

System Identification of Post Stall Aerodynamics for UAV Perching

Warren Hoburg* and Russ Tedrake†

MIT Computer Science and Artificial Intelligence Lab, Cambridge, MA, 02139, USA

For a UAV to perch on a wire, aircraft control systems which operate far outside typical operating envelopes must be developed. The relevant transient aerodynamics at high angle of attack are not addressed today by control-accessible aerodynamic models. In this work, we present a set of physically-inspired basis functions which have enabled system identification of a nonlinear aerodynamics model along perching trajectories. Data is collected using a motion capture system which, critically, allows free-flight data from real system trajectories to be gathered. When simulated forward, the identified model accurately predicts the observed perching trajectories, making it an indispensable tool for designing feedback controllers that stabilize perching trajectories.

Nomenclature

\mathbf{x}	state vector = $[x \ z \ \theta \ \phi \ \dot{x} \ \dot{z} \ \dot{\theta} \ \dot{\phi}]^T$
x, z	position of CG in world coordinates
θ	pitch angle
ϕ	elevator angle
α	wing angle of attack
V	total velocity
s	subset of basis function indices
$(l_e + l_h)$	distance from CG to elevator, m
n	number of basis functions in model
u	servo command
$\hat{\ddot{x}}$	predicted x acceleration (world coords), m/s
β_i^x	linear contribution of basis function i to $\hat{\ddot{x}}_p$
\mathbf{Q}	state-wise weighting for simulation error
<i>Subscript</i>	
i	Variable number
p	plane coordinates
el	elevator

I. Introduction

Birds routinely execute maneuvers that take them far outside the operating bounds of today's aircraft control systems. Designing a UAV that perches like a bird represents a formidable task in control system design and aerodynamics modeling. In order to strike a perch with small horizontal and vertical velocities, a fixed-wing UAV glider must exploit pressure drag at high angles of attack to quickly decelerate while maintaining enough lift or upward momentum to stay aloft. The resulting trajectories are characterized by nonlinear and transient aerodynamics, and a lack of control-accessible first-principles models makes this a natural setting for identifying models using data from the real system. We propose that by choosing a set

*Research Staff, AIAA Student Member.

†Associate Professor, Department of Electrical Engineering and Computer Science, AIAA Member.

of physically-inspired nonlinear basis functions, we can identify a stable aerodynamics model which, when simulated, accurately predicts the trajectories followed by the real plane. Such a model is immediately useful for stabilizing trajectories using linear-time-varying (LTV) control, or other nonlinear control approaches. The choice of a small set of physically-inspired basis functions (as opposed to radial basis functions or barycentric interpolators) makes the model more likely to generalize for data outside the training set.

II. Relation to Previous Work

Cory and Tedrake have demonstrated successful perching of a small foam glider in a motion capture environment. In,¹ the plane was controlled using a feedback policy optimized on a coarse grid over state-space. The dynamics model was identified on the real plane, using a linear combination of barycentric interpolator basis functions to predict lift, drag, and moment coefficients as functions of angle of attack and elevator angle. It was also found that flat plate theory (equation 1 below) was a good first-order approximation to the identified lift and drag coefficient model.

$$C_L = 2 \sin(\alpha) \cos(\alpha) \quad (1)$$

$$C_D = 2 \sin^2(\alpha) \quad (2)$$

The current work essentially builds on the idea that instead of learning a model comprised of basis functions spaced on a grid, physically inspired basis functions can form the core model which is more likely to generalize to data outside the training set.

III. Experimental Setup

The current work uses the same setup as is described in detail in.¹ During a perching trajectory, we control the elevator deflection ϕ by setting a servo command u in a 50 Hz control loop. The plane is launched from a custom crossbow launcher at approximately 6 m/s into a Vicon MX motion capture environment, which uses reflective markers on the plane to track position and orientation at 120 Hz. This controlled environment, which provides sub-mm position data, has proven to be an effective setup for efficient system identification. We argue that collecting free-flight data on the real system is critical if an accurate dynamics model is to be learned from data. This is for three reasons: 1) much faster transients are possible, 2) acceleration-dependent terms are excited, and 3) lack of stag, mount, or wall interference.

