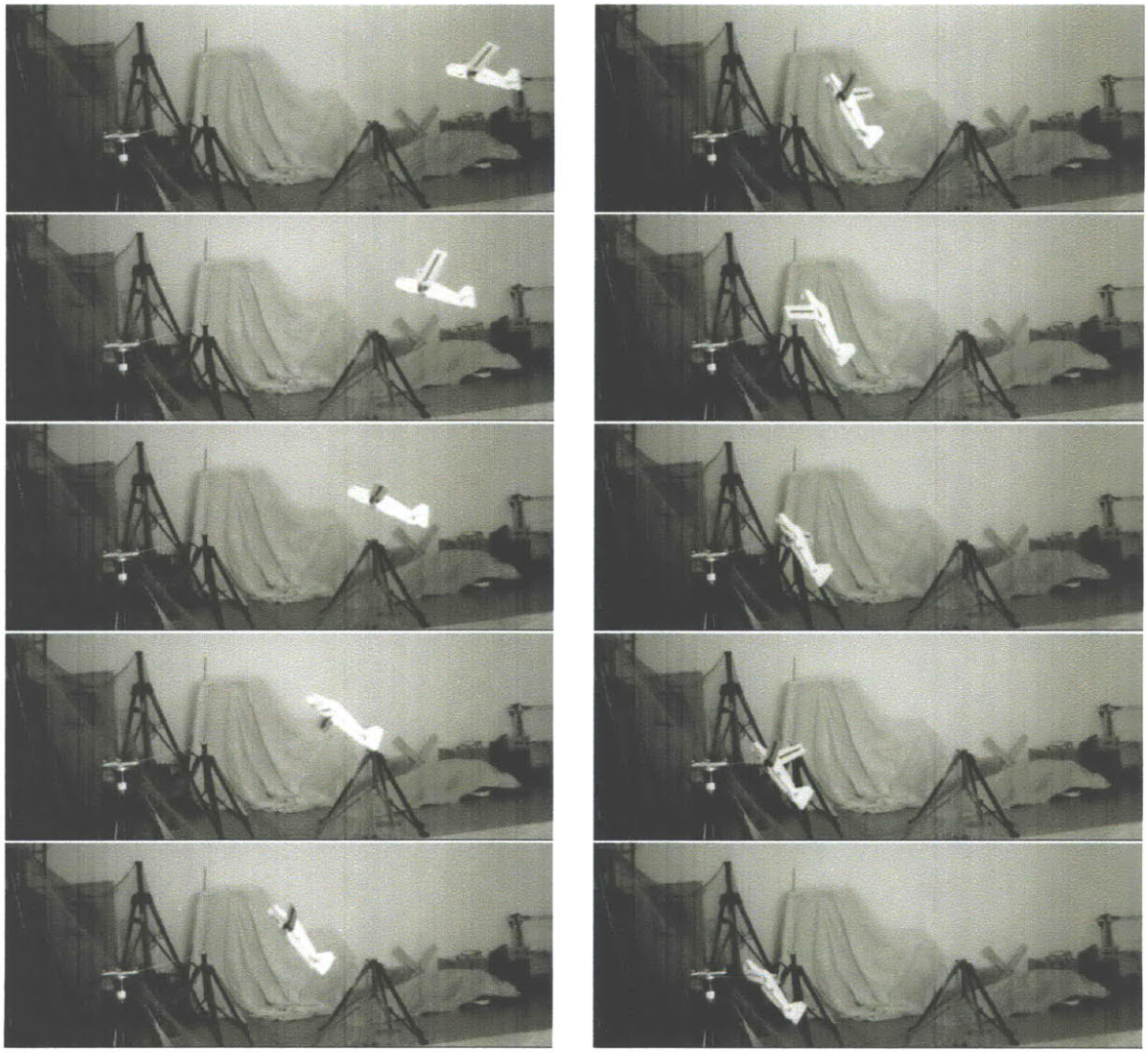


Figure 6-4: A sequence of stills from a hand-tuned open-loop perching maneuver using our flapping-wing airplane. The video suggests an appreciable braking capability with the forward flap right before landing, which coincides with the expected behavior based on our flat-plate aerodynamic model.

