Control Synthesis and Verification for a Perching UAV using LQR-Trees

Joseph Moore[1] and Russ Tedrake[2]

***Abstract*— We consider the problem of designing a feedback controller for a fixed-wing unmanned aerial vehicle (UAV) to execute a dynamic post-stall maneuver and land accurately on a perch. This controller must deal with the nonlinearity of the post-stall dynamics and inherently limited control authority of the actuators. Using a recently proposed algorithm called "LQR-Trees", here we demonstrate that we can generate formal guarantees (using time-varying Lyapunov functions) which verify that a controller generated with trajectory optimization and local linear feedback can achieve the desired perching tolerance from a range of initial conditions. These locally valid controllers are then combined into an library of controllers which cover a space of initial conditions with verified controllers. We describe some of the details of the implementation, which requires formulating and solving large semidefinite programs, and compare the verified regions with numerical samples of the system's true performance. This perching problem represents the most high-dimensional and numerically challenging application of the LQR-Trees algorithm reported to date.**

I. Introduction

Over the past several years, there has been considerable interest in developing fixed-wing unmanned aerial vehicles (UAVs) capable of executing dynamic, high precision landing maneuvers. Such maneuvers, although beyond the reach of modern control systems, have the potential to provide fixed-wing UAVs with several new capabilities such as landing on aircraft carriers, landing on narrow ledges to conduct perch and stare surveillance, and landing on powerlines to inductively recharge [19], [20]. Birds regularly execute these high precision maneuvers and it is our hope to be able to attain, and perhaps even exceed, their flight performance.

In this paper, we develop a closed-loop controller for a fixed-wing perching maneuver, building on previous work on system identification and optimal trajectory design [2], [3], [22], [10], [1]. Using a simple flat-plate approximation of the aerodynamic coefficients which is surprisingly faithful to our experimental data [2], we create locally optimal perching trajectories using a direct collocation method and use time-varying linear quadratic regulator (LQR) design to locally stabilize those trajectories. Then, we use semidefinite optimization to compute a Lyapunov function which certifies for the nonlinear model that a region of initial conditions around the nominal trajectory will achieve the perching target with the desired accuracy in finite time. The level sets of the time-varying Lyapunov function have a graphical interpretation as a funnel, which brings many initial conditions down into a small set of final conditions near the perch. These verified local controllers can then be combined into a LQR-Tree controller [25] that guarantees convergence over a range of flight conditions.

The LQR-Trees algorithm was proposed as a general tool for controller synthesis [25]. The perching application described here, however, is far beyond the complexity of previous systems to which this method was applied. Not only does the perching problem require a higher dimensional model, but its limited controllability makes it numerically challenging for our methods. In this paper we demonstrate that, not only is conducting controller verification and formulating LQR-Trees feasible, but that the resulting funnels are large enough to make LQR-Trees a useful approach.

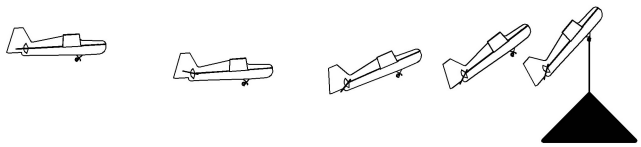

Fig. 1. Illustration of a Post-Stall Perching Trajectory

II. Background

The fixed-wing perching problem has been considered by several researchers over the last five or six years. At Cornell, an effort was made to develop a morphing aircraft which achieves fixed-wing perching by rotating only its fuselage upward but keeping the wings at a low angle-of-attack [29]. At Stanford, a perching aircraft was equipped with arrays of microspines which are able to attach to a wall [5], [6]. The aircraft flies towards the wall, senses the landing surface through ultrasonic range finding, and pitches up into a near vertical orientation before contacting the wall. At MIT, greater emphasis has been placed on nonlinear control design to solve the perching problem. In [2], [10], the authors built a nonlinear, post-stall model for a flat plat glider, designed an optimal open-loop trajectory for that glider, and verified the system in simulation. Since then, the authors have designed a linear time-varying LQR controller about this trajectory and successfully demonstrated their approach experimentally [1].

In addition to fixed-wing UAV perching, substantial work has gone into developing control algorithms for perching quadrotors. In addition to their work in developing techniques for allowing quadrotors to carry out aggressive maneuvers [15], [14], [16], these authors have demonstrated experimentally robust approaches for landing and perching with quadrotors [17].

[1]Ph.D. Candidate, Massachussetts Institute of Technology, Cambridge, MA 01239, USA (joemoore@mit.edu)

[2]Associate Professor of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 01239, USA (russt@mit.edu)

Recently, there has been collaboration between MIT and Stanford on the wall perching aircraft project. In [9], an effort has been made to find a region of attraction for the micro-spine equipped aircraft during the vehicle's landing phase. Because the wall-spine interaction dynamics are very different than the vehicle's flight dynamics, this "landing" funnel was found with the intention of it serving as an invariant set into which other "flight" funnels could terminate. It is our hope that the work discussed here will be able to provide these "flight" funnels.

The basis for the controller design comes from the ideas initially proposed in [25]. In this paper, the author describes the LQR-Tree algorithm approach which seeks to cover the state space of a dynamical system using a "tree" of time-varying trajectories which have been verified via sums of squares programming. Because the verified regions or "funnels" associated with these trajectory-dependent LQR controllers cover a finite region of state space, they can be combined together to create a certificate which verifies that a large range of initial conditions will eventually arrive at the goal. The details of the computational approach used to verify finite-time invariance around the trajectories (e.g., the funnels) were revisited in [26], where it was demonstrated that methods based on bilinear alternations could increase the size (by reducing the conservatism) of the resulting funnels. We use a similar bilinear alternation in this paper to find the time-invariant funnels for our perching glider. A few extensions to this work have also emerged. In [23], the regions of stability, which are based on deterministic systems, are extended to stochastic systems. Furthermore, in [12], a method of verifying regions of attraction is described where the goal region is allowed to vary during online controller execution.