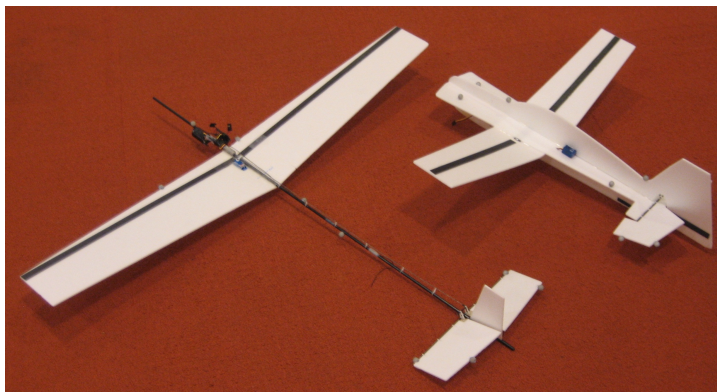


Figure 1. Two foam gliders: the original (right), and the newer version used in the current work (left)

Our fixed-wing glider has undergone a revision since the work of Cory and Tedrake.¹ The foam fuselage was eliminated and replaced with a carbon fiber rod, in hopes of better matching the assumptions of a flat plate wing/elevator model. To our surprise, we found that our new 'simpler' plane does not match equation 1 as well as the original plane. The reasons for this may be subtle, and we believe system identification on the new plane (with fewer aerodynamic surfaces) is most likely to reveal the most relevant dynamics for extrapolating to new planes, even if these dynamics are complicated. The two planes are shown in figure 1. The new plane weighs 90 g, has wing area 0.09 m^2 , elevator area 0.012 m^2 , and has a small dihedral

angle and vertical stabilizer to provide passive roll and yaw stability, thus constraining our problem to 2D longitudinal (pitch) dynamics.

IV. System Identification

We seek to identify a model for the aerodynamics along an observed perching trajectory. To do so, we gather data by repeatedly firing the plane and executing a hand-tuned open-loop control tape which gets the plane near the perch (on average, with some standard deviation). After preprocessing the data from our motion capture system, we simply use least squares to fit a quasi-steady model for acceleration in plane coordinates as a function of current state, \mathbf{x} . We fit accelerations instead of forces since added-mass effects tend to make the mapping from force to acceleration more complicated than linear in mass and inertia. The model for accelerations is a linear combination of physically-inspired basis functions we specify (in appendix A). Since we have a general feel for what sorts of terms to place in our basis functions, but don't know the exact aerodynamics, we specify approximately 50 possible basis functions, and then choose the ones that predict real data with the smallest residual. We aim to keep the number of basis functions in the model small (about 2-3 for each acceleration predicted) in order to minimize overfitting, which tends to make the model inaccurate in simulation.

IV.A. Preprocessing

We begin by filtering raw position and orientation data acausally with a 3rd order low-pass Butterworth filter. We then differentiate twice using a finite difference stencil to get instantaneous velocity and acceleration data, and finish the conversion from 3D to 2D data using the filtered data. Next, we do a coordinate transform to express the observed inertial accelerations in plane coordinates (normal and tangential to the wing), and remove gravity (since we want to predict accelerations due to aerodynamic forces only):

$$\begin{bmatrix} \ddot{x}_p \\ \ddot{z}_p \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{z} + g \\ \ddot{\theta} \end{bmatrix} \quad (3)$$

The choice of coordinates normal and tangential to the wing, as opposed to normal and tangential to velocity (lift and drag) is a break from standard aerodynamic practice, and is sensible because our wing is literally a flat plate, as opposed to an airfoil. We find that the aerodynamics normal and tangential to the wing are more 'decoupled' than the lift and drag aerodynamics.