Chapter 7

Conclusions and Future Work

Optimal control is a powerful framework for solving highly underactuated, non-linear control problems. The perching glider presented in this thesis requires high-performance control to accurately execute a point landing despite having highly non-linear and heavily underactuated dynamics. One of the control challenges for our perching glider stems from the fact that there is one input controlling seven state variables, which exhibit weak coupling as the vehicle speed decreases near the goal. Moreover, sign dependent sign changes between the input and the system response pose yet another challenge. Here we show, on real hardware, that using only an approximate model of the post-stall aerodynamics along with principled tools in optimal control, even a kinematically simple representation of a bird (our glider) is capable of exploiting post-stall pressure drag to robustly execute a dynamic avian-like perching maneuver. Although the aerodynamics of this landing maneuver are quite complex, even rough approximate models are sufficient for good landing control. This potentially supports the notion that birds don't solve Navier-Stokes.

The importance of the results presented in this thesis are as follows: 1) Using our benchmark perching experiment consisting of a real-time data-aquisition and control architecture interfaced with a motion capture arena, we were able to model the aerodynamics (through post-stall) for supermaneuverable perching, using a conventional fixed-wing glider in free flight. 2) Although the dynamics of the maneuver are highly nonlinear (especially during post-stall), a local feedback controller, along with an approximate model of the nonlinear aerodynamics, was sufficient to develop a perching control system whose performance is unmatched by any other unmanned aerial vehicle (UAV). 3) Using local feedback stabilized trajectory libraries and reasoning about their region of stability, we were able to expand the operating envelope of our control system. 4) Our robustness analysis suggests that our locally optimal perching control solutions are fairly robust to one of the most common disturbances, i.e., wind-gusts, even though control authority diminishes as the vehicle approaches the goal. 5) We designed a flapping-wing airplane that suggests, in simulation and on real hardware, an enhanced braking capability provided by flapping wings. 6) Using a free-flight low-speed wind-tunnel (see section 7.1), we were able to acquire real images of the post-stall aerodynamics during perching, illustrating the dominant vortex street wake.

Our studies have also brought to light some important problems that will need to be addressed as we plan to transition to outdoor flight:

- **Transverse Linearization**: The dependence on time of our local control solutions must be relaxed. A well known approach to this problem is that of using a transverse linearization around the nominal trajectory [Shiriaev et al., 2009]. In effect, this type of linearization allows the controller to index the control through the dynamics that are transverse to the local trajectory, using an appropriate indexing variable, e.g., one that is monotonically decreasing over the trajectory.

- **Wind Gust Analysis Over Disturbance Models**: In practice, as we transition to outdoor flight, these analyses will necessarily need to include rotational as well as non-planar disturbances, with varying magnitudes and durations. Given a realistic class of gust disturbance models with varying time scales, we can modify the procedure to simulate the full nonlinear dynamics over the whole time horizon, allowing us to exactly evaluate the performance of the control system without using an estimate of the linearized dynamics.

- **Cost Function Identification**: Quantify the advantages of certain cost functions over others. For example, one that emphasizes agility over accuracy, or visa-versa.

- **Online LQR-Trees**: We can extend the LQR-Trees algorithm to use information from real experiments to guide its search. Once we have computed a control system offline, we can add new trajectories and adjust basins using experiments on the real glider. Instead of simulating, a failed trajectory on the real glider could be used as an indicator of where a new nominal trajectory is needed. If done enough times, the basin 'gaps' would essentially be filled in using the information from trials on the real glider.

In addition, the performance of our control system will always depend on the fidelity of the model used to design it. We discuss our initial efforts and future directions with modeling these dynamics in the following section.

7.1 Acquiring Higher Fidelity Models Through Flow-Visualization

During a perching maneuver, our fixed-wing glider easily exceeds an angle of attack where post-stall aerodynamics dominate. These aerodynamics are characterized by unsteady pressure drag and vortex streets that combine to quickly decelerate the vehicle in a short distance. In order to help characterize these post-stall aerodynamics, one of our goals is to design an experiment that will allow us to visualize, and potentially sense, these complicated air flow structures during the execution of the perching maneuver. Our primary interest in visualizing and sensing the flow is primarily twofold: 1) to use this information in the development of higher-fidelity aerodynamic models to be used for control design, and 2) to potentially sense and use the state of the airflow in a real-time control system.

