Sliding Mode Quadrotor Control

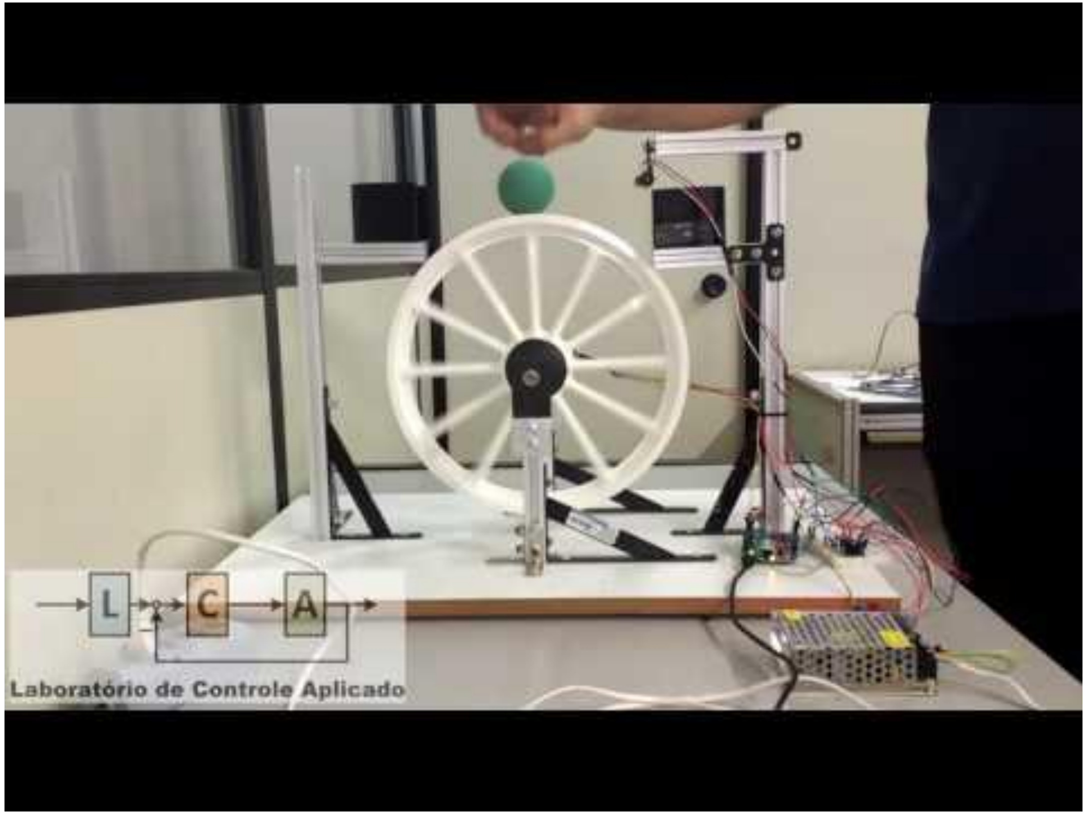
Jesse Wynn

Based on Herrera et. al. “Sliding Mode Control: An approach to Control a Quadrotor”

Introduction

Sliding Mode Control (SMC) is a common nonlinear control technique which uses a non-continuous control signal to drive the system to a subset of the state-space that has desirable characteristics. This space is called the ***Sliding Mode*** or ***Sliding Surface*** ($s = 0$).

Some Motivating Examples



Some Motivating Examples...

Pros of SMC

- Robust
  - Need not be precise
  - Not very sensitive to parameter variations
- The sliding mode (s = 0) can be reached in finite time
  - SMC is a non-continuous control method
  - Better than asymptotic convergence
    - However once on the sliding surface, the system inherits the stability properties of the system on that surface (asymptotic, exponential, etc.)
- Particularly well suited in switching power converter circuits where switching behavior is already naturally present

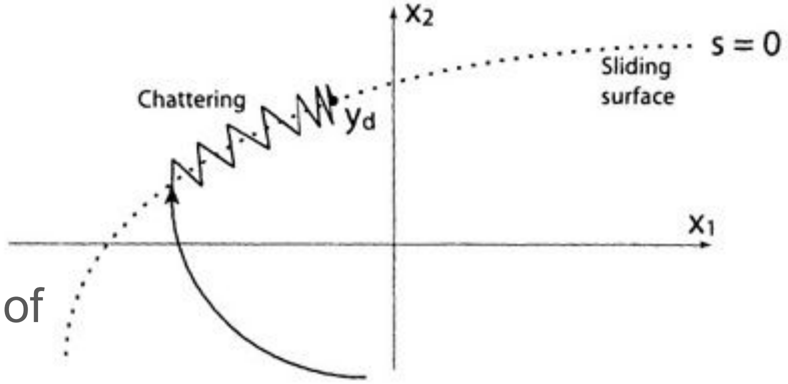
Cons of SMC...