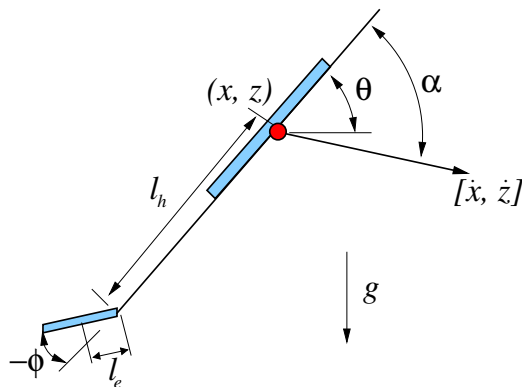


Figure 2. Longitudinal rigid body glider model. The blue segments represent the aerodynamic surfaces of the wings and the elevator, and the red dot represents the vehicle center of mass.

IV.B. Physically-Inspired Basis functions

Our basis functions are ultimately functions of the plane state \mathbf{x} , but it is helpful to first define a set of intermediate variables, which represent the wing and elevator velocities and angles of attack (ignoring downwash):

$$\dot{x}_{el} = \dot{x} + (l_e + l_h)\dot{\theta} \sin(\theta) \quad \dot{z}_{el} = \dot{z} - (l_e + l_h)\dot{\theta} \cos(\theta) \quad (4)$$

$$V = \sqrt{\dot{x}^2 + \dot{z}^2} \quad V_{el} = \sqrt{\dot{x}_{el}^2 + \dot{z}_{el}^2} \quad (5)$$

$$\alpha = \theta - \text{atan2}(\dot{z}, \dot{x}) \quad \alpha_{el} = \theta + \phi - \text{atan2}(\dot{z}_{el}, \dot{x}_{el}) \quad (6)$$

From here we define a number of physically-inspired functions of the above variables. The full list of our basis functions is in appendix A.

IV.C. Least-Squares Fit

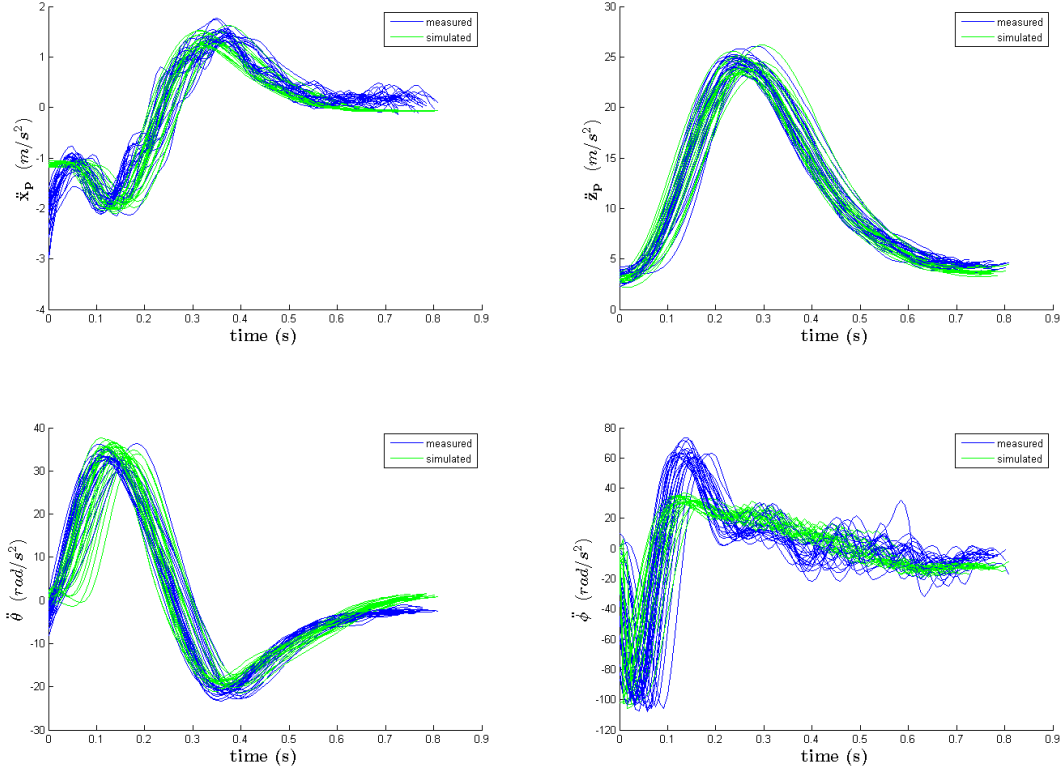


Figure 3. Fit of one-step acceleration data using least squares. Acceleration data is naturally noisy.