For these experiments we designed and built a full-scale, low speed, free-flight, smoke-visualization wind tunnel (shown in Figure 7-2), which we have used to capture real images of the airflow for our perching glider. This 4'x6'x18' wind tunnel is wood constructed and

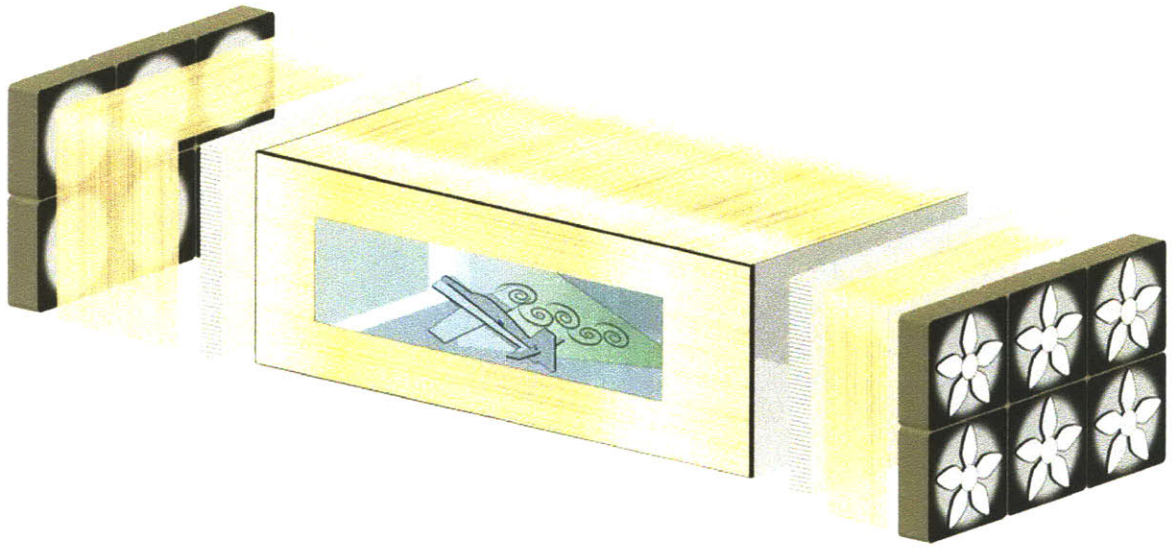


Figure 7-1: Component illustration of wind-tunnel. Our full-size wind tunnel is made up of a 4'x6'x12' test section that allows the glider to fly freely for visualizing the airflow during perching. There are two honey-comb filters that separate an intake and exhaust section, where the air is controlled using 12 Vari-AC controlled box fans.

contains intake and exhaust sections that pull and push the air through the tunnel. The intake consists of six speed-controlled box fans that pull the air through a 4'x6'x4' hollow settling section before going into a honeycomb filter, sandwiched between two wire meshes, used to straighten the airflow, which will have speeds on the order of 1m/s. The exhaust consists of a similar design. A 3'x8' glass window attached to the side gives a full view of the inside of the tunnel while running experiments. During free-flight tests, an automated launcher is used to remotely launch the plane. Small doors on the opposite side of the window give the experimenter access to the inside of the tunnel for experimental setup.