In addition to the methods described above, which utilize sums of squares programming, the controls community has produced a number of other approaches for verifying the behavior of complex dynamical systems. In [18], the authors make use of game theory and solve a Hamilton-Jacobi-Isaacs partial differential equation to compute the set of reachable states for a continuous dynamical system. In [27], this approach is extended to handle hybrid systems. Besides set-based methods, some researchers have explored methods of approximating reachable sets through combining numerical simulation with sensitivity analysis [7]. In [4], the authors merge this technique with rapidly exploring random trees to explore the state space of a dynamical system. It is our belief, as described in section XI, that the LQR-Trees approach provides a number important advantages over these other methods.

III. Problem Formulation

We formulate the perching problem in the same manner as described in [2], [1]. Our experimental aircraft is a glider (no propellor) with slightly tapered flat-plate wings with a 98mm mean chord and an 8:3 aspect ratio. The only actuator we control during a perching maneuver is the elevator, actuated by a HS-55 Hitech servo. Using a crossbow, we launch our glider at 7 m/s (Re $\approx$ 50,000) at about 3.5 meters from the perch (a 0.25 cm diameter string). These initial conditions were selected so that the aircraft was required to execute a post-stall maneuver as cartooned in Fig. 1 in order to slow down and land on the perch. A perching maneuver is deemed to be successful if the aircraft perching talon lands within a 5 cm radius of the perch with a horizontal velocity between 0 and 2 m/s and vertical velocity between -1 m/s and -3 m/s; these are approximately the conditions for which our simple talon can hook the string. In our perching experiments, we use a Vicon Motion Capture system and reflective infrared markers on the glider's fuselage to obtain the state of the aircraft as it approaches the perch.

IV. Aircraft Model

From prior system identification work [2], a quasi-steady flat plate model was shown to be a reasonable approximation of the aircraft's post-stall aerodynamics, even during the rapid pitch-up maneuver involved in perching where the aircraft experiences a rapidly changing angle-of-attack. In the literature [24], these coefficients are given as

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

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

For this reason, we use a flat-plate approximation of the post-stall lift and drag coefficients for the rest of the paper as we carry out our control synthesis. To further simplify our model, we restrict our maneuver to two dimensions, ignoring contributions from yaw or three dimensional aerodynamics. This leads to a seven dimensional model, 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. 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, $u = \dot{\phi}$.

To build our model, we first define unit vectors normal to the control surfaces in the directions of the force vectors illustrated in Fig. 2 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 $c_\gamma = \cos(\gamma)$. Next, we solve for the kinematics of the geometric centroid of the aerodynamic surfaces. For our flat plate model, this 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

$$\alpha_w = \theta - \tan^{-1}(\dot{z}_w, \dot{x}_w), \quad \alpha_e = \theta + \phi - \tan^{-1}(\dot{z}_e, \dot{x}_e) \tag{5}$$

$$\mathbf{F}_w = \rho S_w |\dot{\mathbf{x}}_w|^2 \sin\alpha_w \mathbf{n}_w = f_w \mathbf{n}_w, \tag{6}$$

$$\mathbf{F}_e = \rho S_e |\dot{\mathbf{x}}_e|^2 \sin\alpha_e \mathbf{n}_e = f_e \mathbf{n}_e \tag{7}$$

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, respectively. Finally, the dynamics are given by

$$m\ddot{x} = -f_w s_\theta - f_e s_{\theta+\phi} \tag{8}$$
$$m\ddot{z} = f_w c_\theta + f_e c_{\theta+\phi} - mg \tag{9}$$
$$I\ddot{\theta} = -f_w l_w - f_e(l c_\phi + l_e) \tag{10}$$

which can be rewritten in state-space form as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, u). \tag{11}$$

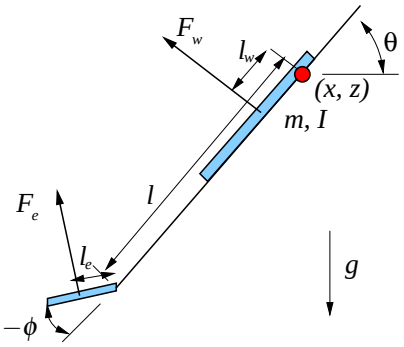

Fig. 2. Aircraft Model. x and z denote the positions of the center of mass, θ denotes the pitch angle, and ϕ denotes the elevator angle.

V. Trajectory Optimization

Despite the relative complexity of our aircraft model with the nonlinear and highly underactuated 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 [28] implemented using SNOPT [8], which allows us to efficiently encode hard constraints on the elevator velocity and position as well as on the final conditions of the trajectory.

We 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 $\dot{\phi} \in [-13, 13]$. For the cost function, we use $\int_{t_0}^{t_f} u^2 dt$. The resulting nominal trajectory is shown in Fig. 3.