Given our basis functions ϕ_i , we wish to identify models for tangential, normal, and angular accelerations. The form we seek (written here for tangential acceleration) is a linear combination of nonlinear basis functions:

$$\hat{\ddot{x}}_p = \sum_{j=s} \beta_j^x \phi_j \quad (7)$$

Where s is the subset of n basis functions which minimizes the prediction residual $\sum_i (\hat{\ddot{x}}_i - \ddot{x}_i)^2$ where i indexes into individual samples. Similar equations give the models for $\hat{\ddot{z}}_p$ and $\hat{\ddot{\theta}}$. To choose the subset of n basis functions which minimizes one-step prediction error, we solve the least squares squares problem for every possible combination of n basis functions, and pick the one with minimum residual. This is a brute force approach, but is practical since we wish to keep n small (usually 1, 2, or 3) to avoid overfitting. The exact choice of n is hand-tuned by the authors, based on a combination of minimizing simulation error and

minimizing overfitting. Increasing n by one always decreases one-step residual, but may increase simulation error by making the model less stable. Overfitting creates quickly changing gradients in the model, which are undesirable if the model is later to be linearized for control purposes.

The specific model we have identified for our current glider is:

$$\hat{\dot{x}}_p = \beta_6^x \phi_6 + \beta_{24}^x \phi_{24} = \beta_6^x V^3 \cos(\alpha) + \beta_{24}^x V_{el}^2 \sin(\alpha_{el}) \sin(\phi) \quad (8)$$

$$\hat{\dot{z}}_p = \beta_1^z \phi_1 + \beta_6^z \phi_6 + \beta_9^z \phi_9 = \beta_1^z V^2 \sin(\alpha) + \beta_6^z V^2 \cos^3(\alpha) + \beta_9^z V \dot{\theta} \cos(\alpha) \quad (9)$$

$$\hat{\dot{\theta}} = \beta_{29}^\theta \phi_{29} + \beta_{30}^\theta \phi_{30} = \beta_{29}^\theta V^2 \sin \alpha \cos \alpha + \beta_{30}^\theta V_{el}^2 \sin(\alpha_{el}) \cos(\alpha_{el}) \cos(\phi) \quad (10)$$

To convert these plane frame accelerations back to world coordinates, we simply invert equation 3.

IV.D. Elevator Model

We also identify a second-order linear model for the elevator angle ϕ based on the lagged control input $u(t - \tau)$, where τ represents the delay associated with sensing and processing (approximately 28 ms). To identify the elevator models, we again use a least squares approach to identify the β_i^ϕ in:

$$\begin{bmatrix} \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \beta_1^\phi & \beta_2^\phi \end{bmatrix} \begin{bmatrix} \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \beta_3^\phi \end{bmatrix} u(t - \tau) \quad (11)$$

