Robust post-stall perching with a simple fixed-wing glider using LQR-Trees

Robust Post-Stall Perching with a Simple Fixed-Wing Glider using LQR-Trees

Joseph Moore, Rick Cory, Russ Tedrake

MIT 32-380, Cambridge, MA 02139

E-mail: [jlmoore,rcory,russt]@csail.mit.edu

Abstract. Birds routinely execute post-stall maneuvers with a speed and precision far beyond the capabilities of our best aircraft control systems. One remarkable example is a bird exploiting post-stall pressure drag in order to rapidly decelerate to land on a perch. Stall is typically associated with a loss of control authority, and it is tempting to attribute this agility of birds to the intricate morphology of the wings and tail, to their precision sensing apparatus, or their ability to perform thrust vectoring. Here we ask whether an extremely simple fixed-wing glider (no propeller) with only a single actuator in the tail is capable of landing precisely on a perch *from a large range of initial conditions*. To answer this question, we focus on the design of the flight control system; building upon previous work which used linear feedback control design based on quadratic regulators (LQR), we develop *nonlinear* feedback control based on nonlinear model-predictive control (NMPC) and "LQR-Trees". Through simulation using a flat-plate model of the glider, we find that both nonlinear methods are capable of achieving an accurate bird-like perching maneuver from a large range of initial conditions; the "LQR-Trees" algorithm is particularly useful due to its low computational burden at runtime and its inherent performance guarantees. With this in mind, we then implement the "LQR-Trees" algorithm *on real hardware* and demonstrate a 95 percent perching success rate over 147 flights for a wide range of initial speeds. These results suggest that, at least in the absence of significant disturbances like wind gusts, complex wing morphology and sensing are not strictly required to achieve accurate and robust perching even in the post-stall flow regime.

1. Introduction

Birds consistently outperform man-made aircraft in post-stall flight regimes. Consider the task of a bird landing on a perch: as the animal approaches its target, it flares its wings and tail and orients its entire body to a very high angle of attack. In this post-stall flight, the bird increases the viscous drag by increasing the surface area of the wing exposed to the flow, but it also exploits the pressure drag caused by the separation of the airflow from the wing; these combine to produce large total drag forces and a rapid deceleration[3, 2, 17]. Despite these large forces and rapid deceleration, the birds are still seemingly able to achieve exceptional accuracy in order to position their bodies relative to the perch. In contrast, few man-made aircraft even attempt to fly at such high angles of attack [29], and those that do usually make use of thrust vectoring and a large thrust-to-weight ratio [1]. Even then, these highly maneuverable aircraft, which can effortlessly execute post-stall maneuver in open air, greatly reduce their angle of attack whenever they must carry out a precise landing maneuver. The automatic control system used on an F-18 fighter jet when landing on an aircraft carrier, for instance, maintains an extremely conservative 8.1 degree angle of attack during the landing [29]. Birds, in comparison, easily reach up to 90 degrees angle of attack when landing on the branch of a tree [7].

One primary reason why many man-made aircraft avoid post-stall flight is because these conditions are often considered synonymous with a loss of control authority. When an airfoil transitions in to post-stall flight, the airflow begins to separate from the wing's leading edge, i.e. the air fails to smoothly follow the contour of wing, causing a sharp drop in lift. This is often problematic because, with such a drastic reduction in lift, many aircraft are unable to maintain their desired altitude. Moreover, detached flow is both unsteady and turbulent, severely complicating the effort to model and control the aircraft. Birds, however, seem undaunted by the challenges posed by post-stall flight, and exploit the complicated flow regimes to land quickly [7].

In this paper, we explore the problem of executing a bird-like post-stall perching maneuver from a large range of initial conditions. Our ultimate goal is to enable this capability in conventional fixed-wing UAVs, which are arguably much less sophisticated than birds in their sensory and mechanical flight systems. For this reason, we wish to understand what level of performance is possible with limited hardware but relatively sophisticated feedback control design. In this paper, we deliberately focus on a minimal aircraft design so that the maneuver's success is almost entirely dependent on the performance of the control system - we study an un-powered glider modeled as a flat plate and controlled by a single actuator at the tail. In [31], the authors apply a local time-varying *linear* controller to this glider model and evaluate the controllability of the aircraft during a perching maneuver. Their findings reveal that both controllability and robustness to disturbances are severely limited. To improve on this, here, we investigate *nonlinear* control design techniques. Because our system is of high dimension and extremely underactuated, we find ourselves limited to two nonlinear control design approaches - Nonlinear Model Predictive Control (NMPC) and "LQR-Trees".

The complexity of the aerodynamics during the post-stall flight regime is difficult to

capture in simulations using lumped-parameter models. Therefore although we make use of simulation for a feasibility analysis, we focus on evaluating our controller experimentally on real flight hardware, and demonstrate that for a large set of initial conditions we are able to successfully transfer the aircraft to a small goal region at the perch. Our results reveal that using carefully-designed feedback control, it is in fact possible to achieve the post-stall perching behavior of birds using a small, un-powered UAV, without relying on the complex morphologies or specialized sensing described in [7, 6].

2. Related Work

Over the last few years, the design and control of perching aircraft has received considerable attention from the aerospace and controls communities for their potential in expanding the capabilities of autonomous aerial vehicles. For example, as proposed in [22, 23], the ability to perch and recharge on a power-line can drastically increase flight range and endurance for search and rescue vehicles that would otherwise need to return to base to recharge.

Perhaps the most closely related work, in terms of aerodynamic performance and agility, is the morphing, perching vehicle project from Wickenheiser et al [40, 41]. This vehicle is designed to pitch the body up to obtain extra drag, but also to rotate the wings and tail back down to a low angle of attack to maintain attached flow. As a consequence, the vehicle only exploits a fraction of the available drag, but can make use of more conventional aerodynamic control. Similar work in [19, 30] presents a design for a sectional morphing wing aircraft. The work we report here emphasizes a relatively much simpler mechanical design and uses the entire body and wing to contribute to drag, focusing instead on the problem of post-stall control.

Another approach to developing perching aircraft is through vertical take-off and landing (VTOL) technology. Advances in power plant miniaturization and high power-density batteries have paved the way for high thrust-to-weight ratio hobby airplanes capable of executing VTOL maneuvers [14, 18, 5]. Using a powerful propeller attached to the nose of these small airplanes, one can essentially hover them in place (known as a prop-hang), using the backwash of the propeller over the control surfaces to provide the needed control authority. Although a simple hand-tuned proportional-derivative (PD) control system provides this vehicle with a tremendous region of stability [8] and provides sufficient control for a VTOL perched landing (using proper sensing) [14], it does so at the great expense of speed and energy efficiency. In fact, our glider demonstrates that propulsion is not necessary for high performance perching. Moreover, it can execute the maneuver an order of magnitude faster than its high-propulsion VTOL counterpart.

Recent work on perching on vertical surfaces [12] shares some of our motivation for biologically inspired landings. In their work, the authors equip a fixed-wing glider with a pair of compliant spiny feet attached to the bottom of the wings in order to land and stick to a vertical wall. Similar to a perching maneuver, a vertical surface landing with their vehicle requires a transition to post-stall flight in order to expose these feet to the landing surface. The

focus of this work is different, as the large landing target (a wall) reduces the requirements on sensing and control - a single sonar sensor triggers an elevator deflection when the airplane is within a few meters from the wall. Thereafter, the airplane follows an open-loop ballistic trajectory after which the airplane impacts and sticks to the wall.

Our results previously reported in [9] presented the first known successful demonstration of a post-stall bird-like perching maneuver on a real aircraft. This was motivated primarily by studies carried out on bird flight kinematics during perching, such as [17], [2], and [3], which indicate that birds are exploiting high angles-of-attack in ways conventional aircraft never have. To achieve this bird-like flight behavior, an approximate optimal control design procedure was used to compute a feedback policy over the entire state space, albeit with a very coarse discretization mesh, which allowed our vehicle to successfully land on a perch approximately 4 out 20 trials, for only a single nominal initial condition.

