Inverting Learned Dynamics Models for Aggressive Multirotor Control

Alexander Spitzer and Nathan Michael

{spitzer, nmichael}@cmu.edu

Problem

How can we follow trajectories accurately with a multirotor in the presence of disturbances and unmodeled dynamics?

Challenges:

- External disturbances
- Poorly characterized vehicle
- Unmodeled, complex dynamics
- Time-varying dynamics / degradation

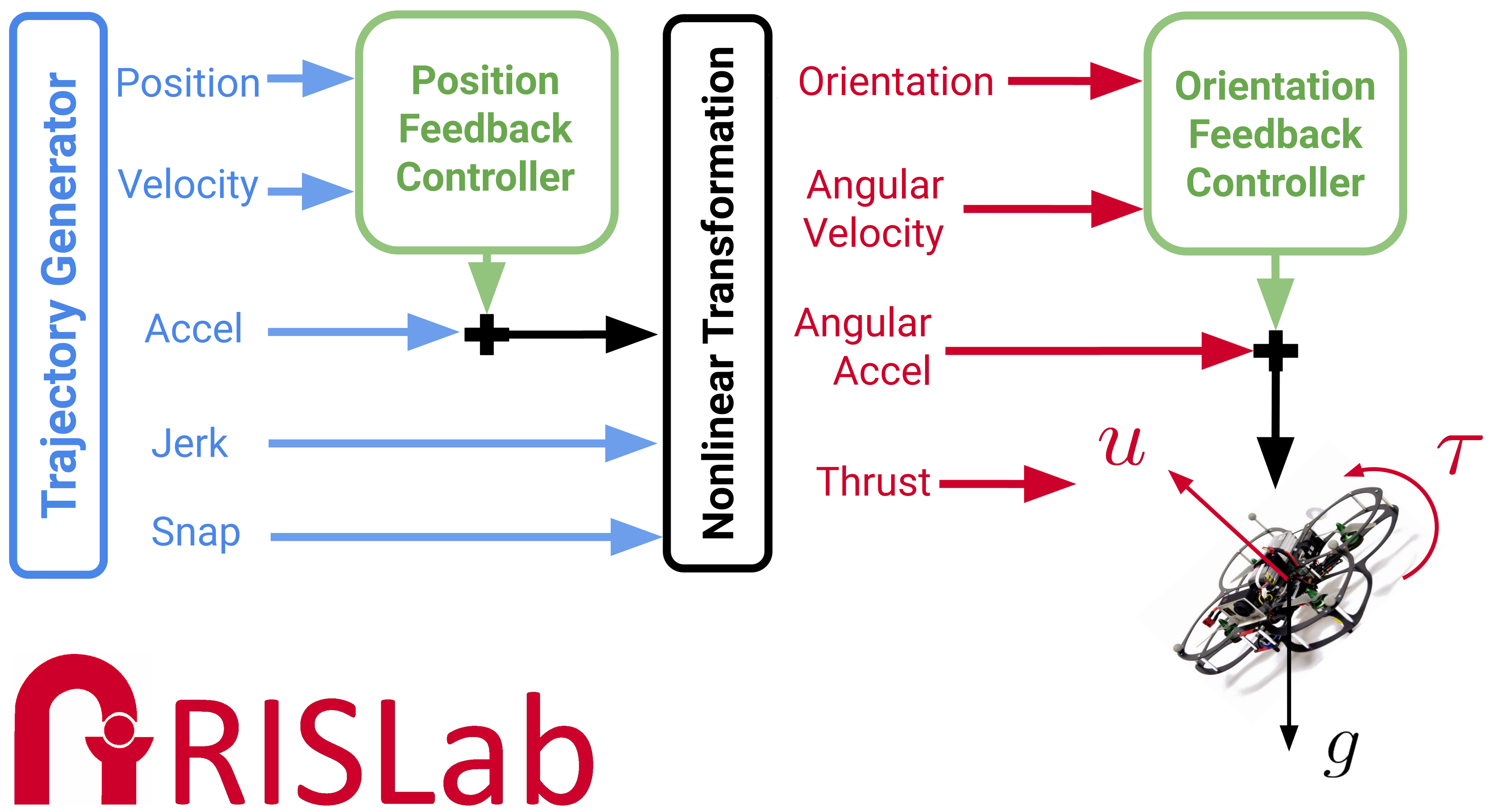
Background

Traditional multirotor control strategies **convert smooth trajectories into thrust, orientation, angular velocity, and angular acceleration using differential flatness**. The orientation and angular velocity are used as setpoints in the orientation feedback controller, while the thrust and angular acceleration are directly used as control inputs to the vehicle. This transformation uses a simple acceleration model that only considers acceleration from the motors and gravity. Since this **model is often incomplete and inaccurate,** the generated control inputs are not accurate and **tracking performance suffers**.

Acceleration Model

$a = u + g$

Smooth Trajectory (accel., jerk, and snap) → Control Inputs (thrust, angle, ang. vel., ang. accel.)