Cons of SMC...

Chatter...

Cons of SMC

- Chatter or ‘chattering’ in the neighborhood of the sliding surface
- Can destroy actuators of physical systems
- Works better in theory
    - Instantaneous actuation can’t actually happen on physical systems
- Can excite unmodeled dynamics or modes of the system



Should SMC even be used for quadrotors???

Probably not ...

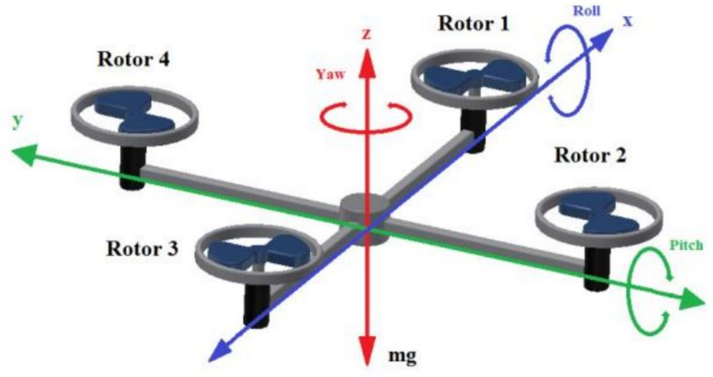
Should SMC even be used for quadrotors???

But let’s do it anyway.

- Quadrotors have a lot of natural damping
  - Viscous damping and slip of propeller spinning through the air
  - Not like the meshing of teeth between gears
- Motor and spinning propellers are small and don’t have tons of mass or inertia (relatively)
- Typical motors (without active braking) have a smooth ‘spin down’ that softens things out

The Paper: Dynamics

Standard quadrotor dynamics except FLU instead of NED



$$X = [x\ \dot{x}\ y\ \dot{y}\ z\ \dot{z}\ \phi\ \dot{\phi}\ \theta\ \dot{\theta}\ \psi\ \dot{\psi}\ ]$$

$$\ddot{x} = \frac{cos(\phi)sin(\theta)cos(\psi) + sin(\phi)sin(\psi)}{m}U_1$$

$$\ddot{y} = \frac{cos(\phi)sin(\theta)sin(\psi) - sin(\phi)cos(\psi)}{m}U_1$$

$$\ddot{z} = -g + \frac{cos(\phi)cos(\theta)}{m}U_1$$

$$\ddot{\phi} = \dot{\phi}\dot{\psi}\left(\frac{I_y - I_z}{I_x}\right) - \frac{Jr}{I_x}\dot{\theta}.\Omega + \frac{l}{I_x}U_2$$

$$\ddot{\theta} = \dot{\phi}\dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) + \frac{Jr}{I_y}\dot{\theta}.\Omega + \frac{l}{I_y}U_3$$

$$\ddot{\psi} = \dot{\theta}\dot{\phi}\left(\frac{I_x - I_y}{I_z}\right) + \frac{1}{I_z}U_4$$

The Paper: Sliding Surface Selection

"... the sliding surface or decision rule must be selected. This selection is made based on performance criteria, since the sliding surface equation determines the dynamics of the system. Therefore, it can be written as:

$$s = \dot{e} + \lambda e$$

The Paper: Controller Derivation (Altitude)

Control input *U* consists of a continuous and a discontinuous part:

$$U(t) = u_{eq}(t) + u_D(t)$$

The discontinuous part can be written as:

$$u_D(t) = k_D \, sign(s(t))$$

And to reduce the chattering effect:

$$u_D(t) = k_D \frac{s(t)}{|s(t)| + \delta}$$

The Paper: Controller Derivation (Altitude)

We find the continuous part of the control input by applying the sliding condition:

$$\dot{s} = 0$$

Then from our sliding surface equation we have:

$$\dot{s} = \ddot{e} + \lambda \dot{e} = 0 \quad \text{or} \quad \dot{s} = (\ddot{z}_d - \ddot{z}) + \lambda(\dot{z}_d - \dot{z})$$

And then plugging in our dynamics and solving for our input:

$$u_{eq} = [g + \lambda(\dot{z}_d - \dot{z}) + \ddot{z}_d\,] \frac{m}{\cos(\phi)\ \cos(\overset{\theta}{\cancel{\psi}})}$$