IV.E. Results: Prediction in Simulation

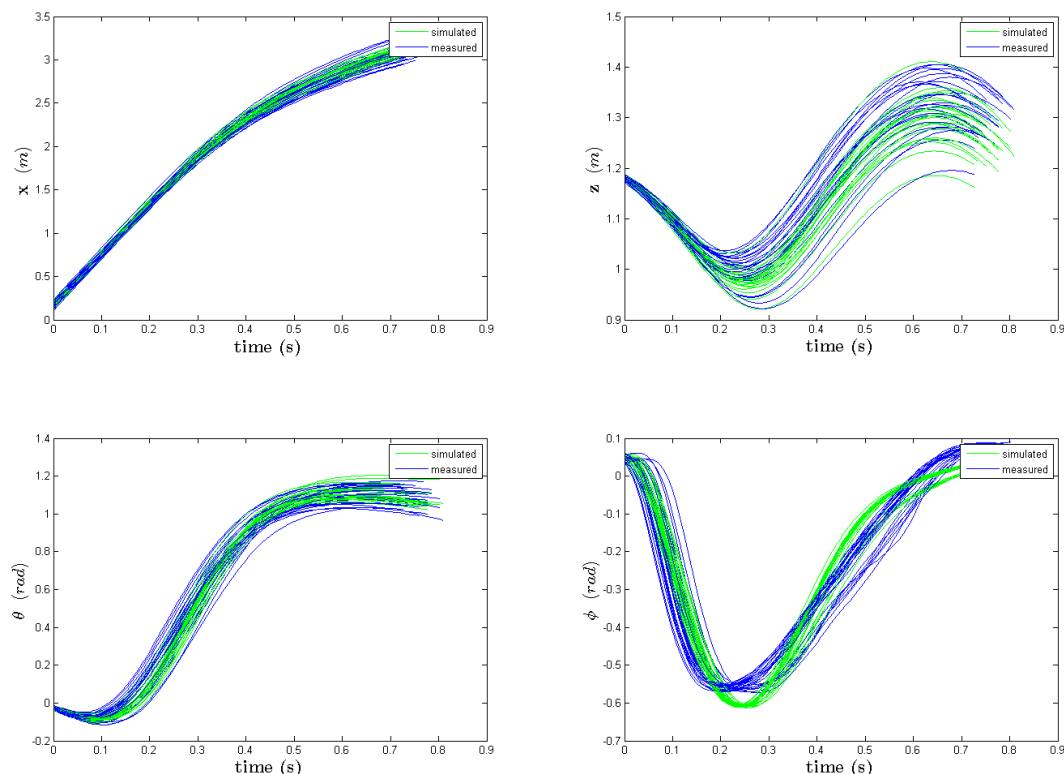


Figure 4. Accurate simulation of identified model forwards. Here each blue trace is an observed trajectory on the real glider, and has a corresponding green trace, which is the identified glider model simulated forward from the observed initial condition.

To evaluate our identified model, we attempt to use it to simulate forward from the same initial condition as the observed data, running the same open loop control tape. We pick as our initial condition a point in the trajectory where the elevator has begun to deflect, which removes transients associated with the plane leaving our launching mechanism. As shown in figure 4 the simulated and observed trajectories match closely, with a maximum error (over all trajectories) of 13 cm in x , 12 cm in z , 6.6 degrees in θ , and 5.6 degrees in ϕ , and with a mean error of 2 cm in x , 2 cm in z , 1.5 degrees in θ , and 1.5 degrees in ϕ . This implies that the model is stable, and therefore useful for control tasks such as LTV stabilization of the observed trajectory.

V. Model Refinement through Gradient Descent

While the model predicts the observed data well, we can do better through optimization of the right cost function. Above, we used a simple least squares approach to minimize the one-step prediction error $\sum_i (\hat{x}_i - \tilde{x}_i)^2$. However, ultimately we would like our model to minimize long term simulation errors, and we cannot expect minimizing one-step prediction error to correspond to minimizing simulation error. We define the simulation error as the (squared) error between observed and simulated trajectories which start at the same initial condition:

$$\int_0^T (\mathbf{x}_{sim} - \mathbf{x}_{meas})^T \mathbf{Q} (\mathbf{x}_{sim} - \mathbf{x}_{meas}) dt \quad (12)$$

Where \mathbf{Q} is a weighting on errors in each state variable - usually unity for x , z , and θ , and zero elsewhere. Starting from the least squares solution, we can tweak our model parameters (using gradient descent) to bring this simulation error to a local minimum. To do so, we must calculate the gradients of the simulation error cost function (equation 12) with respect to changes in the parameters of our model, β_i . This is accomplished using well-known methods, such as back-propagation through time (BPTT) or Real-Time Recurrent Learning (RTRL).²