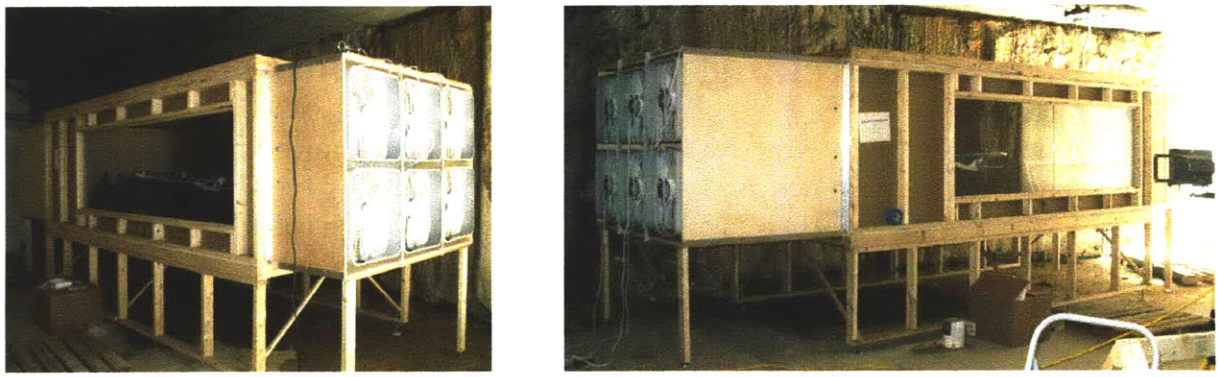


Figure 7-2: Actual pictures of our free flight wind-tunnel.

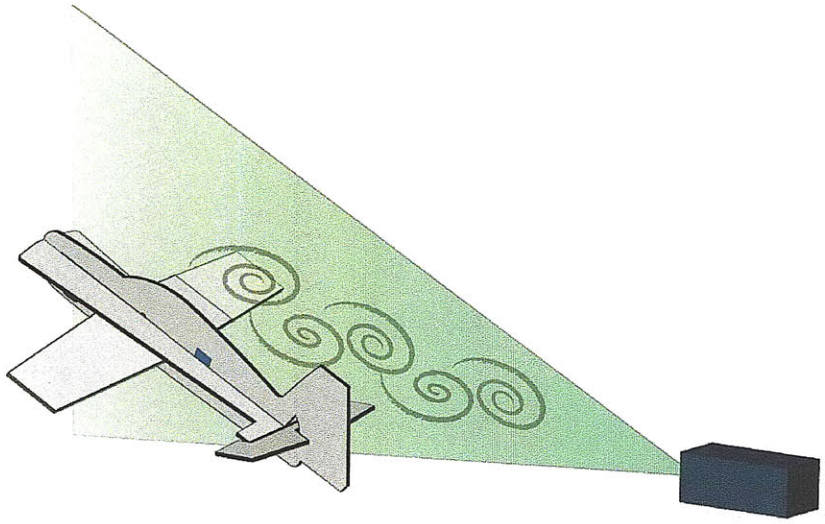


Figure 7-3: Shown is an illustration of our experimental apparatus inside the wind-tunnel. A 2-watt laser is planarized and used to illuminate a 2-dimensional cross section of the airflow over the glider's wing, which is seeded with a titanium-tetrachloride solution used to generate smoke.

In order to obtain our flow visualization images, we seed the leading edge of the glider wing with a TiCl4 chemical solution which reacts with the ambient moisture in the air to generate smoke. As the glider is launched inside our wind tunnel, we use a planarized laser sheet to visualize the airflow over a two-dimensional cross section of the glider's wing. The basic setup is illustrated in Figure 7-3. Using this approach, we have able to acquire real images of the post-stall aerodynamics during the perching maneuver. Figure 7-4 shows a set of stills taken during the high angle of attack transition. The vortex street is clearly visible and its structure is surprisingly clean. Our hope is that in the future we will be able to seed the air with small particles and have the ability to track these particles using a particle imaging velocimetry (PIV) algorithm, which will provide information about the velocity field of the flow. This information will be useful in designing higher-fidelity models as well as real-time air flow sensing for control.