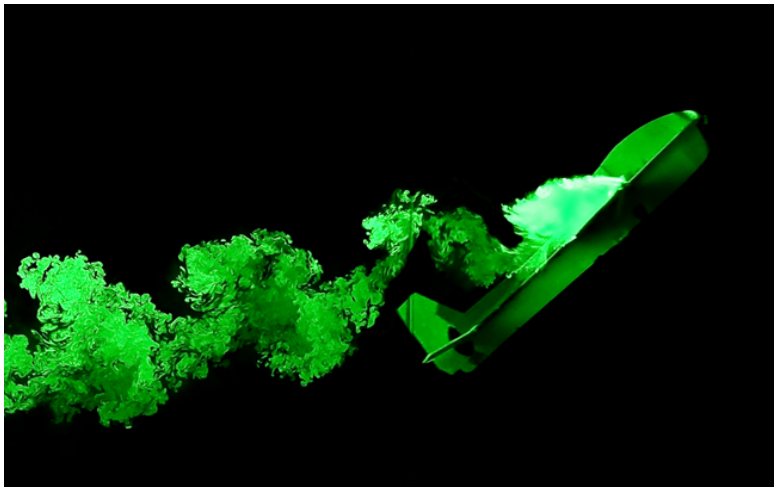
As mentioned previously, in [31], the authors apply a local time-varying linear controller to the glider in [9] and evaluate the controllability of the aircraft during a perching maneuver. This work effectively exposes many of the fundamental challenges which are make the perching problem hard, but the interpretation of the results is potentially limited by the linearization-based analysis.

Here we present a nonlinear control system, initially proposed in [24], which exhibits dramatically improved perching performance and is capable of stabilizing a large range of flight speeds while reliably landing the vehicle within only 6.5 centimeters from the perch. This performance improvement is largely due to an advanced control design procedure which adequately reasons about the underactuated aircraft dynamics over a wide range of initial conditions. A key feature of our work is the experimental results, which confirm that our controller truly does possess the robustness to initial conditions and modeling errors that it promises to provide.

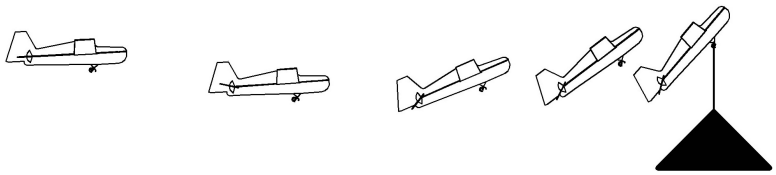
3. Problem Formulation

To parallel post-stall perching in birds, we consider the problem of landing on a string with a $24\frac{1}{2}$ inch wingspan glider (no onboard propulsion) with a $9\ \frac{1}{2}$ inch cord. The size of our aircraft is comparable to that of common rock pigeon (Columbia livia), which has a wingspan ranging between 24 and 28 inches [27]. As shown in Figure 1(b), the glider is launched at a distance of 3.5 meters from the perch with an initial speed of approximately 7 m/s (Re $\approx$ 50,000). It then must decelerate to almost zero velocity and come to rest on the string in a fraction of a second. This too matches well with the initial flow conditions observed in [3], where pigeons (a chord of 8.8 inches) are recorded approaching a perch from 2m away with initial speeds around 4 m/s (Re $\approx$ 55,000) and landing with speeds below 1 m/s.

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 surface, we demand that the glider's landing gear land within 6.5 cm of the string - this is approximately the capture region for our



(a)



(b)

Figure 1. (a) Flow visualization of the air behind the perching glider. The observed vortex shedding is characteristic of such high angle of attack maneuvers. **(b)** A depiction of the glider executing its post-stall perching maneuver. The vehicle begins at approximately 7 m/s and ends at a very high angle of attack.

simple hook landing gear. This post-stall perching strategy is similar to what is employed by the pigeon, which as reported in [3, 7], pitches up to body angles beyond 90 degrees when landing.

4. Experimental Setup

In order to isolate the flight control problem, we designed an indoor experimental facility which mitigates many of the complications of sensing, computation, and repeatability. Our facility is equipped with a Vicon motion capture system consisting of 16 cameras with a combined capture volume of 27 m^3. These cameras track a set of small reflective markers attached to the glider's fuselage and elevator and allow the motion capture system to relay real-time position and orientation information of the glider and elevator, in addition to the perch location, at about 90 frames per second. To protect our aircraft, we hang a large safety net just above the floor under the entire experiment.

Using a custom crossbow launching device, the glider is launched from one end of the capture volume at a range of initial speeds from 6-8 m/s towards a suspended string on the opposite side of the capture volume. Upon launch, the motion capture system begins relaying position and orientation information over Ethernet to the ground control station, a Linux PC.

This ground control station then estimates the relevant vehicle states from the raw motion capture data and uses these states to evaluate the feedback control policy at the same rate as the incoming motion capture data, 90 Hz. Once the control policy is evaluated, the control station sends an elevator velocity command by serial port to the buddy-box interface of a hobby radio transmitter, which then wirelessly transmits the command to the glider's onboard receiver.

5. Vehicle Design

Our experimental glider design was largely inspired by commercially available aerobatic hobby aircraft. To minimize complexity, we control the longitudinal dynamics with a single servo for the elevator, but forgo all other control surfaces and rely on passive stability (and short flight durations) in roll and yaw. The entire glider is made of laser-cut 2.8 mm thick depron foam sheets, carries a GWS four channel micro receiver, a HS-50 Hitec hobby servo and a Full River 250 mah 2-cell lithium polymer battery to actuate the elevator. Including the payload, the aircraft weights 85 g. For ease of fabrication and modeling, the glider's fuselage, tail, and wings are composed of symmetric foam flat plates, i.e., the wings and tail have no camber. The wings have a 9 mm mean chord ($\approx 3\%$ thickness-to-chord ratio), a 8:3 aspect ratio, with a maximum (total surface) lift to drag ratio of approximately 3.5 (at zero elevator angle). These wings are slightly tapered (113 mm root chord, 83 mm tip chord) and are carbon fiber reinforced using a thin 14 mm tape along the span. The total surface area of all lifting surfaces (including the fuselage is 0.102 m^2. for a total wing/surface loading of 0.753 kg/m^2. Six 11 mm reflective markers are 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 less than 6.5 cm.

6. System Identification and Modeling

For feedback control design and analysis, we model our aircraft as a single rigid body subjected to gravitational and aerodynamic forces. Initially, these aerodynamic forces were derived by approximating all lifting surfaces as flat plates. However, to obtain our desired perching performance, we were required to improve this model by using more descriptive functional approximations of the aircraft's aerodynamic coefficients. This was achieved by using the flat plate glider model as a baseline and augmenting the lift and drag coefficients using radial basis functions. Here we describe both the flat plate glider model and its modifications.

6.1. Flat Plate Glider Model

Following [9], our flat plate glider model is based on the aerodynamic coefficients put forth in [34], which are given as

$$C_L(\alpha) = 2\sin(\alpha)\cos(\alpha) \tag{1}$$
$$C_D(\alpha) = 2\sin^2(\alpha), \tag{2}$$

where α is the angle of attack of the lifting surface. As in [9], our model restricts the aircraft dynamics to two dimensions, ignoring contributions from yaw, roll, and three dimensional aerodynamics. This leads to a seven dimensional control differential equation,

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

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

As illustrated in Figure 2(a), we define the unit vectors normal to the control surfaces in the directions of the force vectors as

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

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

$$\mathbf{x}_w = \begin{bmatrix} x - l_w c_\theta \\ z - l_w s_\theta \end{bmatrix}, \quad \mathbf{x}_e = \begin{bmatrix} x - l c_\theta - l_e c_{\theta+\phi} \\ z - l s_\theta - l_e s_{\theta+\phi} \end{bmatrix}, \tag{3}$$
$$\dot{\mathbf{x}}_w = \begin{bmatrix} \dot{x} + l_w \dot{\theta} s_\theta \\ \dot{z} - l_w \dot{\theta} c_\theta \end{bmatrix}, \quad \dot{\mathbf{x}}_e = \begin{bmatrix} \dot{x} + l\dot{\theta} s_\theta + l_e(\dot{\theta}+\dot{\phi}) s_{\theta+\phi} \\ \dot{z} - l\dot{\theta} c_\theta - l_e(\dot{\theta}+\dot{\phi}) c_{\theta+\phi} \end{bmatrix}. \tag{4}$$