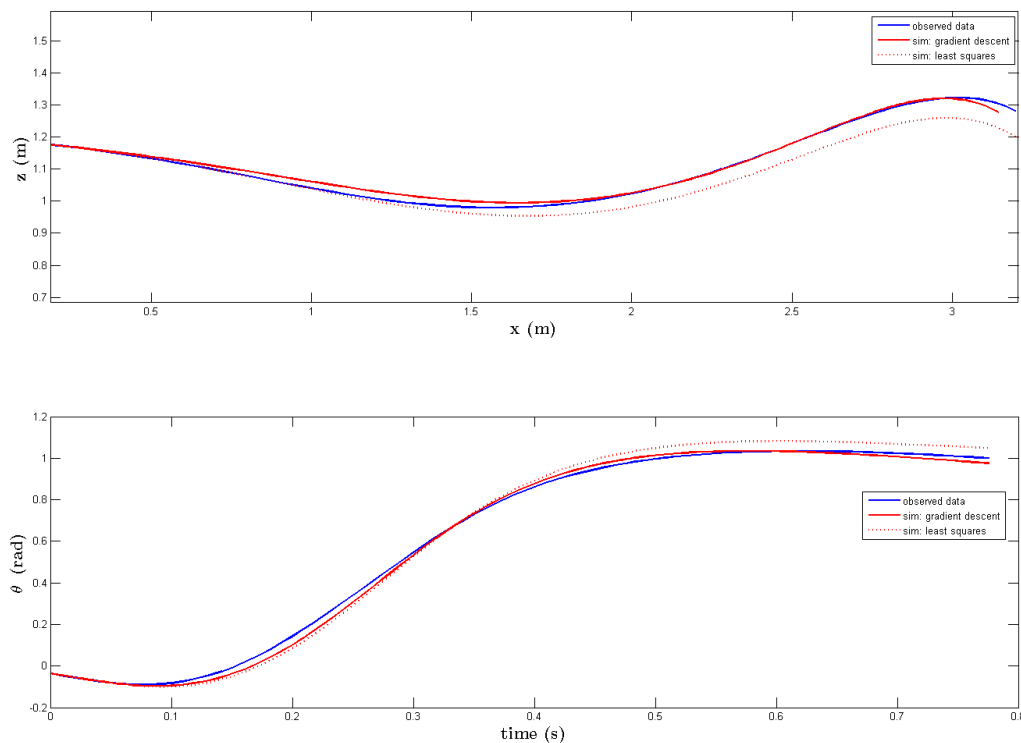


Figure 5. Reduction of model simulation error using gradient descent. The parameters are initialized to the least squares solution, and then tweaked using gradient descent to bring simulation error to a local minimum.

Using gradient descent to bring the simulated trajectory in closer agreement with a single observed trajectory (as shown in figure 5) is especially useful when attempting to stabilize a trajectory using feedback. Whether we choose to stabilize the observed or the simulated trajectory, discrepancies between the modeled and observed dynamics show up as disturbances, and minimizing them is critical for performance.

VI. Future Work: LTV Stabilization

Ultimately, the authors' modeling work is motivated by a desire to stabilize feasible perching trajectories using feedback. The feedback control we envision requires an accurate dynamics model (such as the one we have presented), and will enable perching from perturbed initial conditions and with disturbances such as gusts. One control approach which may be well suited to this problem is Linear Time Varying (LTV) stabilization of one or many desired trajectories.⁴ This is work in progress, but initial simulation results suggest LTV control using our identified model successfully reduces the final distance from the perch, when starting from a randomly perturbed initial condition.