VI. Trajectory Stabilization

To stabilize our nominal trajectory, we use a time-varying linear quadratic regulator (LQR). To apply this control methodology, we linearize our nonlinear system about the nominal trajectory to obtain a time-varying linear system which can be represented as:

$$\dot{\bar{\mathbf{x}}}(t) = \mathbf{A}(t)\bar{\mathbf{x}}(t) + \mathbf{B}(t)\bar{\mathbf{u}}(t), \tag{12}$$

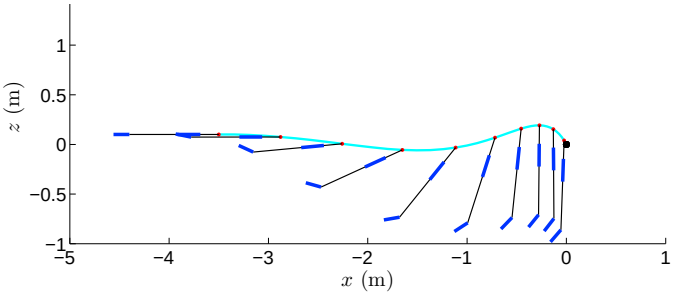

Fig. 3. Optimized nominal perching trajectory.

where $\bar{\mathbf{x}}(t) = \mathbf{x}(t) - \mathbf{x}_0(t)$ and $\bar{\mathbf{u}}(t) = \mathbf{u}(t) - \mathbf{u}_0(t)$. Let us define the following cost function for the maneuver:

$$J = \bar{\mathbf{x}}(t_f)^T \mathbf{Q}_f \bar{\mathbf{x}}(t_f) + \int_0^T \bar{\mathbf{x}}(t)^T \mathbf{Q} \bar{\mathbf{x}}(t) + \bar{\mathbf{u}}(t)^T \mathbf{R} \bar{\mathbf{u}}(t), \tag{13}$$

with $\mathbf{Q} = diag([10, 10, 10, 1, 1, 1, 1])$, $\mathbf{R} = .1$, and $\mathbf{Q}_f = diag([400, 400, \frac{1}{9}, \frac{1}{9}, 1, 1, \frac{1}{9}])$. 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 $\mathbf{Q}_f = diag(\bar{\mathbf{x}}_{f,max})^{-2}$. Finally, the entries for $\mathbf{Q}$ were chosen with respect to $\mathbf{Q}_f$ so that the solutions to the Riccati equation remained well conditioned. LQR design yields the control law

$$\bar{\mathbf{u}}(t) = -\mathbf{R}^{-1}\mathbf{B}^T \mathbf{S}(t)\bar{\mathbf{x}}(t), \tag{14}$$

and the cost-to-go

$$J = \bar{\mathbf{x}}^T \mathbf{S}(t)\bar{\mathbf{x}}, \tag{15}$$

where

$$\dot{\mathbf{S}}(t) = \mathbf{Q} - \mathbf{S}(t)\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}(t) + \mathbf{S}(t)\mathbf{A} + \mathbf{A}^T\mathbf{S}(t), \tag{16}$$
$$\mathbf{S}(t_f) = \mathbf{Q}_f. \tag{17}$$

The closed-loop system can now be written as

$$\dot{\bar{\mathbf{x}}} = \mathbf{f}(t, \bar{\mathbf{x}}) - \dot{\mathbf{x}}_0. \tag{18}$$

VII. Finite-Time Verification

Once the aircraft trajectory has been successfully stabilized, recent advances in sums of squares (SOS) analysis can be used to find an inner approximation of the region of state space about that stabilized trajectory for which the system is guaranteed to achieve the goal (e.g., land on the perch) with some tolerance. Following [26], we define this region as a "funnel" in to the goal region G.

To find a funnel for the glider, we first define the time-varying Lyapunov function $W(t, \bar{\mathbf{x}})$ over the interval $[t_0, t_f]$. We parameterize this function as

$$W(t, \bar{\mathbf{x}}) = \frac{V(t, \bar{\mathbf{x}})}{\rho(t)}, \tag{19}$$

where $V(t, \bar{\mathbf{x}})$ is the cost-to-go function, $\bar{\mathbf{x}}^T \mathbf{S}(t)\bar{\mathbf{x}}$, from LQR and $\rho(t)$ is the class time varying functions such that $\rho(t) > 0$ and $\rho(t_f) \leq 1$. By defining $\rho(t)$ in this way, we are able

to guarantee positive definiteness of $W(t,\bar{\mathbf{x}})$ and that the outlet of the funnel is contained in goal region $G = \bar{\mathbf{x}}^T \mathbf{S}(t_f)\bar{\mathbf{x}}$. To ensure that $W(t,\bar{\mathbf{x}})$ guarantees set invariance over some region, we impose the constraint

$$V(t,\bar{\mathbf{x}}) = \rho(t) \implies \dot{V}(t,\bar{\mathbf{x}}) - \dot{\rho}(t) \leq 0, \tag{20}$$

where

$$\dot{V}(t,\bar{\mathbf{x}}) = \frac{\partial V(t,\bar{\mathbf{x}})}{\partial \bar{\mathbf{x}}}\dot{\bar{\mathbf{x}}} + \frac{\partial V(t,\bar{\mathbf{x}})}{\partial t}. \tag{21}$$

Using the generalized S-Procedure, we can now write

$$-\dot{V}(t,\bar{\mathbf{x}}) + \dot{\rho}(t) - \mu(t,\bar{\mathbf{x}})(V(t,\bar{\mathbf{x}}) - \rho(t)) \geq 0, \tag{22}$$

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

$$-\dot{V}_i(\bar{\mathbf{x}}) + \dot{\rho}_i - \mu(\bar{\mathbf{x}})_i(V_i(\bar{\mathbf{x}}) - \rho_i) \geq 0, \tag{23}$$

where N is the total number of sample points and the subscript i denotes the given function evaluated at $t = t_i$, 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

$$\rho(t) = \rho_i + (t - t_i)\dot{\rho}_i, \quad i = \text{floor}\left(\frac{t - t_0}{\Delta t}\right) \tag{24}$$

$$\dot{\rho}_i = \frac{\rho_{i+1} - \rho_i}{\Delta t}. \tag{25}$$

To search for ρ_i, we replace the test for positivity with the test for sums of squares positivity[21].