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

$$\alpha_w = \theta - \tan^{-1}(\dot{z}_w, \dot{x}_w), \quad \alpha_e = \theta + \phi - \tan^{-1}(\dot{z}_e, \dot{x}_e) \tag{5}$$
$$\mathbf{F}_w = \frac{1}{2}\rho |\mathbf{x}_w|^2 S_w \big(C_L(\alpha_w) + C_D(\alpha_w)\big)\mathbf{n_w}, \tag{6}$$
$$\mathbf{F}_e = \frac{1}{2}\rho |\mathbf{x}_e|^2 S_e \big(C_L(\alpha_e) + C_D(\alpha_e)\big)\mathbf{n_e} \tag{7}$$

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

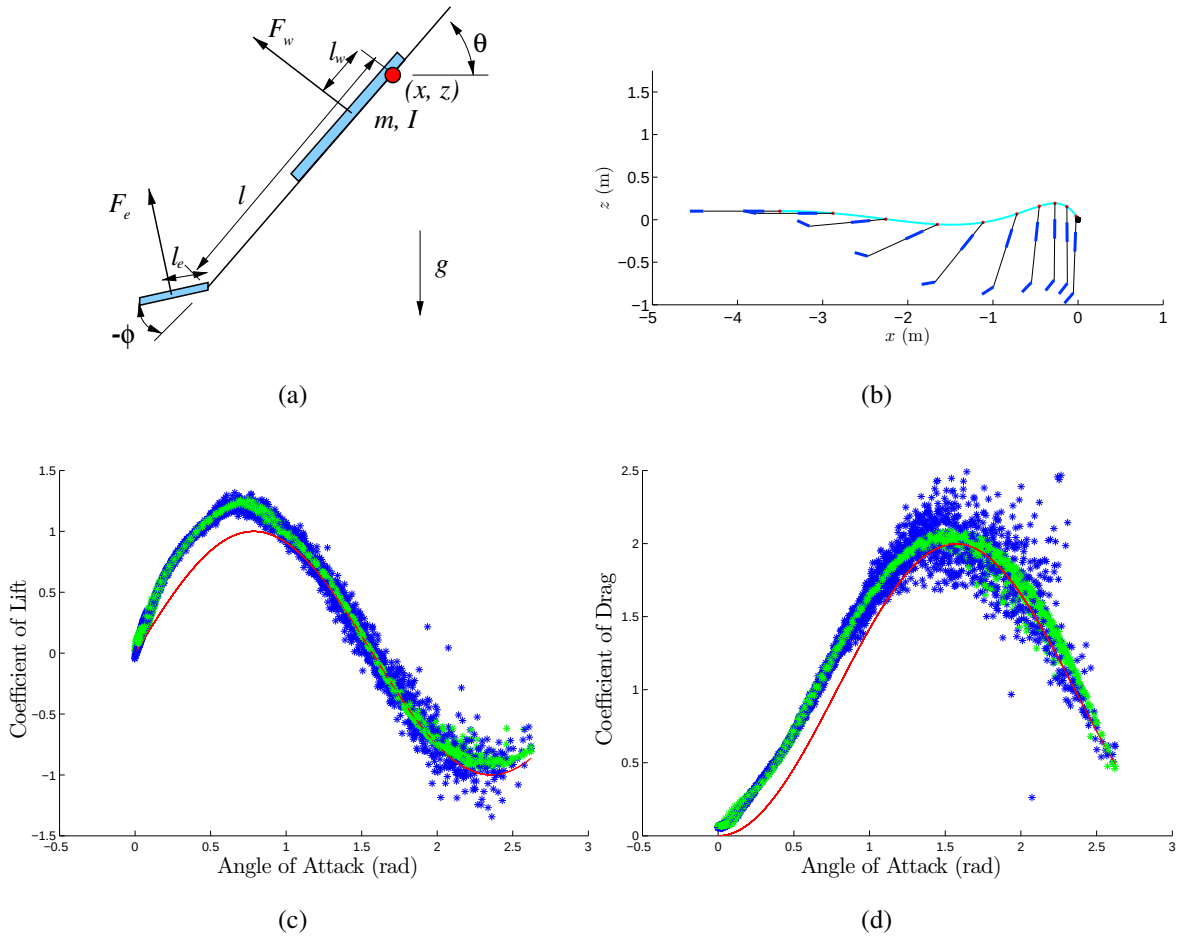


Figure 2. **(a)** Aircraft Model. x and z denote the positions of the center of mass, θ denotes the pitch angle, and ϕ denotes the elevator angle. **(b)** Optimized nominal perching trajectory computed using direct collocation. **(c-d)** Plot of Lift and Drag Coefficients: Blue represents the flight data, red represents the flat plate model, and green is the flat plate model augmented with radial basis functions

respectively. Finally, the dynamics are given by

$$m\begin{bmatrix}\ddot{x}\\ \ddot{z}\end{bmatrix} = \mathbf{F}_w + \mathbf{F}_e - \begin{bmatrix}0\\ mg\end{bmatrix} \tag{8}$$

$$I\ddot{\theta} = \begin{bmatrix}l_w\\ 0\end{bmatrix} \times \mathbf{F}_w + \begin{bmatrix}-l - l_e c_\theta\\ -l + l_e s_\theta\end{bmatrix} \times \mathbf{F}_e. \tag{9}$$

6.2. Radial Basis Function Augmentation

To improve the aircraft model, an effort was made to augment the aircraft lift and drag coefficients with Guassian radial basis functions, using the aforementioned flat plate glider as a baseline. To achieve this, data was collected by launching the aircraft approximately 50 times over a range of initial velocities from 6 to 8 m/s using a set of optimal elevator

trajectories computed for the flat plate model. To obtain the aircraft accelerations, the position measurements produced by the Vicon motion capture system were differentiated twice and filtered acausally.

Once all the data was collected, the accelerations predicted by flat plate theory were subtracted from the aircraft's accelerations. These residual accelerations were then assumed to contribute to residual lift, drag, and moment coefficients of the entire aircraft. In other words, residual aerodynamic coefficients $C_{l,r}$, $C_{d,r}$ and $C_{m,r}$ were modeled as functions of both wing angle of attack and elevator position, using Gaussian radial basis functions were specified on a two dimensional grid over a range of wing angle of attacks from $\mu_{\alpha_w,k} \in [0, \pi]$, elevator angles from $\mu_{\phi,k} \in [-\frac{\pi}{4}, \frac{\pi}{4}]$, and a fixed covariance $\Sigma = \text{diag}\begin{bmatrix}0.1 & 0.1\end{bmatrix}$, such that:

$$C_{i,r} = \sum_{k=1}^{N} \psi_k e^{-\frac{1}{2}\left\| \begin{bmatrix} \alpha_w - \mu_{\alpha_w,k} \\ \phi - \mu_{\phi,k} \end{bmatrix} \right\|_\Sigma^2} \tag{10}$$

Regularized least squares was then used to fit the radial basis function magnitudes and develop a more accurate aircraft model. Figure 2(c) compares the resulting model's aerodynamic coefficients with the data post-processed from Vicon.

6.3. Motor saturation and delay

The final two features of our aircraft that we must take in to consideration are the saturation of the servomotor and closed-loop delay. The Hitec servomotor at the elevator saturates at approximately 11.5 rad/s; in order to avoid this saturation we command a velocity explicitly by sending virtual position commands and exploiting the internal trapezoidal vleocity control of the servo. The total delay in our closed-loop system is approximately 60ms. Rather than adding a finite-order approximation of this delay to our continuous dynamics (and increasing the state dimension), we accomodate for it at runtime by simulating the model system forward for 60ms from the estimated current state and evaluating the feedback controller from this future statel.