7.2 Outdoor Perching with Flexible Flapping Wings

Radio controlled ornithopters have existed in the hobbyist community for a number of years and their simple yet highly practical designs have already begun to serve as useful research platforms. Using these designs as a starting point, we have designed a mechanical bird named Phoenix (Figure 7-5) which is robust to crashes, easy to repair, and (most importantly) is capable of carrying a large payload consisting of the onboard computation, sensing, and power needed for autonomous outdoor flight. Although we are also conducting flapping control experiments with off-board computation in an indoor motion capture environment, outdoor flight avoids constraints on air space and onboard computation avoids large feedback delays inherent with off-board computation. Our hope is that in the future we will be able

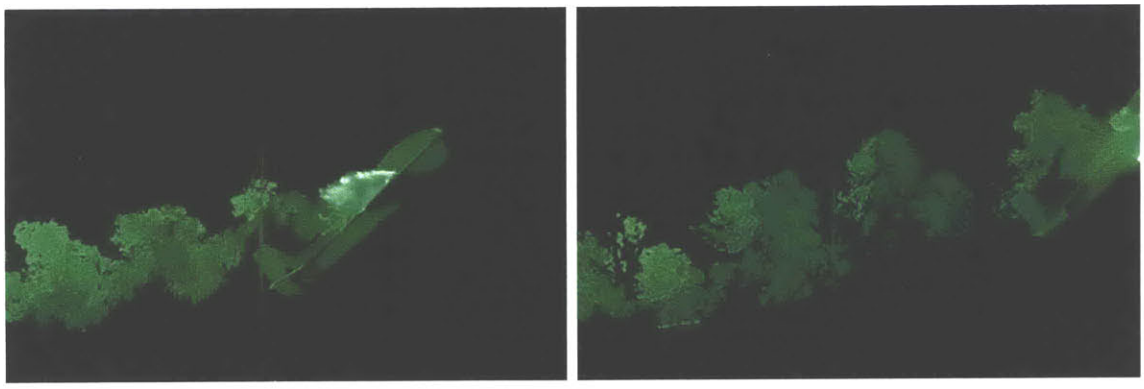


Figure 7-4: Flow visualization stills of our free flight perching experiments. One can clearly see the flow separation behind the wing as well as the vortex street structures present in the wake.

to use this platform to match the agility of a real bird and eventually be able to land on a branch of a tree as gracefully as a real bird.

7.3 Concluding Remarks

Here we presented a successful demonstration of a model based controller that was able to execute a highly dynamic maneuver on a highly nonlinear and underactuated robot. However, as our robots get more and more complicated, e.g., flexible flapping-wings in real outdoor environments, it is very likely that our model approximations will start to suffer. The hope is that for even the most complicated robots, using even a very rough approximation of the dynamics will allow us to bootstrap the control design procedure. Moreover, it is almost certain that real-time control adaptation will play a key role for high performance robots.

Figure 7-5: Our Phoenix ornithopter. This robotic bird carries an embedded computer running linux in its belly, with gyros, accelerometers, magnetometers, and sonar sensors all on board. Communication is done through an on-board wi-fi access point. Design by Zack Jackowski.

Bibliography

[Abbeel et al., 2007] Abbeel, P., Coates, A., Quigley, M., and Ng, A. Y. (2007). An application of reinforcement learning to aerobatic helicopter flight. In *In Advances in Neural Information Processing Systems 19*, page 2007. MIT Press.

[Alcorn et al., 1996] Alcorn, C. W., Croom, M. A., Francis, M. S., and Ross, H. (1996). The x-31 aircraft: advances in aircraft agility and performance. *Progess in Aerospace Sciences*, 32:377–413.

[Amitay and Parekh, 2004] Amitay, M. and Parekh, D. E. (2004). Active flow control on the stingray uninhabited air vehicle: Transient behavior. *AIAA Journal*, 42.

[Anderson et al., 2009] Anderson, M. L., Perry, C. J., Hua, B. M., Olsen, D. S., Parcus, J. R., Pederson, K. M., and Jensen, D. D. (2009). The sticky-pad plane and other innovative concepts for perching uavs. In *47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition*. AIAA.

[Bayrakar and Feron, 2008] Bayrakar, S. and Feron, E. (2008). Experiments with small helicopter automated landings at unusual attitudes. Technical report, Ga. Tech.

[Bertsekas, 2000] Bertsekas, D. P. (2000). *Dynamic Programming and Optimal Control*. Athena Scientific, 2nd edition.

[Blauwe et al., 2007] Blauwe, H. D., Bayraktar, S., Feron, E., and Lokumeu, F. (2007). 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*. AIAA.