In order to make use of existing software tools for sums of squares [11], [13], we make a polynomial approximation of the nonlinear dynamics $\dot{\bar{\mathbf{x}}} = \mathbf{f}(\bar{\mathbf{x}}) - \dot{\mathbf{x}}_0$ using a third-order Taylor series. Letting $\Sigma[\bar{\mathbf{x}}]$ be the set of all sums of squares polynomials, we search for ρ_i by setting up the optimization problem

$$\begin{aligned} \underset{\rho_i,\mu_i}{\text{maximize}} \quad & \sum_i^N \rho_i \\ \text{subject to} \quad & \rho_N = 1 \\ & \forall i \in [1, N-1]: \\ & \rho_i > 0, \\ & -\dot{V}_i(\bar{\mathbf{x}}) + \dot{\rho}_i - \mu_i(\bar{\mathbf{x}})(V_i(\bar{\mathbf{x}}) - \rho_i) \in \Sigma[\bar{\mathbf{x}}]. \end{aligned} \tag{26}$$

Unfortunately, this optimization problem is not jointly convex in the decision parameters ρ_i and (the coefficients of) $\mu_i(x)$. Following [26], we apply a bilinear alternation to approximate the maximum. We begin the bilinear alternation by first searching for the Lagrange multipliers using an initial guess of $\rho(t)$. For our initial guess, we use the function suggested in [26] where

$$\rho(t) = e^{c\frac{t - t_f}{t_f - t_i}}. \tag{27}$$

We steadily increased c from $c = 5$ until the search for the Lagrange multipliers became feasible at $c = 13.5$.

The funnels resulting from this method are depicted in Figs. 4 and 5. The optimization converged after 7 iterations, each of which took approximately 36 seconds (see Table I). Our verification sampled the system at 38 knot points.

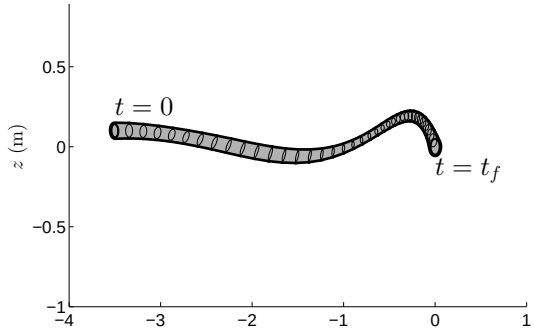

Fig. 4. Slice of the high-dimensional funnel x,z plane obtained by optimizing over $\rho(t)$. The nominal trajectory for the plane begins at x=-3.5m and proceeds to the perch location x=0m, z=0m.

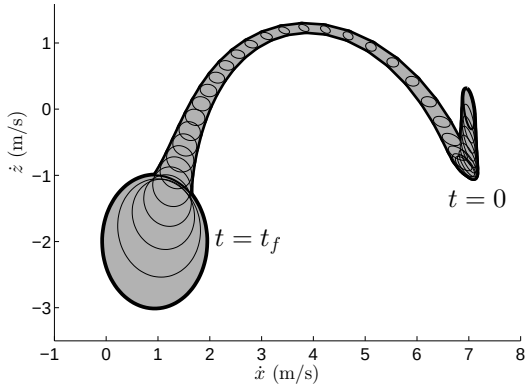

Fig. 5. Slice of the high-dimensional funnel in the $\dot{x},\dot{z}$ plane obtained by optimizing over $\rho(t)$. Note that the nominal trajectory begins *on the right* with a positive velocity, and ends on the left with near zero velocity.

VIII. S-Parameterization

We found the funnels estimated using the $\rho(t)$ parameterization to be overly conservative when compared against a simulation-based analysis. To improve these results, we optimized over $\mathbf{S}(t)$, where $V(t,\bar{\mathbf{x}}) = \bar{\mathbf{x}}^T\mathbf{S}(t)\bar{\mathbf{x}}$ and $\dot{V}(t,\bar{\mathbf{x}}) = 2\bar{\mathbf{x}}^T\mathbf{S}(t)\dot{\bar{\mathbf{x}}} + \bar{\mathbf{x}}^T\dot{\mathbf{S}}(t)\bar{\mathbf{x}}$, instead of simply searching for level sets of our initial Lyapunov candidate.

Consider the sums of squares constraints from the previous optimization procedure given in 26. To find an initial set of Lagrange multipliers, we use the initial $\rho(t)$ from the previous section. However, in the next step of the optimization procedure, instead of searching over a piecewise linear polynomial representation of $\rho(t)$, we search over a parameterization of $\mathbf{S}(t)$.