7. Feedback Control Design

The lumped parameter model described in the previous section helps elucidate the difficulty of the perching control problem. The dynamics of the model are highly nonlinear - even the slope of the lift and drag coefficients will change considerably over the course of a perching maneuver. The control system has only a single actuator to control seven dynamic state variables, making the problem "underactuated" [35]; this combined with the saturations of the motor imply that the dynamics of the system cannot be "feedback linearized". Feedback linearization forms the basis for many common approaches to nonlinear control. Similarly, computational methods like based on dynamic programming also fail because the size of the state space is prohibitively large [16]. Partial feedback linearization [32], while possible, provides control in only one dimension in the worst case, and is not enough to transfer the

glider from the initial conditions to the goal state, which bounds final position, velocity and pitch errors simultaneously.

Two general methods of for nonlinear control which are applicable here are Nonlinear Model Predictive Control (NMPC) and LQR-Trees. In an attempt to distinguish the control possibilities independently from the control design approach, we investigate both methods and compare their results. Both methods start by making use of nonlinear optimal trajectory design to manage the system's nonlinearities, so we begin our discussion there.

7.1. Optimal Trajectory Design

The first step for applying both NMPC and LQR-Trees involves finding a nominal trajectory for the perching maneuver. Despite the relative complexity of our aircraft dynamics, standard tools for trajectory optimization work well for designing a nominal, locally optimal trajectory. To find an optimal trajectory for the glider, we chose to use a direct-collocation method [39] implemented using SNOPT [15]. We prefer this direct method for trajectory optimization over the shooting methods based on the adjoint equations [4] since, by treating both the control inputs $\{u_0, u_1, ..., u_{N-1}\}$ and the system states $(x_0, x_1, ..., x_{N-1})$ as variables in the optimization routine, it is very easy for constraints can be placed on the states and the inputs.

To find our optimal trajectory we solve the following problem

$$\min_{x_n, u_n, h} J = \sum_{n=0}^{N-1} hRu_n^2 + \mathbf{x}_n^T Q \mathbf{x}_n \tag{11}$$

$$\text{s.t. } x_{n+1} = x_n + h f(x_n, u_n). \tag{12}$$

In the above equations, R is the cost on action, Q is the cost on state, and h is the size of the timestep. For our problem, we set $Q = \text{diag}[10, 10, 10, 10, 10, 10, 10]$ and $R = 100$. We then optimize our trajectory using an initial state vector $\mathbf{x}(t_0) = [-3.5, 0.1, 0, 0, 7, 0, 0]$. Our final value constraints are represented by upper and lower bounds on the final state, $\mathbf{x}(t_f)$. They are $\mathbf{x}_u(t_f) = [0, 0, \frac{\pi}{2}, \frac{\pi}{8}, 2, 0, \infty]$ and $\mathbf{x}_l(t_f) = [0, 0, \frac{\pi}{8}, -\frac{\pi}{3}, 0, -2, -\infty]$ respectively. The constraints on ϕ are $\phi \in [-\frac{\pi}{3}, \frac{\pi}{8}]$ and the constraints on $\dot{\phi}$ are $\dot{\phi} \in [-13, 13]$. The resulting nominal trajectory is shown in Figure 2(b).

7.2. Nonlinear Model-Predictive Control

Trajectory optimization is often considered as an offline design process used to plan the nominal behavior of the aircraft. However, if the trajectory optimization approach is computationally fast and robust, a natural approach to nonlinear control is to re-solve the trajectory optimization for every control decision given the current estimated state as an initial conditions. This approach is called nonlinear model-predictive control (NMPC). While trajectory optimization can be computationally demanding, online NMPC can take advantage of a trajectory that is optimized offline to provide a nearly optimal initial solution, and reduce the online computation to just a few iterations of the nonlinear optimization, performed at

every control timestep, to determine the control response from regions in state space where the aircraft ends up due to unexpected modeling errors or disturbances.

As we report in Figure 3, our implementation NMPC was able to accomplish the task (bringing the glider to a final condition within 6.5cm of the perch) from a large range of initial conditions using the model in simulation. However, this performance was still too computationally demanding for real time implementation.

7.3. Time-Varying Linear Quadratic Regulator

One way to avoid the computational burden of NMPC is to linearize the nonlinear system dynamics about the nominal trajectory described in Section 7.1 and use optimal control to derive a closed-form approximation to the NMPC problem which is valid in a local neighborhood around the trajectory. For the cost function used here, this approximation is given by the solution to a time-varying linear quadratic regulator (TVLQR) problem, the basic building block of the LQR-Tree controller.

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

$$\dot{\bar{\mathbf{x}}}(t) = \mathbf{A}(t)\bar{\mathbf{x}}(t) + \mathbf{B}(t)\bar{\mathbf{u}}(t), \tag{13}$$

where $\bar{\mathbf{x}}(t) = \mathbf{x}(t) - \mathbf{x}_0(t)$ and $\bar{\mathbf{u}}(t) = \mathbf{u}(t) - \mathbf{u}_0(t)$. Although it is possible to solve explicitly for a final state constraint [35], in practice we obtain better performance by relaxing the final state constraint with a final state cost, resulting in a cost function of the form:

$$J = \bar{\mathbf{x}}(t_f)^T \mathbf{Q}_f \bar{\mathbf{x}}(t_f) + \int_0^T \left[\bar{\mathbf{x}}(t)^T \mathbf{Q} \bar{\mathbf{x}}(t) + \bar{\mathbf{u}}(t)^T \mathbf{R} \bar{\mathbf{u}}(t)\right] dt. \tag{14}$$

We also change the relative costs to $\mathbf{Q} = diag([10,10,10,1,1,1,1])$, $\mathbf{R} = .1$, and $\mathbf{Q}_f = diag([400,400,\frac{1}{9},\frac{1}{9},1,1,\frac{1}{9}])$, where the changes in $\mathbf{Q}$ were made to encourage the vehicle to stay close to the nominal trajectory (where the linearization is valid) and the cost on $\mathbf{R}$ was chosen so that over a reasonable range of initial conditions, control saturations were limited. The entries of $\mathbf{Q}_f$ were selected by approximating the desired goal region as an ellipsoid, $G = \bar{\mathbf{x}}^T \mathbf{Q}_f \bar{\mathbf{x}}$, so that the maximum distances are defined by $\bar{\mathbf{x}}_{f,max} = [0.05, 0.05, 3, 3, 1, 1, 3]$ where $\mathbf{Q}_f = diag(\bar{\mathbf{x}}_{f,max})^{-2}$. Furthermore, the entries for $\mathbf{Q}$ were also chosen with respect to $\mathbf{Q}_f$ so that the solutions to the Riccati equation remained well conditioned.

Finally, we solve the differential Riccati equation backwards in time

$$-\dot{\mathbf{S}}(t) = \mathbf{Q} - \mathbf{S}(t)\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}(t) + \mathbf{S}(t)\mathbf{A} + \mathbf{A}^T\mathbf{S}(t), \tag{15}$$

$$\mathbf{S}(t_f) = \mathbf{Q}_f. \tag{16}$$

which yields the TVLQR control law

$$\bar{\mathbf{u}} = -\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}(t)\bar{\mathbf{x}}(t), \tag{17}$$