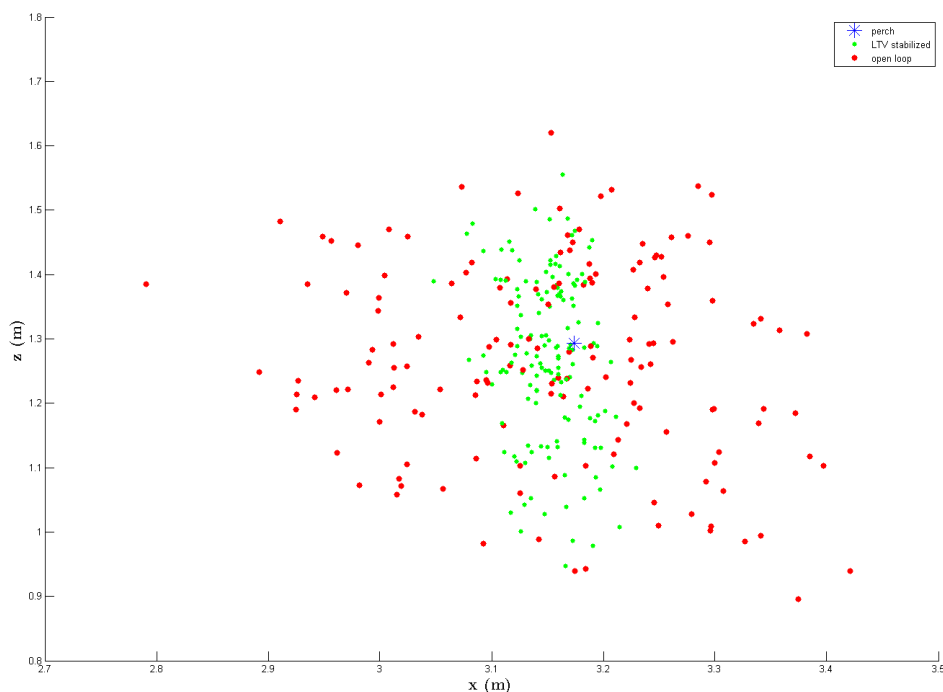


Figure 6. Stabilization of perching trajectory using LTV control. The glider dynamics were simulated forward from random initial conditions using 1) an open loop control tape (red dots), and 2) LTV control (green dots). Green dots are more tightly packed around the perch than red dots, showing that the LTV controller is working. Ongoing LTV research is essentially focused on clumping the green dots closer to the perch.

VII. Conclusions

We have presented a simple set of physically-inspired basis functions which have enabled us to identify a compact model for the aerodynamics of a glider following a specific perching trajectory. The model is identified from free-flight data in a motion capture environment, which ensures that we have observed all the relevant (transient) system dynamics. Starting with a least squares fit that minimizes one-step prediction error, the model is fine-tuned using gradient descent to minimize the simulation error over the entire perching trajectory. Initial results indicate that LTV control (trajectory stabilization) successfully brings the glider close to the perch from perturbed initial conditions.

Acknowledgments

The authors thank labmates Rick Cory, John Roberts, and Joseph Moore for their help with the experiments. This research was supported by the a Microsoft Research New Faculty Fellowship and by the MIT Lincoln Laboratory Advanced Concepts Committee.

References

- ¹Cory, R. and Tedrake, R., “Experiments in Fixed-Wing UAV Perching,” *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, AIAA, 2008.
- ²Williams, R. J. and Zipser, D., “A Learning Algorithm for Continually Running Fully Recurrent Neural Networks,” *Neural Computation*, Vol. 1, 1989, pp. 270–280.
- ³Williams, D., Collins, J., Quach, V., Kerstens, W., Buntain, S., Colonius, T., Tadmor, G., and Rowley, C., “Low Reynolds Number Wing Response to an Oscillating Freestream with and without Feed Forward Control,” slides from talk at AIAA ASM Orlando 2009, 2009.
- ⁴Roberts, J. W., Cory, R., and Tedrake, R., “On the Controllability of Fixed-Wing Perching,” *Accepted in the Proceedings of the American Controls Conference (ACC)*, 2009.
- ⁵Anderson, A., Pesavento, U., and Wang, Z. J., “Unsteady aerodynamics of fluttering and tumbling plates,” *J. Fluid Mech.*, Vol. 541, 2005, pp. 65–90.
- ⁶Tedrake, R., Jackowski, Z., Cory, R., Roberts, J. W., and Hoburg, W., “Learning to Fly like a Bird,” *Under review*, 2009.
- ⁷Wickenheiser, A. M. and Garcia, E., “Optimization of Perching Maneuvers Through Vehicle Morphing,” *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 4, July-August 2008, pp. 815–824.
- ⁸Hoburg, W., Roberts, J. W., Moore, J., and Tedrake, R., “The Perching Number: A Dimensionless Analysis of Post-stall Maneuvering in Birds and Planes,” *Working Draft*, 2009.
- ⁹Tedrake, R., “LQR-Trees: Feedback motion planning on sparse randomized trees,” *Under review*, 2009.
- ¹⁰Langelaan, J. W. and Bramesfeld, G., “Gust Energy Extraction for Mini- and Micro- Uninhabited Aerial Vehicles,” *Proceedings of the 46th Aerosciences Conference*, 2008, p. 15.
- ¹¹Andreas Schtte, G. E. and Raichle, A., “Numerical Simulation of Maneuvering Aircraft by Aerodynamic, Flight Mechanics and Structural Mechanics Coupling,” *Journal of Aircraft*, Vol. 46, Jan/Feb 2009, pp. 53–64.