A. S-matrix Representation

To parameterize our S-matrix, we consider a point-wise rescaling of the elements in the original $\mathbf{S}_0(t)$ matrix obtained from the Riccati equation. This can be represented using the Hadamard product as,

$$\mathbf{S}(t) = \mathbf{S}_0(t) \odot \Phi(t), \tag{28}$$

where $\Phi(t) \geq 0$. $\dot{\mathbf{S}}(t)$ can then be written as

$$\dot{\mathbf{S}}(t) = \dot{\mathbf{S}}_0(t) \odot \Phi(t) + \mathbf{S}_0(t) \odot \dot{\Phi}(t) \tag{29}$$

where $\dot{\mathbf{S}}_0$ comes from the solution to the Riccati equation and $\Phi(t)$ is a piecewise linear polynomial function such that

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

$$\dot{\Phi}_i = \frac{\Phi_{i+1} - \Phi_i}{\Delta t}. \tag{31}$$

Our proposed representation for $\mathbf{S}(t)$ possesses a number of advantages over searching directly for $\mathbf{S}(t)$, where $\mathbf{S}(t)$ is some piecewise polynomial. First of all, our formulation guarantees positive definiteness of $\mathbf{S}(t)$ between individual knot points, since $\Phi(t)$ and $\mathbf{S}_0(t)$ are both positive definite and the Hadamard product of two positive definite matrices is always positive definite. While this would be also guaranteed for a piecewise linear representation of $\mathbf{S}(t)$, it would not be guaranteed for higher order piecewise polynomials. Secondly, our formulation also guarantees that, when $\Phi(t)$ is some small enough scalar multiple of the all ones matrix, a feasible solution will exist for our SOS optimization. This feature is important if we are to use the initial $\rho(t)$ from our previous approach as a starting point for the bilinear alternations. Last of all, using $\mathbf{S}_0(t)$ as a baseline whose value can be evaluated with high precision at any time sample between $[t_i, t_f]$ enables us to obtain a more dynamic representation for $\mathbf{S}(t)$ than could be achieved using a piecewise linear formulation with the same number of decision parameters and sample points. Initially, we were concerned that our representation for $\mathbf{S}(t)$ could suffer from numerical problems since as $\Phi(t)$ approaches the all ones matrix, the solution will lie on the edge of the semi-definite cone if $\Phi(t)$ is constrained to be positive definite. However, in practice, we have not found numerical conditioning to be a problem.

We also explored searching directly over a piecewise linear representation of $\mathbf{S}$(t) as well as searching over the form $\mathbf{S}(t) = \Sigma(t)^T \Phi(t) \Sigma(t)$ where $\mathbf{S}_0(t) = \Sigma(t)^T \Sigma(t)$. However, in both cases, the optimization routine produced much smaller final funnel volumes. Furthermore, in the case of the piecewise linear representation, the funnel was not conservative when compared to numerical simulations.

B. Cost Function

To use our S-matrix parameterization in a SOS optimization routine, we used a cost function to guide our search. Ideally, we would prefer our cost function to be proportional to funnel volume. Unfortunately, however, the volume of an ellipsoid is nonlinear in the matrix decision parameters and therefore not amenable to our SOS optimization routine. Therefore, we explored a linearizion of this cost function about a nominal $\mathbf{P}(t)$, where $\mathbf{P}(t) = \mathbf{S}(t)/\rho(t)$ and updated this operation point during each iteration of the optimization routine by using $\mathbf{P}(t)$ from the previous iteration.

Knowing that

$$\text{vol}(\mathscr{F}) \propto \int_0^t \sqrt{\det(\mathbf{P}(t)^{-1})}dt = \int_0^t \frac{1}{\sqrt{\det(\mathbf{P}(t))}}dt, \tag{32}$$

we can approximate this funnel volume by linearizing the nonlinear part of the function by some operating point $\mathbf{P}_0(t)$. Dropping the dependence on t we have:

$$\det(\mathbf{P})^{-\frac{1}{2}} \approx \det(\mathbf{P}_0)^{-\frac{1}{2}} + (-\tfrac{1}{2})\det(\mathbf{P})^{-\frac{3}{2}} \frac{d\det(\mathbf{P})}{d\mathbf{P}}\bigg|_{\mathbf{P}=\mathbf{P}_0} \Delta\mathbf{P}. \tag{33}$$

Using the identity,

$$d\det(\mathbf{P}) = \text{tr}(\text{adj}(\mathbf{P})d\mathbf{P}) = \text{tr}(\det(\mathbf{S})\mathbf{S}^{-1}d\mathbf{P}) \tag{34}$$

and letting $d\mathbf{P} = (\mathbf{P} - \mathbf{P}_0)$, we have

$$\det(\mathbf{P})^{-\frac{1}{2}} \approx \det(\mathbf{P}_0)^{-\frac{1}{2}}(1 - \tfrac{1}{2}\text{tr}(\mathbf{P}_0^{-1}\mathbf{P}) + \tfrac{1}{2}\text{tr}(\mathbf{P}_0^{-1}\mathbf{P}_0)). \tag{35}$$

We can now replace our initial cost function J with

$$J = \text{tr}(\mathbf{P}_0^{-1}\mathbf{P}), \tag{36}$$

where $\mathbf{P}_0$ is the P-matrix from the previous iteration step.

IX. Results

The funnels resulting from using the fully parameterized S-matrix are considerably larger, as shown in Figs. 7 and 8. A quantitative comparison of the resulting volume is given in Fig. 6, which reveals the improvement from using both the additional parameters and the linearized-volume cost function. Table I compares other details, such as computation time and number of iterations.