and the cost-to-go

$$J = \bar{\mathbf{x}}^T\mathbf{S}(t)\bar{\mathbf{x}}. \tag{18}$$

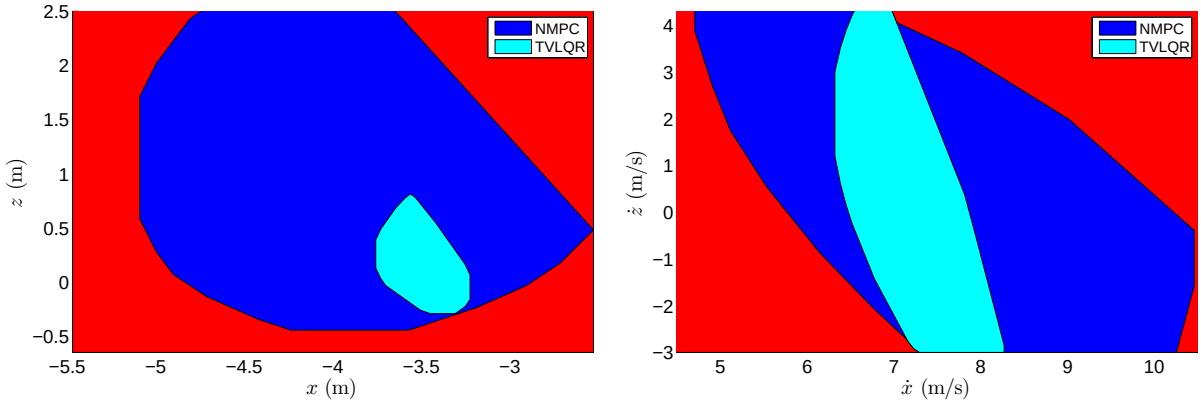


Figure 3. Comparison of NMPC with TVLQR via exhaustive simulation from a subspace of initial conditions. Cyan indicates the initial conditions for which the TVLQR controller navigates the aircraft through the goal region, blue indicates the same for NMPC. (a) Initial conditions were densely sampled from $\mathbf{x}(t_0) = [x, z, 0, 0, 7, 0, 0]$, (b) initial conditions were densely sampled from $\mathbf{x}(t_0) = [-3.5, 0.1, 0, 0, \dot{x}, \dot{z}, 0]$.

7.4. NMPC vs. TVLQR

The local linear approximation of the dynamics and locally quadratic approximation of the objective provides a feedback control law which is trivial to compute at run time (evaluating one piecewise polynomial spline which provides the feedback gains, then one matrix multiplication), but we cannot necessarily expect this controller to perform well for initial conditions far away from the nominal trajectory. To quantify this, we conducted exhaustive numerical simulations of the closed-loop systems from a range of initial conditions. Since the state space is ≥ 7 dimensions, this exhaustive evaluation is only possible in small subspaces of initial conditions. Given a particular initial condition, the closed-loop system was considered successful if the state of the aircraft during forward simulation entered the goal region around the perch at *any* time in the trajectory (a more generous evaluation than only testing at the specific final time of the nominal trajectory). Figure 3 shows the results, which confirms that the nonlinear controller from NMPC significantly outperforms the optimal time-varying linear controller. Unfortunately, this quantitative comparison is limited to our simulation model, as we are unable to evaluate the NMPC controller at real-time rates in the physical experiments.

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

creation cost of the library. This control library idea, along with a recipe for constructing an efficient library coverage with minimal controllers, is called "LQR-Trees"[36].

In order to achieve coverage, the LQR-Trees algorithm needs to be able to efficiently *verify* the local controllers, as in Figure 3. But this verification must work in the full state-space of the model, not just a subspace of initial conditions; and to be practical it must be much more efficient than the exhaustive simulations. To achieve this, we examine algorithms for producing efficient inner-approximations of these verified regions.

7.5. Verification Methods

The controls community has produced a number of approaches for verifying the behavior of complex dynamical systems. In [21], the authors make use of game theory and solve a Hamilton-Jacobi-Isaacs partial differential equation by discretizing on a grid to compute the set of reachable states for a continuous dynamical system. In [38], this approach is extended to handle hybrid systems. Besides grid-based methods, some researchers have explored methods of approximating reachable sets through combining numerical simulation with sensitivity analysis [13]. In [11], the authors merge this technique with rapidly exploring random trees to explore the state space of a dynamical system.

Here, we explore the an alternative approach to verification which uses of sums-of-squares (SOS) optimization [26] to compute an inner approximation of the regions displayed in Figure 3, more formally defined as the "backwards reachable set to the goal". This approach has some merits compared to the techniques mentioned above. First of all, this method reasons algebraically about the governing equations, eliminating the need discretize the systems state space. This requires that the closed-loop system of interest be described or approximated in closed-form using polynomials. Moreover, because computationally costly PDE solvers are replaced with semi-definite program solves, the SOS optimization routines scale much more reasonably with state than other methods. Since our system of interest has a large state space but is described with a smooth closed-form expression, we focus on the sums-of-squares approach here.

7.6. Sums-of-Squares Verification

Verification using sums-of-squares optimization is accomplished by searching for a verification certificate in the form of a Lyapunov function [26], which we will denote $V(\mathbf{x})$. For dynamical systems with polynomial vector fields, SOS makes it possible to verify the Lyapunov conditions ($V > 0, \dot{V} < 0$) for a candidate Lyapunov function as a convex optimization [26]. It is also possible to search for a Lyapunov function which satisfies these conditions, to search for a Lyapunov function which proves stability in a bounded invariant region, and/or to simultaneously optimize a controller in order to maximize a verified invariant region by solving a series of convex optimizations [20, 28, 33, 42].

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

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

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

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

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

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

In SOS optimization, we replace the test for positivity with the condition that the polynomial is a sum-of-squares. To isolate the analysis to the region defined by the sub-level set of B we apply a technique called the S-procedure[26] which is analogous to the use of Lagrange multipliers in constrained optimization. Using Σ to denote the sums-of-squares polynomials over $\mathbf{x}$, we write

$$V \in \Sigma \tag{19}$$

$$-\dot{V} + s_1(V - \rho) \in \Sigma \tag{20}$$

$$s_1 \in \Sigma \tag{21}$$

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

If we let $V = \bar{\mathbf{x}}^T \mathbf{S} \bar{\mathbf{x}}$, for some $\mathbf{S} = \mathbf{S}^T \succ 0$, then the constraint in equation 19 is satisfied trivially. We can now perform the optimization in the following steps:
STEP 1:

$$\begin{aligned} \underset{s_1,\gamma}{\text{maximize}} \quad & \gamma \\ \text{subject to} \quad & \gamma > 0, \\ & -\dot{V} + s_1(V - \rho) - \gamma \in \Sigma. \\ & s_1 \in \Sigma \end{aligned} \tag{22}$$

STEP 2:

$$\begin{aligned} \underset{V,\rho}{\text{maximize}} \quad & \rho \\ \text{subject to} \quad & \rho > 0, \\ & -\dot{V} + s_1(V - \rho) \in \Sigma. \\ & s_1 \in \Sigma \end{aligned} \tag{23}$$

These steps then repeat until convergence is observed in ρ.

7.7. Verification along Trajectories

The approach presented in the preceding section can also be applied to computing the backwards reachable set to a goal region along finite-time trajectories. To achieve this, we generalize our Lyapunov function to be a positive function of time as well as state, $V(\mathbf{x},t)$. Rather than requiring stability to some fixed-point, we search for a bounded invariant region

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

which can be verified by ensuring that $\dot{V} < \dot{\rho}$ for all states on the boundary of the region, $V = \rho$. Following [36, 37], we call these bounded invariant regions "funnels". We are then able to reformulate our constraints as:

$$V(\mathbf{x},t) = \rho(t) \implies \dot{V}(\mathbf{x}) - \dot{\rho}(t) \leq 0, \tag{24}$$

where

$$\dot{V}(\mathbf{x},t) = \frac{\partial V(\mathbf{x},t)}{\partial \mathbf{x}}\dot{\mathbf{x}} + \frac{\partial V(\mathbf{x},t)}{\partial t}. \tag{25}$$

Using the generalized S-Procedure, we can now write

$$-\dot{V}(\mathbf{x},t) + \dot{\rho}(t) - \mu(\mathbf{x},t)(V(\mathbf{x},t) - \rho(t)) \geq 0, \tag{26}$$

where $\mu(\mathbf{x},t)$ is a polynomial Lagrange multiplier. To further simplify the problem, we sample in time to eliminate t from the above expression, as recommended in [37]. This results in N positivity constraints

$$-\dot{V}(\mathbf{x},t_i) + \dot{\rho}(t_i) - \mu_i(\mathbf{x})(V(\mathbf{x},t_i) - \rho(t_i)) \geq 0, \tag{27}$$

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

$$\rho(t) = \rho_i + (t - t_i)\dot{\rho}_i, \quad i = \text{floor}\left(\frac{t - t_0}{\Delta t}\right) \tag{28}$$

$$\dot{\rho}_i = \frac{\rho_{i+1} - \rho_i}{\Delta t}. \tag{29}$$

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

$$\Phi(t) = \Phi_i + (t - t_i)\dot{\Phi}_i, \quad i = \text{floor}\left(\frac{t - t_0}{\Delta t}\right) \tag{30}$$

$$\Phi_i^T = \Phi_i \succ 0, \quad \dot{\Phi}_i = \frac{\Phi_{i+1} - \Phi_i}{\Delta t}. \tag{31}$$

This ensures that $S^T(t) = S(t) \succ 0$. To search for Φ_i and ρ_i, we once again replace the test for positivity with the test for sums-of-squares positivity. We then hold the scale of $\mathbf{S}(t)$ fixed and use bilinear alternations to search iteratively over ρ_i and μ_i in a manner similar to the steps

showed in equations 22 and 23. In this instance, the alternations terminate when $\sum_{i=1}^{N} \rho_i$ attains a local maximum.