[Bowman et al., 2007] Bowman, J., Sanders, B., Cannon, B., Kudva, J., Joshi, S., and Weisshaar, T. (2007). Development of next generation morphing aircraft structures. In *48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. AIAA.

[Burridge et al., 1999] Burridge, R. R., Rizzi, A. A., and Koditschek, D. E. (1999). Sequential composition of dynamically dexterous robot behaviors. *The International Journal of Robotics Research*, 18(6):534–555.

[Cadogan and Smith, 2004] Cadogan, D. and Smith, T. (2004). Morphing inflatable wing development for compact package unmanned aerial vehicles. In

IAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference. AIAA, AIAA.

[Chen, 1998] Chen, C.-T. (1998). *Linear System Theory and Design.* Oxford University Press, USA, 3rd edition.

[Collins et al., 2005] Collins, S. H., Ruina, A., Tedrake, R., and Wisse, M. (2005). Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307:1082–1085.

[Command, 1998] Command, U. N. A. T. (1998). *Out-of-Control Flight, Flight Procedures - Flight Support Lecture Guide.* NAS, Corpus Christi, p-1250 07/24/07 edition.

[Cook, 1997] Cook, M. (1997). *Flight Dynamics Principles.* Butterworth-Heinemann.

[Cooper and Jr., 1969] Cooper, G. E. and Jr., R. P. H. (1969). The use of pilot rating in the evaluation of aircraft handling qualities. In *AGARD Report 567.* AGARD.

[Cory, 2008] Cory, R. (2008). Perching with fixed wings. Master's thesis, Massachusetts Institute of Technology (MIT).

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

[Cory and Tedrake, 2008] Cory, R. and Tedrake, R. (2008). Experiments in fixed-wing uav perching. In *AIAA Guidance, Navigation, and Control Conference.* AIAA.

[de Marmier and Wereley, 2003] de Marmier, P. and Wereley, N. M. (2003). Morphing wings of a small scale uav using inflatable actuators for sweep control. In *44th AIAA/ASME/ASCE/AHS Structures, Structural Dynamics, and Materials Conference.* AIAA.

[el Hak, 2001] el Hak, M. G. (2001). Flow control: The future. *Journal of Aircraft*, 38(3):402–418.

[Frank et al., 2007] Frank, A., McGrew, J. S., Valenti, M., Levine, D., and How, J. P. (2007). Hover, transition, and level flight control design for a single-propeller indoor airplanes. In *AIAA Guidance Navigation Navigation and Controls Conference and Exhibit.*

[Frazzoli et al., 2002] Frazzoli, E., Dahleh, M. A., and Feron, E. (2002). Real-time motion planning for agile autonomous vehicles. *Journal of Guidance, Control, and Dynamics.*

[Gal-Or, 1990] Gal-Or, B. (1990). *Vectored Propulsion, Supermaneuverability and Robot Aircraft.* Springer-Verlag.

[Gallaway, 1985] Gallaway, C. R. (1985). Aerodynamics perspective of supermaneuverability. In *AIAA 3rd Applied Aerodynamics Conference.*

[Gavrilets, 2003] Gavrilets, V. (2003). *Autonomous Aerobatic Maneuvering of Miniature Helicopters.* PhD thesis, MIT.

[Gavrilets et al., 2001] Gavrilets, V., Frazzoli, E., Mettler, B., Piedmonte, M., and Feron, E. (2001). Aggressive maneuvering of small autonomous helicopters: A human-centered approach. *The International Journal of Robotics Research*, 20(10):795–807.

[Gavrilets et al., 2002] Gavrilets, V., Martinos, I., Mettler, B., and Feron, E. (2002). Control logic for automated aerobatic flight of a miniature helicopter. In *AAIAA Guidance, Navigation, and Control Conference and Exhibit*. AIAA.

[Green and Oh, 2006] Green, W. E. and Oh, P. Y. (2006). A fixed-wing aircraft for hovering in caves, tunnels, and buildings. In *IEEE American Control Conference (ACC)*. ACC.

[Gursul et al., 2007] Gursul, I., Wang, Z., and Vardaki, E. (2007). Review of flow control mechanisms of leading-edge vortices. *Progress in Aerospace Sciences*, 43(7-8):246–270.