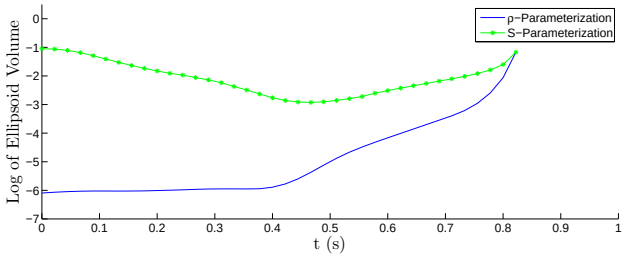

Fig. 6. Comparison of the volumes of the funnels obtained from optimizing over $\rho(t)$ compared with optimizing over $\Phi(t)$.

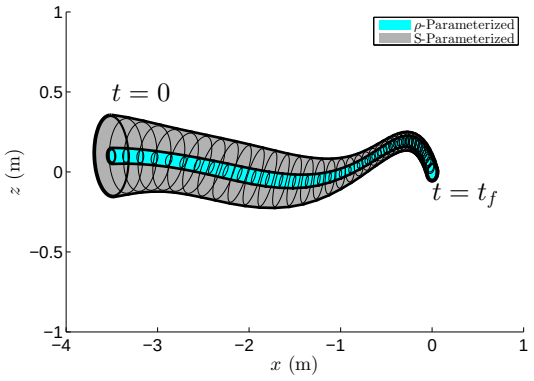

Fig. 7. Slice of the high-dimensional funnel in the x,z plane obtained by optimizing over $\rho(t)$ (cyan), compared with the funnel obtained by optimizing over $\Phi(t)$ (gray). The nominal trajectory for the plane begins at x=-3.5m and proceeds to the perch location x=0m, z=0m.

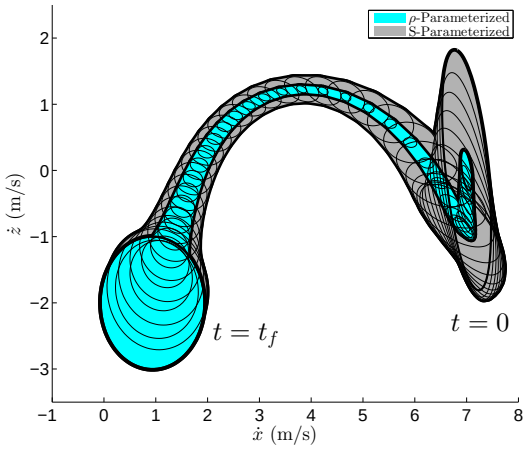

Fig. 8. Slice of the high-dimensional funnel in the $\dot{x}$,$\dot{z}$ plane obtained by optimizing over $\rho(t)$ (cyan), compared with the funnel obtained by optimizing over $\Phi(t)$ (gray). Note that the nominal trajectory begins *on the right* with a positive velocity, and ends on the left with near zero velocity.

A. Comparison with Simulation

To get an idea of how tight these funnels are to the actual time-varying invariant set, simulations were carried out and plotted against the invariant regions produced by the convex bilinear alternations at different sample times. The results of this simulation can be seen in Figs. 9-12. Given all of the potential conservatism taken along the length of the trajectory, we consider the resulting estimate of the funnels to be surprisingly tight. Note that simulating all relevant initial conditions from the higher dimensional space, and along the length of the trajectory, would be effectively intractable, which is why we limit ourselves to two dimensional slices.

TABLE I
FUNNEL COMPARISON

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

X. LQR-TREES

To improve the performance of the time-varying LQR controller, it would be desirable to cover the entire range of initial conditions. However, this can not be achieved with only one funnel. A possible solution is to use a finite number of trajectories to cover as much of the desired initial conditions as possible. An algorithm for achieving this, known as LQR-Trees, was proposed in [25]. Following this algorithm, we first initialize the tree with the final goal region. Then, we sample randomly from our set of initial conditions. For each sample point that does not fall in a verified region, we compute a new optimal trajectory to the goal state. Next, we verify the trajectory by computing a funnel and add this new verified region to the tree. We then repeat this process until all sample points from the set of initial conditions we wish to cover fall in the verified region.

To ensure that all our our funnels end in the desired goal region, we first search for $\rho(t_f)$ such that

$$\lambda(1-G)+(\bar{\mathbf{x}}^T\mathbf{S}(t_f)\bar{\mathbf{x}}-\rho(t_f))\geq 0 \tag{37}$$

where $\lambda > 0$. In order to make sure that this did not make the funnels too small, we slightly relaxed the goal region to $Q_f = diag([400,400,\frac{1}{9},\frac{1}{9},\frac{4}{9},\frac{4}{9},\frac{1}{9}])$ and we discarded any funnel where $\rho(t_f) < 0.6$.

The resulting LQR-Trees can be found in Figs. 13 and 14. We observed that, although the final value constraints for our optimal trajectories were set to be anywhere along the LQR-Tree, most of the trajectories connected to the goal region instead of connecting to a verified region somewhere along the tree's other branches. We hypothesize that this is due to the relatively short flight time of our glider.

XI. DISCUSSION

Our proposed method of controller synthesis and verification has a number distinct advantages. First of all, our approach preserves continuity in state by eliminating the need to solve a partial differential equation. In addition, when the Lagrange multipliers are of fixed order, our method scales polynomially with dimension, thus alleviating the curse of dimensionality to some extent. In fact, recently funnels for systems up to $\mathbb{R}^{10}$ have been computed in under a few hours.