It is this ability to verify along trajectories that allows us to apply our verification methods to the perching maneuver. The "funnels" found for this system can be seen in Figure 4(a) and 4(b). We also plot the case where the Lyapunov function is only parametrized by a re-scaling of $\bar{\mathbf{x}}^T \mathbf{S}_0(t) \bar{\mathbf{x}}$ to show the advantage of parameterizing the Lyapunov function by the entire $\mathbf{S}$-matrix using the Φ_i terms.

7.8. Robust Verification

These verifications methods can also be applied to situations where uncertainty exists in the system dynamics using the notion of a "common Lyapunov function". Essentially, we must verify that the Lyapunov conditions are met for all possible values of the uncertain parameters. This can be achieved by applying the following constraints:

$$\forall a_i \in [a_{i,min}, a_{i,max}], \quad V(\mathbf{x},t) = \rho(t) \implies$$
$$\dot{V}(\mathbf{x},t,\mathbf{a}) = \frac{\partial V(\mathbf{x},t,\mathbf{a})}{\partial x} f(\mathbf{x},t,\mathbf{a}) + \frac{\partial V(\mathbf{x},t,\mathbf{a})}{\partial t} < \dot{\rho}(t) \qquad (32)$$

where a_i is the i^{th} bounded uncertain parameter and $\rho(t)$ is the value of the level set of interest. As before, SOS can be used to search for an invariant set about the system's nominal trajectory. For the glider, we applied the bounded uncertainty term as a bounded process noise on the system dynamics. The resulting robust funnel is shown in Figure 4(c) and 4(d), where it is compared to the funnel for the system without uncertainty. We note that, to our knowledge, this is the most complex system to date for which this backwards-reachable set computations have been carried out using SOS.

7.9. SOS verification compared to exhaustive numerical simulation

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

7.10. LQR-Trees

As demonstrated in [24], these verified regions can be combined together to form a LQR-Tree to cover a sufficiently large region of state space. This is achieved by sampling randomly

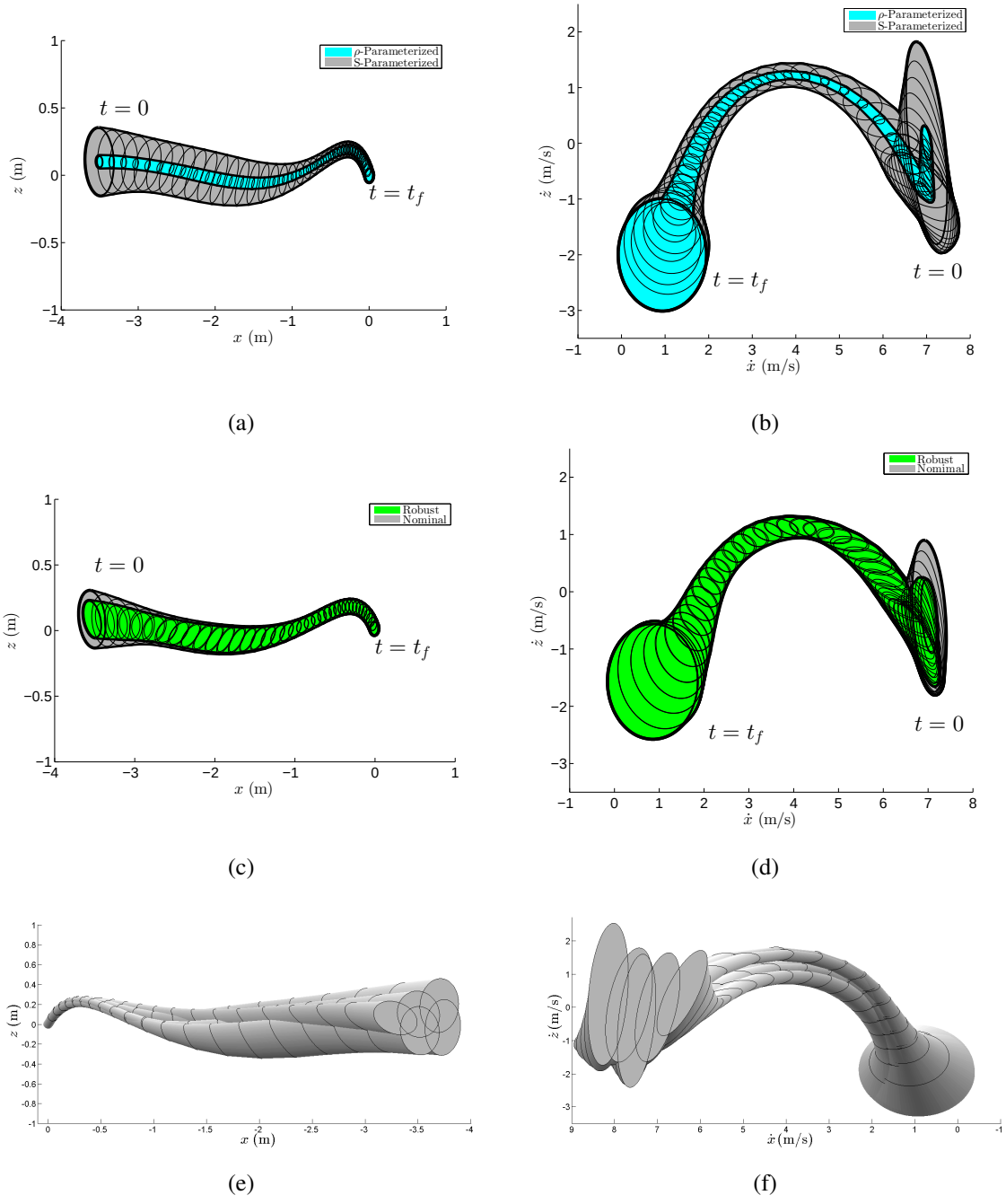


Figure 4. **(a-b)** Slices of the high-dimensional funnels in the x-z and $\dot{x}$-$\dot{z}$ planes respectively, obtained by optimizing only over $\rho(t)$ (cyan), compared with the funnel obtained by optimizing over $\Phi(t)$ and $\rho(t)$ (gray). The nominal trajectory for the plane begins at x=-3.5m and proceeds to the perch location x=0m, z=0m. **(c-d)** Slices of the high-dimensional funnel in the x-z and $\dot{x}$-$\dot{z}$ planes respectively, with (green) and without (gray) accounting for uncertainty. For the robust funnel, the uncertainty is modeled as a bounded input disturbance on the x-acceleration. **(e-f)** Slices of the high-dimensional LQR-Tree in the x-z plane and $\dot{x}$-$\dot{z}$ plane respectively. Noticed the increased initial condition coverage