[Harris and Black, 1996] Harris, J. J. and Black, G. (1996). F-22 control law development and flying qualities. In AIAA, editor, *AIAA Atmospheric Flight Mechanics Conference*. AIAA, AIAA.

[Herbst, 1983] Herbst, W. B. (1983). Supermaneuverability. In *AFOSR Proc. of the Workshop on Unsteady Separated Flow*.

[Hoburg and Tedrake, 2009] Hoburg, W. and Tedrake, R. (2009). System identification of post stall aerodynamics for uav perching. In *AIAA Infotech@Aerospace Conference*. AIAA.

[Ivanco and Scott, 2007] Ivanco, T. G. and Scott, R. C. (2007). Validation of the lockheed martin morphing concept with wind tunnel testing. In *48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. AIAA.

[Kalman, 1960] Kalman, R. (1960). Contributions to the theory of optimal control. *Boletin de la Sociedad Matematica Mexicana*, 5:102–119.

[Kosman and Lentink, 2009] Kosman, S. and Lentink, D. (2009). Roboswift: Bio-inspired morphing-wing micro aerial vehicle, http://www.roboswift.nl/.

[Kwakernaak and Sivan, 1972] Kwakernaak, H. and Sivan, R. (1972). *Linear Optimal Control Systems*. John Wiley & Sons, Inc.

[LaValle, 2006] LaValle, S. (2006). *Planning Algorithms*. Cambridge University Press. available at http://planning.cs.uiuc.edu/.

[Lentink et al., 2007] Lentink, D., Mller, U., Gestel, W. v., Stamhuis, E., Videler, J., Kat, R. d., Veldhuis, L., Henningsson, P., Hedenstrm, A., and Leeuwen, J. v. (2007). How swifts control their glide performance with morphing wings. In *Abstracts of the Annual Main Meeting of the Society for Experimental Biology*.

[Levine, 1996] Levine, W. S. (1996). *The Control Handbook*. CRC Press, 1 edition.

[Lukens, 2008] Lukens, J. M. (2008). Wing mechanization design and analysis for a perching micro air vehicle. In *49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. AIAA.

[Lussier-Desbiens and Cutkosky, 2009] Lussier-Desbiens, A. and Cutkosky, M. (2009). Landing and perching on vertical surfaces with microspines for small unmanned air vehicles. In *UAV'09, 2nd International Symposium on Unmanned Aerial Vehicles*.

[Menon et al., 1987] Menon, P., Badgett, M., and Walker, R. (1987). Nonlinear flight test trajectory controllers for aircraft. *Journal of Guidance, Control, and Dynamics*, 10:67–72.

[Mettler, 2002] Mettler, B. (2002). *Identification, Modeling and Characteristics of Miniature Rotorcraft*. Kluver Academic Publishers, Boston, MA.

[MIL-F-8785C, 1980] MIL-F-8785C, M. S. (1980). Flying qualities of piloted airplanes. MIL-F-8785C.

[Munos and Moore, 1998] Munos, R. and Moore, A. (1998). Barycentric interpolators for continuous space and time reinforcement learning. In Kearns, M. S., Solla, S. A., and Cohn, D. A., editors, *Advances in Neural Information Processing Systems*, volume 11, pages M. S. Kearns, S. A. Solla, and D. A. Cohn. NIPS, MIT Press.

[NASA, 2009] NASA (2009). X-29 fact sheet. [online] `http://www.nasa.gov/centers/dryden/news/FactSheets/FS-008-DFRC.html`.

[Ng et al., 2004] Ng, A. Y., Kim, H. J., Jordan, M. I., and Sastry, S. (2004). Autonomous helicopter flight via reinforcement learning. In *In International Symposium on Experimental Robotics*. MIT Press.

[Nise, 2003] Nise, N. S. (2003). *Control Systems Engineering*. Wiley, 4th edition.

[Pamadi, 2004] Pamadi, B. (2004). *Performance, Stability, Dynamics, and Control of Airplane*. AIAA, second edition.

[Prandtl, 1904] Prandtl, L. (1904). Uber flussigkeitsbewegung bei sehr kl ei ner reibung. In *Proceedings of the Third International Math. Congress*.