One of the greatest shortcomings of our method is that we must approximate the nonlinear dynamics of our system as a third-order polynomial system. However, from Figs. 10 and 12, it seems as though the polynomial dynamics are a very good approximation of the glider dynamics about the nominal trajectory since for the entire flight, the funnels exhibit both tightness and conservatism when compared to the simulated invariant sets. Another disadvantage of this approach is that, while it can adequately handle variations in initial conditions, it can not verify systems with model inaccuracies or process noise. Without a doubt, in outdoor environments managing stochastic wind gusts will be critical for achieving a successful perching maneuver. Moreover, even in deterministic circumstances, it is nearly impossible to build aerodynamic models capable of representing all of the different flow regimes which can occur along the wing. It is interesting to note, however, that if formulated correctly, the verification procedure can be modified to meet specified robustness criteria as shown in [12] and [23].

XII. CONCLUSION AND FUTURE WORK

In this paper, we have demonstrated that it is possible to find regions of attraction for a fixed-wing UAV executing a post-stall perching maneuver. By parameterizing our Lyapunov function as the $\mathbf{S}(t)$ from the Riccati equation pointwise multiplied by a positive semi-definite parameter matrix, $\Phi(t)$, larger funnels were able to be computed than what had been originally computed using the ρ-parameterization. Using grid-based simulation, we then proceeded to show that, while not completely covering the invariant set, the time-varying basins found using SOS were a sufficiently large, conservative approximation of the true region. Finally, we

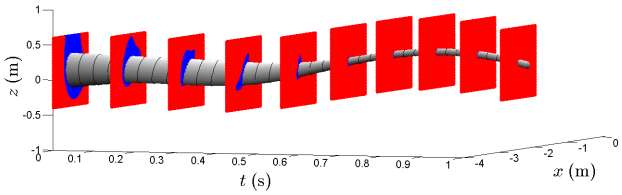

Fig. 9. Comparison of SOS funnel's x-z slice with the invariant set found by simulating the full non-polynomial, nonlinear system forward from the relevant initial conditions.

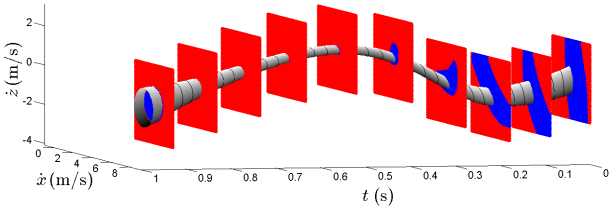

Fig. 11. Comparison of SOS funnel's $\dot{x}$-$\dot{z}$ slice with the invariant set found by simulating the full non-polynomial, nonlinear system forward from the relevant initial conditions.

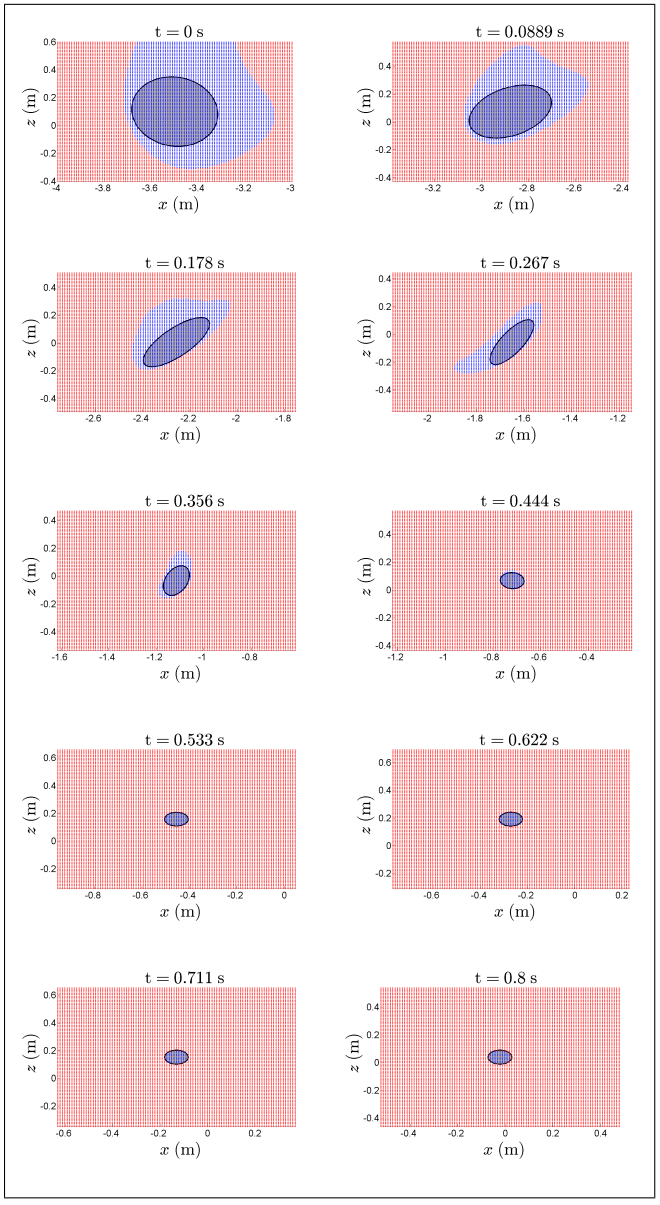

Fig. 10. 2D plots of the x-z funnel slice time cross-sections found in 9. Red represents simulated trajectories which failed to reach the goal region, blue represents those that succeeded, and gray represents the final SOS funnel.