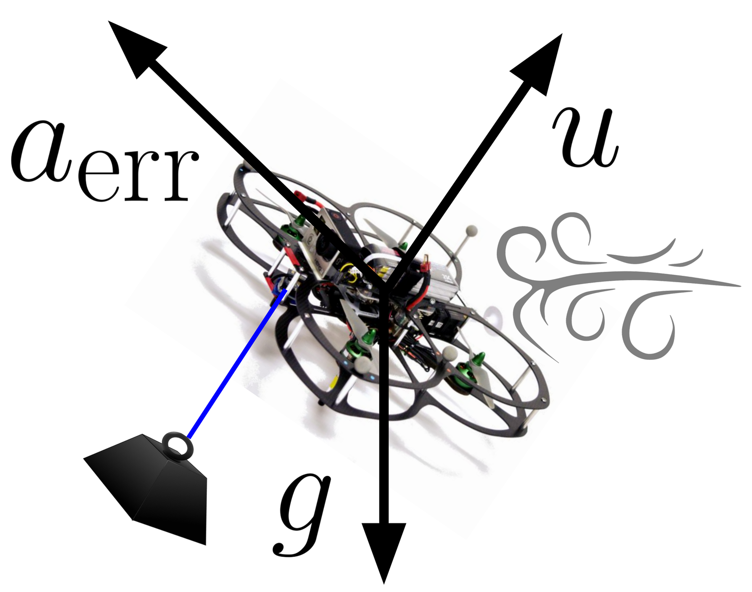
Solution

1. **Correct acceleration model** by learning to predict vehicle accelerations.
2. **Solve acceleration balance** for thrust vector.
3. **Invert learned dynamics** model to correct higher-order orientation references.

Model Learning

Learn additive correction to acceleration model:

$$a = u + g + a_{\text{err}}$$



Single-step velocity difference to measure acceleration:

$$a_{\text{err}} = \frac{\dot{x}(t+T) - \dot{x}(t)}{T} - (u+g)$$

Incrementally update linear function of state and input features:

$$a_{\text{err}} = w^\top \phi(x, u)$$

Simulation: $\phi(x,u) = (x \;\; \dot{x} \;\; \sin(\theta) \;\; u)$

Hardware: $\phi(x,u) = (\dot{x} \;\; u)$

Acceleration Balance

Accurate control inputs require solving the acceleration equation numerically for the thrust vector u when there is a dependence on the control input.

find u s.t. $\quad a = u + g + a_{\text{err}}(u, t)$

This is solved efficiently by warm starting the Newton-Raphson root-finding method.

Model Inversion

First, compute thrust vector's first and second time derivative.

$$\dot{u} = \dot{a} - \dot{a}_{\text{err}} \qquad \ddot{u} = \ddot{a} - \ddot{a}_{\text{err}}$$

Then, compute orientation references from thrust vector and its derivatives. The orientation is set to align to the thrust vector.

orientation $\quad R = f(u)$ $\qquad$ angular velocity $\quad \omega = f'(u)\dot{u}$

angular acceleration $\quad \alpha = \left(f''(u)\dot{u}\right)\dot{u} + f'(u)\ddot{u}$

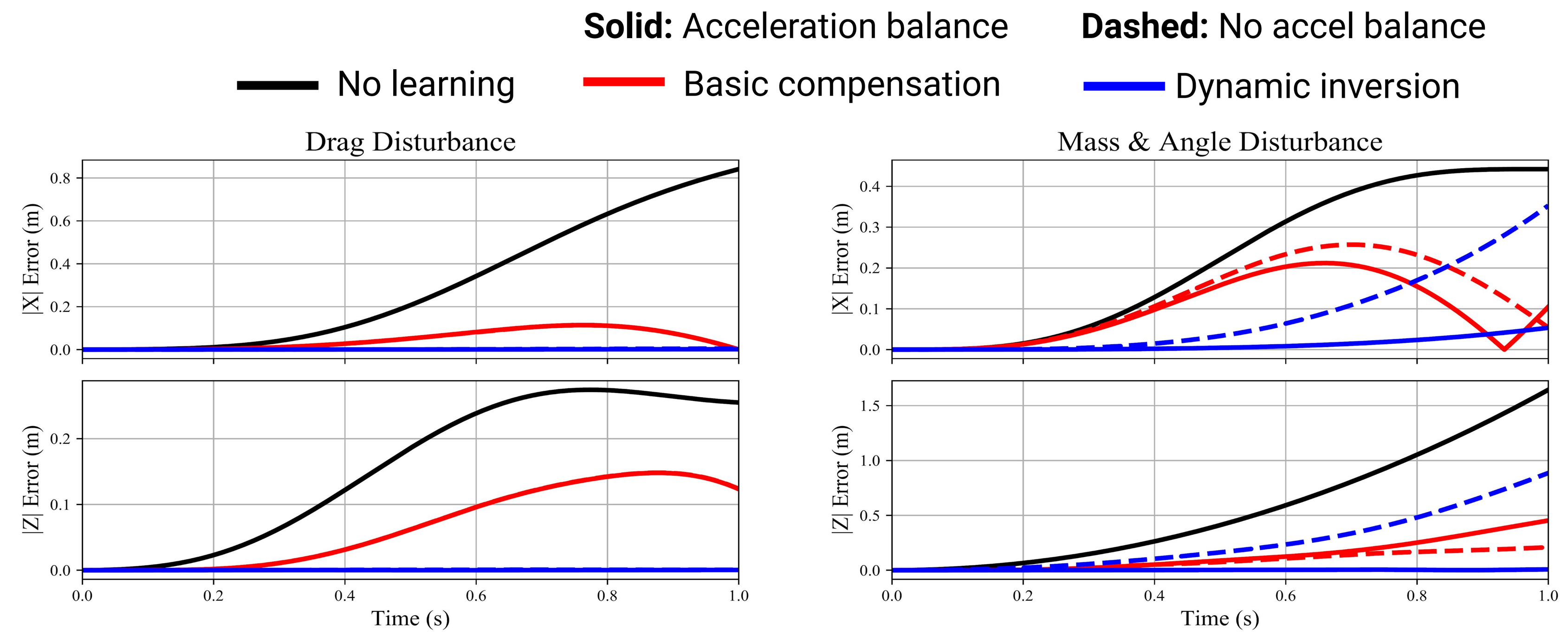
Accounting for learned dynamics requires querying learned model's 1st and 2nd derivatives.