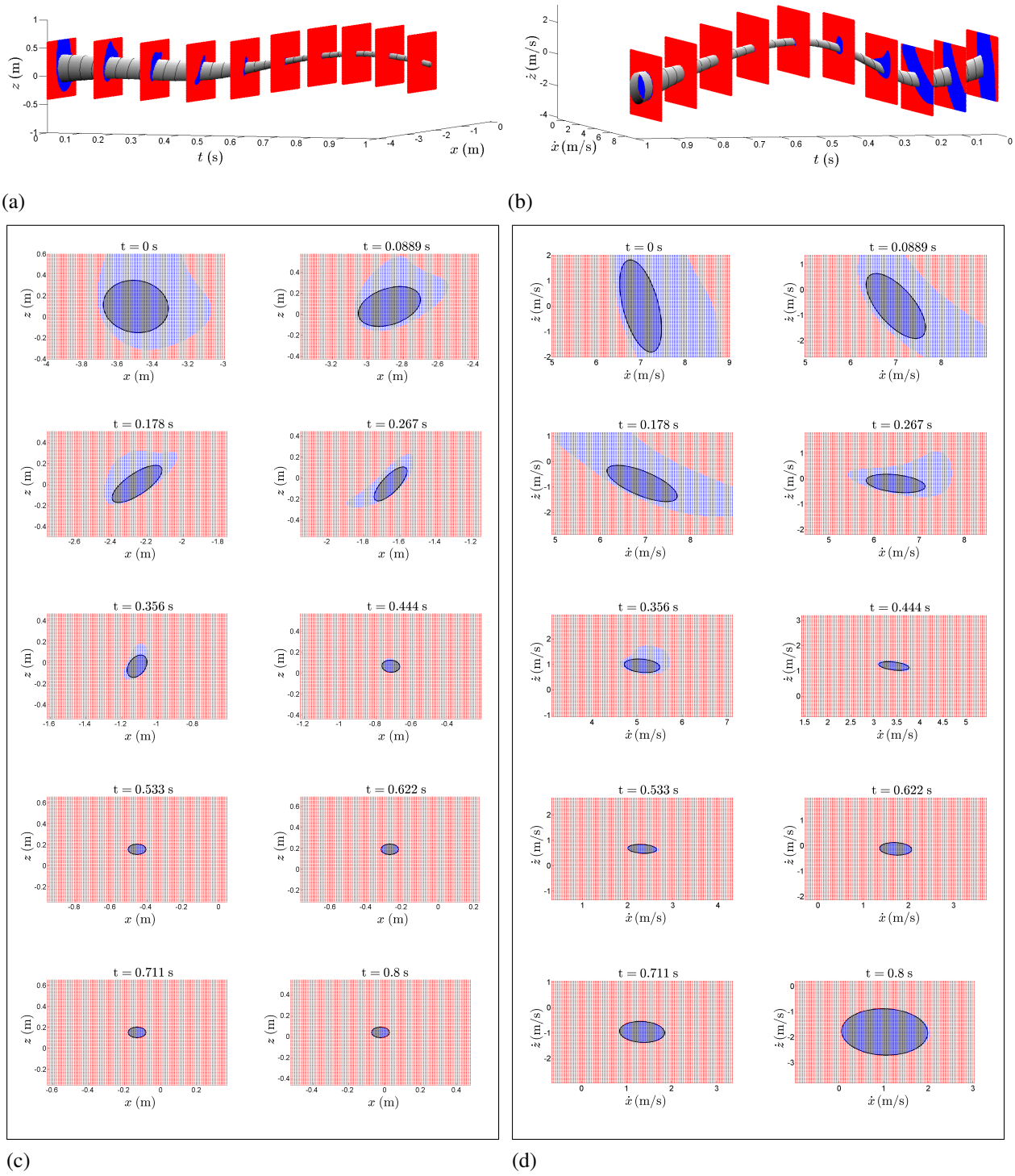


Figure 5. **(a-b)** Comparison of SOS funnel's x-z and $\dot{x}$-$\dot{z}$ slices with the invariant set found by simulating the full non-polynomial, nonlinear system forward from the relevant initial conditions. **(c-d)** 2D plots of the x-z and $\dot{x}$-$\dot{z}$ funnel slice time cross-sections found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

from a set of initial conditions and checking if the sample is already in the verified region. If it is, then the set of initial conditions is re-sampled. However, if the sample is not in the verified region, then a new trajectory is designed, where the trajectory's terminal constraints are specified to be along the tree's already existing trajectories. Once a trajectory from the initial condition to the goal is found, this trajectory is verified and a new funnel is added to the tree. This process repeats until no sampled initial conditions are outside the tree. A LQR-Tree for the glider is shown in Figure 4(e) and 4(f).

Notice how the LQR-tree in Figure 4(e) and 4(f) verifies a much larger range of initial velocities when compared to the region verified by any one funnel. It is also interesting to note that, although attempts were made to reconnect initial conditions to the nearest point along the original nominal trajectory, the algorithm still connected all the new trajectories to the goal region. This is most likely because, given the aircraft's extremely short flight time, it is more optimal for the trajectories to terminate at the goal region rather than at some earlier point along the nominal trajectory.

8. Experimental Results

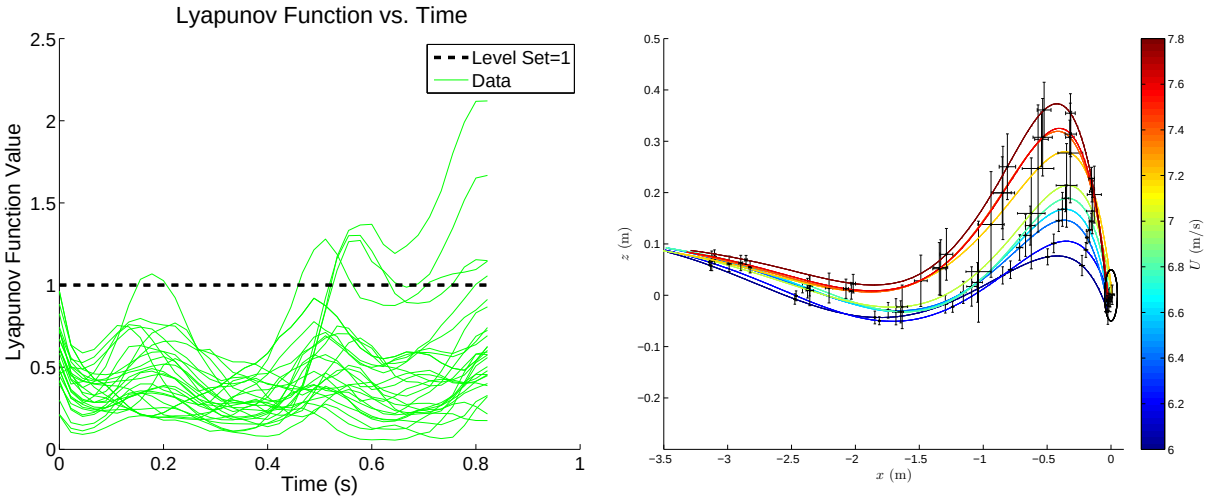
After confirming that our SOS funnels provided accurate, but conservative approximations of the invariant set of the the simulated system, we then proceeded to test our funnels in hardware.

8.1. Hardware verification of an individual funnel

Each verified funnel was computed using a detailed model of the system. Therefore, the ability of the analysis to verify the performance of the real system depends heavily on the quality of the model. To make our models as accurate as possible, we fit the radial basis functions for the "slop terms" in the aerodynamic coefficients independently for each controller in the library by collecting a number of trials using the open-loop nominal trajectory. We also required that the funnels be robust to uncertainty in the model, as described in section 7.8, by adding bounded process noise to the x-accelerations. When we evaluate trajectories of controlled perching maneuvers close to the nominal trajectory's initial speed of 7 m/s, we observe that if the initial conditions for our trajectory start in the funnel, 24 out of the 27 trajectories stay in the funnel. Moreover, those that did leave the funnel still remain within the 0.065 cm desired final perch range.