[Prickett and Parkes, 2001] Prickett, A. L. and Parkes, C. J. (2001). Flight testing of the f/a-18e/f automatic carrier landing system. In *2001 IEEE Aerospace Conference*.

[Reich et al., 2009] Reich, G. W., Wojnar, M. O., and Albertani, R. (2009). Aerodynamic performace of a notional perching mav design. In *AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition*. AIAA.

[Reist and Tedrake, 2010] Reist, P. and Tedrake, R. (2010). Simulation-based lqr-trees with input and state constraints. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*.

[Roberts, 2009] Roberts, J. W. (2009). Motor learning on a heaving plate via improved-snr algorithms. Master's thesis, Massachusetts Institute of Technology.

[Roberts et al., 2009] Roberts, J. W., Cory, R., and Tedrake, R. (2009). On the controllability of fixed-wing perching. In *American Controls Conference (ACC)*.

[Ruben and Svitil, 2007] Ruben, J. and Svitil, K. (2007). Nature: Raptorforce. [DVD].

[Shiriaev et al., 2009] Shiriaev, A. S., Freidovich, L. B., and Gusev, S. V. (2009). Transverse linearization for mechanical systems with several passive degrees of freedom with applications to orbital stabilization. In *American Control Conference (ACC)*.

[Simon and Stubeerud, 1970] Simon, K. and Stubeerud, A. (1970). Duality of linear estimation and control. *Journal of Optimization Theory and Applications*, 6(1):55–67.

[Snell et al., 1992] Snell, S. A., Enns, D. F., and Jr., W. L. G. (1992). Nonlinear inversion flight control for a supermaneuverable aircraft. *Journal of Guidance, Control, and Dynamics*, 15(4).

[Szepesvary and Munos, 2005] Szepesvary, C. and Munos, R. (2005). Finite time bounds for sampling based fitted value iteration. In *International Conference on Machine Learning (ICLM)*.

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

[Tedrake, 2004] Tedrake, R. (2004). *Applied Optimal Control for Dynamically Stable Legged Locomotion*. PhD thesis, Massachusetts Institute of Technology.

[Tedrake, 2009] Tedrake, R. (2009). Lqr-trees: Feedback motion planning on sparse randomized trees. In *Accepted for the Proceedings of Robotics: Science and Systems (RSS)*.

[Tedrake et al., 2009] Tedrake, R., Jackowski, Z., Cory, R., Roberts, J. W., and Hoburg, W. (2009). Learning to fly like a bird. Submitted to Communications of the ACM.

[Tischler, 1996] Tischler, M. B. (1996). *Advances in Aircraft Flight Control*. CRC Press, 1st edition.

[Tucker, 1998] Tucker, V. A. (1998). Gliding flight: Speed and acceleration of ideal falcons during diving and pull out. *The Journal of Experimental Biology*, 201:408–414.

[Turbary, 2010] Turbary (Accessed January, 2010). Checkers: Eurasian eagle owl high-speed landing. [online] `http://www.turbarywoods.co.uk/news/`.

[US-Airforce, 2009] US-Airforce (2009). F-22 raptor fact sheet. [online] `http://www.af.mil/information/factsheets/factsheet.asp?id=199`.

[van Dommelen, 2009] van Dommelen, L. (2009). Stalling wing cross sections. [online] `http://www.eng.fsu.edu/~dommelen/research/airfoil/airfoil.html`.

[Wickenheiser, 2008] Wickenheiser, A. (2008). *Dynamics and Trajectory Optimization of Morphing Aircraft in Perching Maneuvers.* PhD thesis, Cornell University.

[Wikipedia, 2009] Wikipedia (2009). Pugachev's cobra. [online] `http://en.wikipedia.org/wiki/Pugachev's_Cobra`.

[Williams et al., 2009] Williams, D., Quach, V., Kerstens, W., and Buntain, S. (2009). Low-reynolds number wing response to an oscillating freestream with and without feed forward control. In AIAA, editor, *Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition.* AIAA, AIAA.

[Wilson, 2008] Wilson, P. (2008). Speed & angels. [DVD].