Results

Accounting for learned acceleration model dynamics by correcting orientation references improves tracking performance.

Simulation

- Straight line trajectory
- Two training trajectories enough for learned model to converge



Dynamic inversion is required for perfect tracking in the presence of drag.

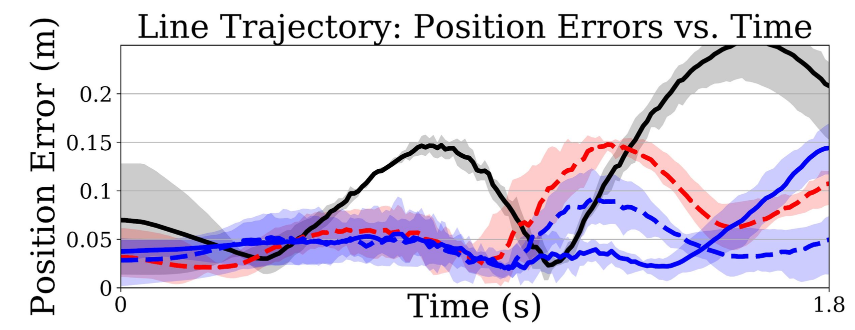
Acceleration balance is required for perfect tracking in the presence of input-dependent disturbances.

Hardware

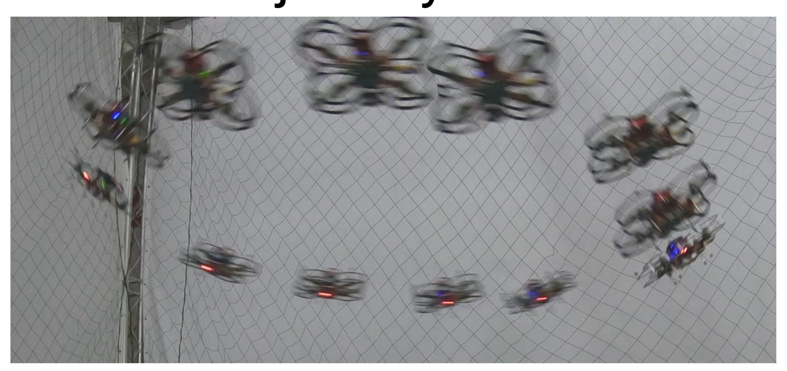
We test our approach in hardware on a 750 g quadrotor inside a motion capture arena.

Line trajectory: 6.3 m / s^2

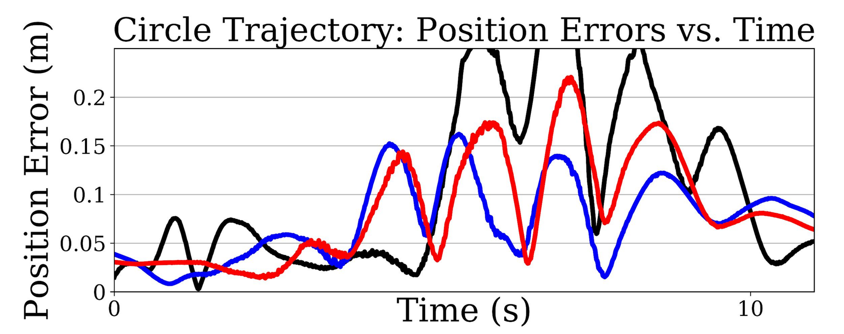
30% reduction in tracking error



Circle trajectory: 6.2 m / s^2



27% reduction in tracking error



RISLab

Carnegie Mellon University

The Robotics Institute