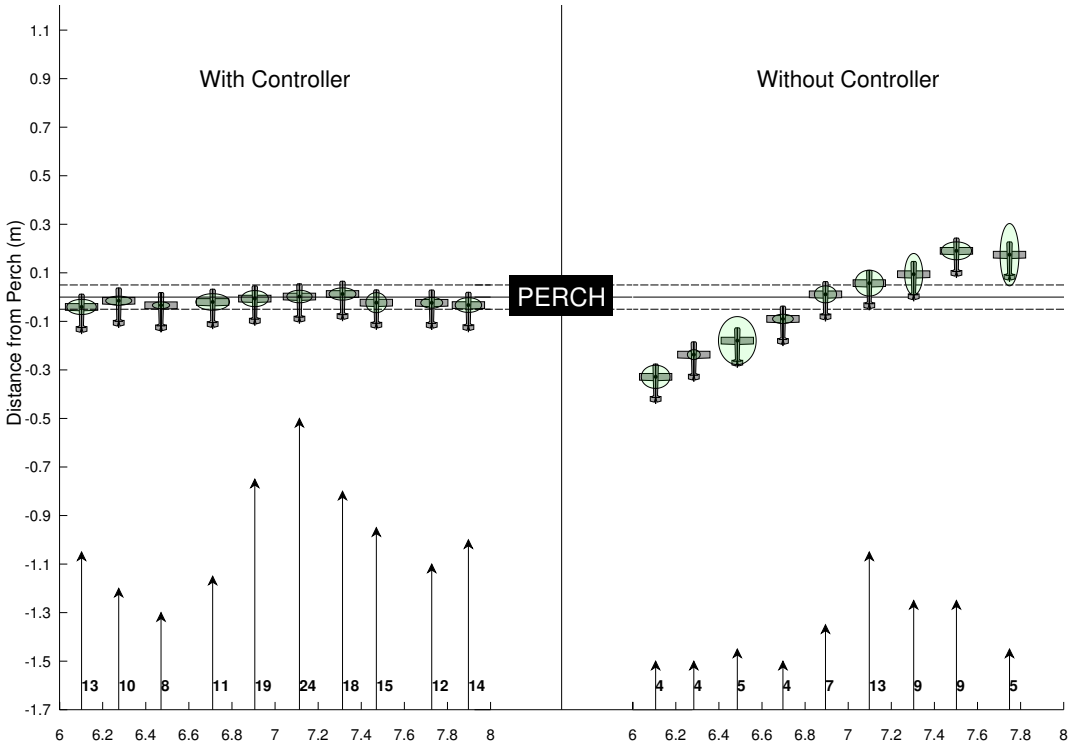
8.2. LQR-Trees

To implement the LQR-Tree algorithm in hardware, we initialize our tree with our original funnel designed for a launch speed of 7 m/s. We then sample from a range of initial velocities between 6 and 8 m/s to find a launch speed which is *not* in the original funnel. Using this new initial condition, we design a nominal trajectory, build a local model and compute a funnel for the locally stabilized system. We then repeat this process until the the entire range of initial speed conditions from 6 to 8 m/s are covered by a verified region. In this paper,



(a)

(b)



(c)

Figure 6. **(a)** Experimental trajectory verification for single stabilized trajectory. 24 out of the 27 flights stay in the robust funnel, and of those that left, all of them were in the 0.065 cm target range. **(b)** Averaged x-z trajectories from LQR-Tree perching experiments. Each curve represents an average of multiple trajectories with similar initial conditions; error bars are included to show the spread of the trajectories. The colors, from cold to hot, represent the variation in the initial speed of the aircraft. Notice how 94 percent of the trajectories, despite their various initial speeds, terminate in the 6.5 cm goal region. **(c)** Plot shows perching performance of LQR-Trees controller versus the open-loop controller. Each plane represents a set of trials grouped by initial speed. The ellipses represent the variation in position error (x-axis) and the variation in the initial speed (y-axis). The arrows represent the flight direction as well as the number of flights in each bin. Notice how 94 percent of the trials using the LQR-Trees controller landed in the 0.065 cm target range.

we focus primarily on perturbations due to initial speed since this is the most direct means of perturbing the system in a controlled fashion for our given launch method. The results of the LQR-Trees algorithm applied to the fixed-wing perching problem can be seen in Figure 8.1. Here, the algorithm is compared to a simple open loop control scheme. The improvement that the LQR-Trees algorithm makes in comparison to the open loop controller is quite dramatic, achieving a 94 percent success rate over 147 trials.

Compared to our previous work using linear feedback control [10], the resulting closed-loop controller achieves successful perching for a dramatically larger set of initial conditions. Moreover, in order to perch successfully using the previous approach, the glider had to be carefully launched using the specialized launching mechanism. Using the LQR-Trees controller, we can simply toss the airplane by hand and, if it starts in one of the funnels, we can expect it to land on the perch nearly every time (see multimedia attachment).

9. Discussion and Conclusions

In this paper we have demonstrated that, through the use of carefully-designed nonlinear feedback control, it is possible to achieve accurate and robust bird-like perching using nothing but a simple fixed-wing glider with a single actuator at the tail. This begs the fascinating question of the extent to which birds rely on their apparently superior morphology and sensing, or their seemingly impressive control system. But it also gives hope that our future UAVs may be able to reproduce some of the post-stall performance of birds without significant additional complexity in their hardware.

The application of the LQR-Trees algorithm presented here represents one of the most complex (in terms of dimensionality of the state space, and numerical conditioning of the Riccati differential equation due to limited controllability) demonstrations of this nonlinear feedback control approach to date. Compared to NMPC, these methods provide stronger theoretical guarantees of performance for the modeled system, the ability to directly incorporate model uncertainty, and efficient run-time performance, making them appealing for use on real experimental systems. The techniques also generalize to a large class of nonlinear control problems which may include other problems of interest for biology. Although the specific algorithm was motivated by technical advanced in planning and verification methods, the core idea of decomposing the controller into a family of simpler primitives is a common notion in biological motor control (c.f.[25]).

We hope that the work presented here will not only serve as an example of what can be achieved with convention UAVs using modern control design methods, but that it might inspire biologists to study nature's control systems in ever increasing depth. Deeper understanding of the post-stall control strategies used by birds could be invaluable to the development of future flight control systems.

10. Acknowledgements

This work was supported by an ONR MURI (N00014-10-1-0951) and NSF Award No: IIS-0915148.

References

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

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

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

[4] John T. Betts. Survey of numerical methods for trajectory optimization. *Journal of guidance, control, and dynamics*, 21(2):193–207, 1998.

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

[6] Anna C Carruthers, Graham K Taylor, Simon M Walker, and Adrian LR Thomas. Use and function of a leading edge flap on the wings of eagles. *AIAA Paper*, 43:2007, 2007.

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

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

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

[10] Rick E. Cory. Perching with fixed wings. Master's thesis, Massachusetts Institute of Technology, 2008.

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

[12] Alexis Lussier Desbiens, Alan Asbeck, and Mark Cutkosky. Hybrid aerial and scansorial robotics. *ICRA*, May 2010.

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

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

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

[16] J.H. Gillula, H. Huang, M.P. Vitus, and C.J. Tomlin. Design of guaranteed safe maneuvers using reachable sets: Autonomous quadrotor aerobatics in theory and practice. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1649–1654. IEEE, 2010.

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

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

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

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

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

[22] Joseph Moore and Russ Tedrake. Powerline perching with a fixed-wing UAV. In *Proceedings of the AIAA*

Infotech@Aerospace Conference, Seattle, WA, April 2009. AIAA.

[23] Joseph Moore and Russ Tedrake. Magnetic localization for perching UAVs on powerlines. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, September 2011.

[24] Joseph Moore and Russ Tedrake. Control synthesis and verification for a perching uav using lqr-trees. *In Proceedings of the IEEE Conference on Decision and Control*, December 2012.

[25] F.A. Mussa-Ivaldi, S.F. Giszter, and E. Bizzi. Linear combinations of primitives in motor control. *Proc Natl Acad Sci USA*, 91(16):7534–8, Aug 1994.

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

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

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

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

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

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

[32] Mark Spong. Partial feedback linearization of underactuated mechanical systems. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, volume 1, pages 314–321, September 1994.

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

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

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

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

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

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

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

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

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

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