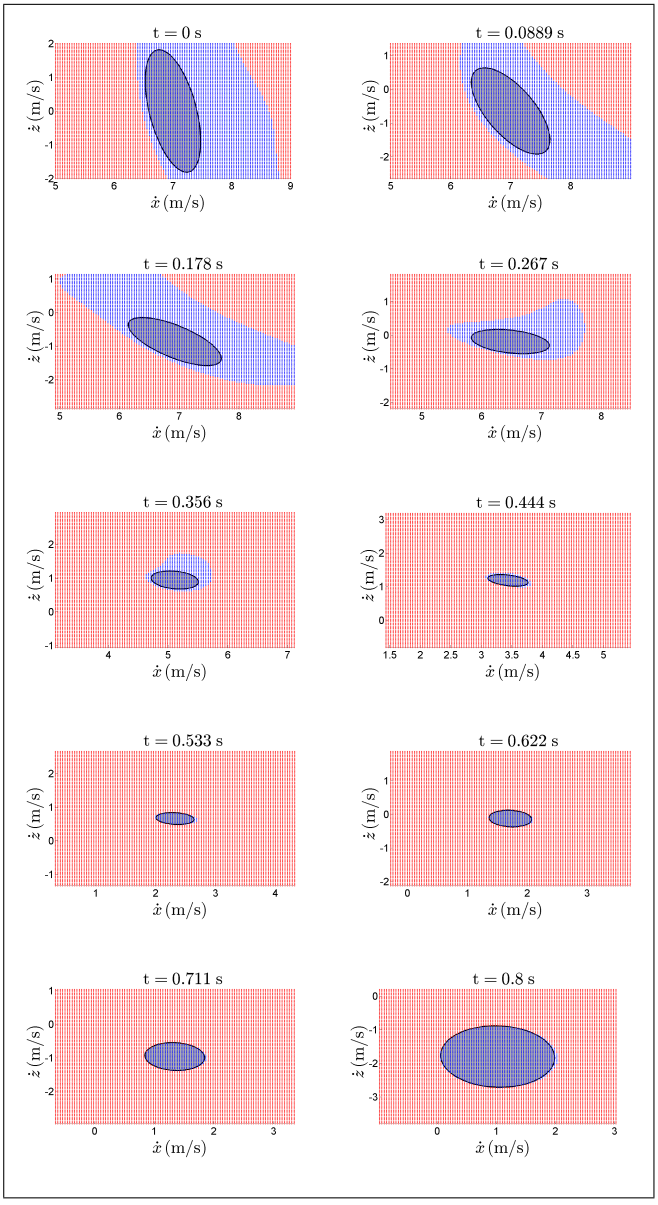

Fig. 12. 2D plots of the $\dot{x}$-$\dot{z}$ funnel slice time cross-sections found in 11. Red represents simulated trajectories which failed to reach the goal region, blue represents those that succeeded, and gray represents the final SOS funnel.

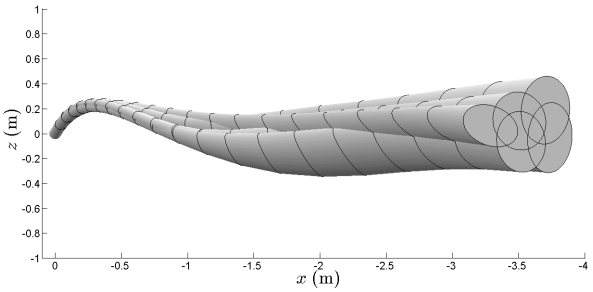

Fig. 13. LQR Tree Position Coverage

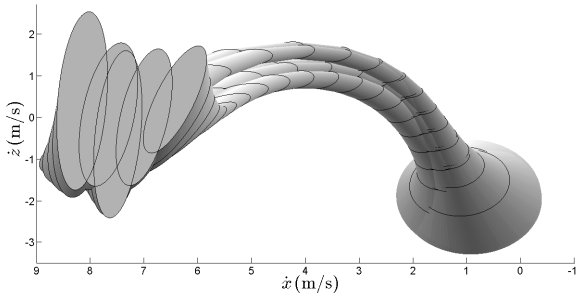

Fig. 14. LQR Tree Velocity Coverage

showed that our verification method could be used to build a tree of verified trajectories via the LQR-Trees algorithm, stabilizing a wider range of initial conditions. In the future, we hope not only to demonstrate the LQR-Trees algorithm experimentally on our fixed-wing glider, but to extend our approach to handle cases of external disturbances and model uncertainty. It is our belief that the LQR-Trees algorithm and its associated verification procedures will prove to be a general solution to a large number of nonlinear aerospace controls problems and we hope that as the method matures, it will be a significant step towards developing highly agile bird-scale UAVs.

ACKNOWLEDGEMENTS

This work was supported by NSF award 0915148 and ONR MURI contract N00014-09-1-1051.

REFERENCES

[1] Rick Cory. *Supermaneuverable Perching*. PhD thesis, MIT, June 2010.

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

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

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

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

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

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

[8] 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.

[9] Elena Leah Glassman, Alexis Lussier Desbiens, Mark Tobenkin, Mark Cutkosky, and Russ Tedrake. Region of attraction estimation for a perching aircraft: A Lyapunov method exploiting barrier certificates. In *Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

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

[11] J. Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[12] Anirudha Majumdar, Mark Tobenkin, and Russ Tedrake. Algebraic verification for parameterized motion planning libraries. In *Under review*, 2012.

[13] A. Megretski. Systems polynomial optimization tools (SPOT), available online: `http://web.mit.edu/ameg/www/`. 2010.

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

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

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

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

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

[19] 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.

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

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

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

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

[24] 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.

[25] 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.

[26] 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.

[27] 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.

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

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