A. Full List of Basis Functions

$$\begin{aligned}
 \phi_1 &= V^2 \sin(\alpha) & \phi_{22} &= V_{el} \dot{\phi} \\
 \phi_2 &= V^2 \cos(\alpha) & \phi_{23} &= V_{el}^2 \sin(\alpha_{el}) \cos(\phi) \\
 \phi_3 &= V^2 \sin(\alpha) \cos^2(\alpha) & \phi_{24} &= V_{el}^2 \sin(\alpha_{el}) \sin(\phi) \\
 \phi_4 &= V^2 \sin^2(\alpha) \cos(\alpha) & \phi_{25} &= V_{el} \dot{\phi} \sin(\alpha_{el}) \\
 \phi_5 &= V^2 \sin^3(\alpha) & \phi_{26} &= V_{el} \dot{\phi} \cos(\alpha_{el}) \\
 \phi_6 &= V^2 \cos^3(\alpha) & \phi_{27} &= V_{el}^2 \sin(\alpha_{el}) \cos(\alpha_{el}) \\
 \phi_7 &= V \dot{\theta} & \phi_{28} &= V \dot{\theta} \sin(\alpha) \cos(\alpha) \\
 \phi_8 &= V \dot{\theta} \sin(\alpha) & \phi_{29} &= V^2 \sin(\alpha) \cos(\alpha) \\
 \phi_9 &= V \dot{\theta} \cos(\alpha) & \phi_{30} &= V_{el}^2 \sin(\alpha_{el}) \cos(\alpha_{el}) \cos(\phi) \\
 \phi_{10} &= \dot{\theta}^2 & \phi_{31} &= V_{el}^2 \sin(\alpha_{el}) \cos(\alpha_{el}) \sin(\phi) \\
 \phi_{11} &= \dot{\theta} & \phi_{32} &= \dot{\theta} |\dot{\theta}| \\
 \phi_{12} &= \sin(\phi) & \phi_{33} &= V \dot{\theta} |\dot{\theta}| \\
 \phi_{13} &= V^2 \sin(\phi) & \phi_{34} &= \dot{\theta} |\dot{\theta}| \sin(\alpha) \cos(\alpha) \\
 \phi_{14} &= V \dot{\theta} \sin(\phi) & \phi_{35} &= \dot{\theta} |\dot{\theta}| \sin(\alpha) \\
 \phi_{15} &= V \dot{\phi} \sin(\phi) & \phi_{36} &= \dot{\theta} |\dot{\theta}| \cos(\alpha) \\
 \phi_{16} &= V (\dot{\theta} + \dot{\phi}) \sin(\phi) & \phi_{37} &= \dot{\theta}^2 \sin(\alpha) \cos(\alpha) \\
 \phi_{17} &= V \dot{\theta} \cos(\phi) & \phi_{38} &= V_{el} (\dot{\theta} + \dot{\phi}) \\
 \phi_{18} &= V (\dot{\theta} + \dot{\phi}) \sin(\phi) \cos(\phi) & \phi_{39} &= V_{el} (\dot{\theta} + \dot{\phi}) \sin(\alpha_{el}) \\
 \phi_{19} &= \sin(\alpha_{\phi}) & \phi_{40} &= V_{el} (\dot{\theta} + \dot{\phi}) \cos(\alpha_{el}) \\
 \phi_{20} &= V_{el}^2 \sin(\alpha_{el}) & \phi_{41} &= V_{el} (\dot{\theta} + \dot{\phi}) \sin(\alpha_{el}) \cos(\alpha_{el}) \\
 \phi_{21} &= V_{el} \dot{\theta} \sin(\alpha_{el}) & &
 \end{aligned}$$