The Paper: Controller Derivation (Altitude)

Just like we did in class they define a Lyapunov function:

$$V = \frac{1}{2}s^2 > 0$$

$$\dot{V} = s\dot{s} < 0$$

And they claim that this is inequality is satisfied when:

$$u_D = k_D sign(s)$$

And so the complete control input is then:

$$U_1 = [g + \lambda\dot{e} + \ddot{z}_d\,]\ \frac{m}{cos(\phi)\ cos(\overset{\theta}{\cancel{\psi}})} + k_D \frac{s}{|s| + \delta}$$

The Paper: Controller Derivation

Similarly for roll, pitch, and yaw:

$$u_{eq\phi} = [\lambda(\dot{\phi}_d - \dot{\phi}) + \ddot{\phi}_d - \dot{\theta}\dot{\psi}\left(\frac{I_y - I_z}{I_x}\right) + \frac{J_r}{I_x}\dot{\theta}.\Omega]\frac{I_x}{l}$$

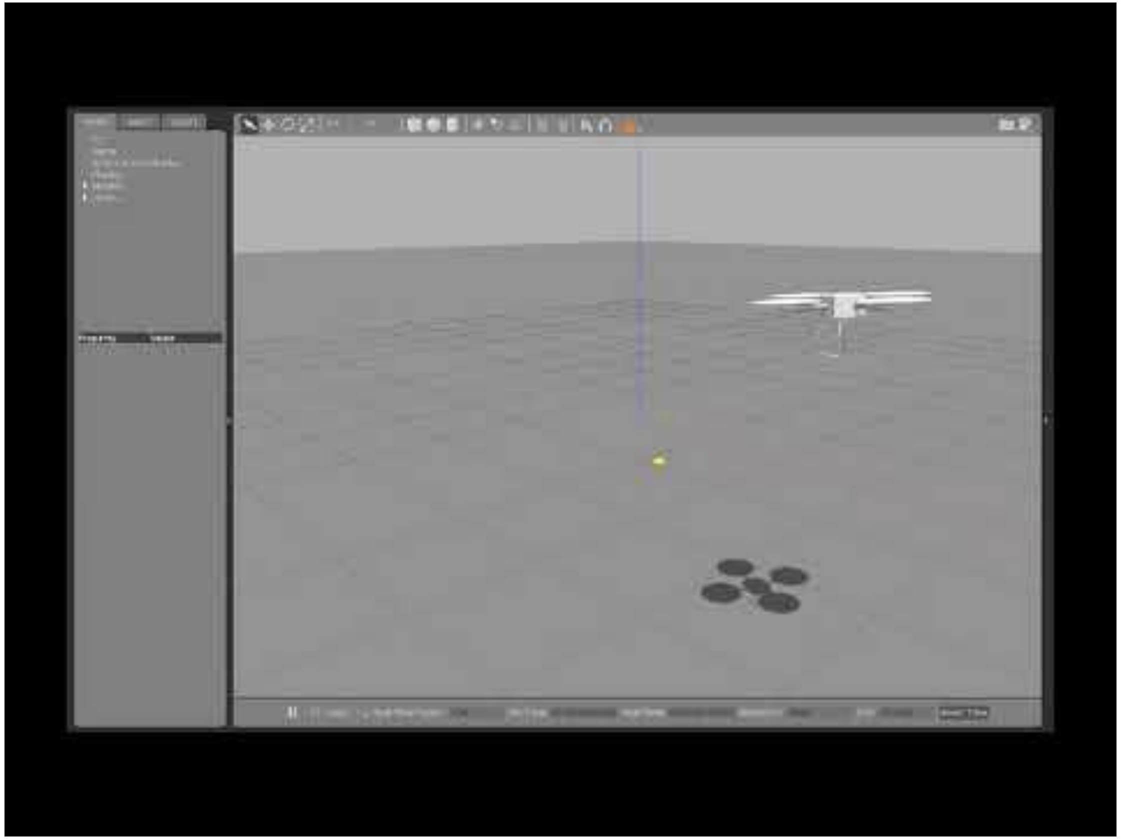
$$u_{eq\theta} = [\lambda(\dot{\theta}_d - \dot{\theta}) + \ddot{\theta}_d - \dot{\phi}\dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) - \frac{J_r}{I_y}\dot{\theta}.\Omega]\frac{I_y}{l}$$

$$u_{eq\psi} = [\lambda(\dot{\psi}_d - \dot{\psi}) + \ddot{\psi}_d - \dot{\theta}\dot{\phi}\left(\frac{I_x - I_y}{I_z}\right)]I_z$$

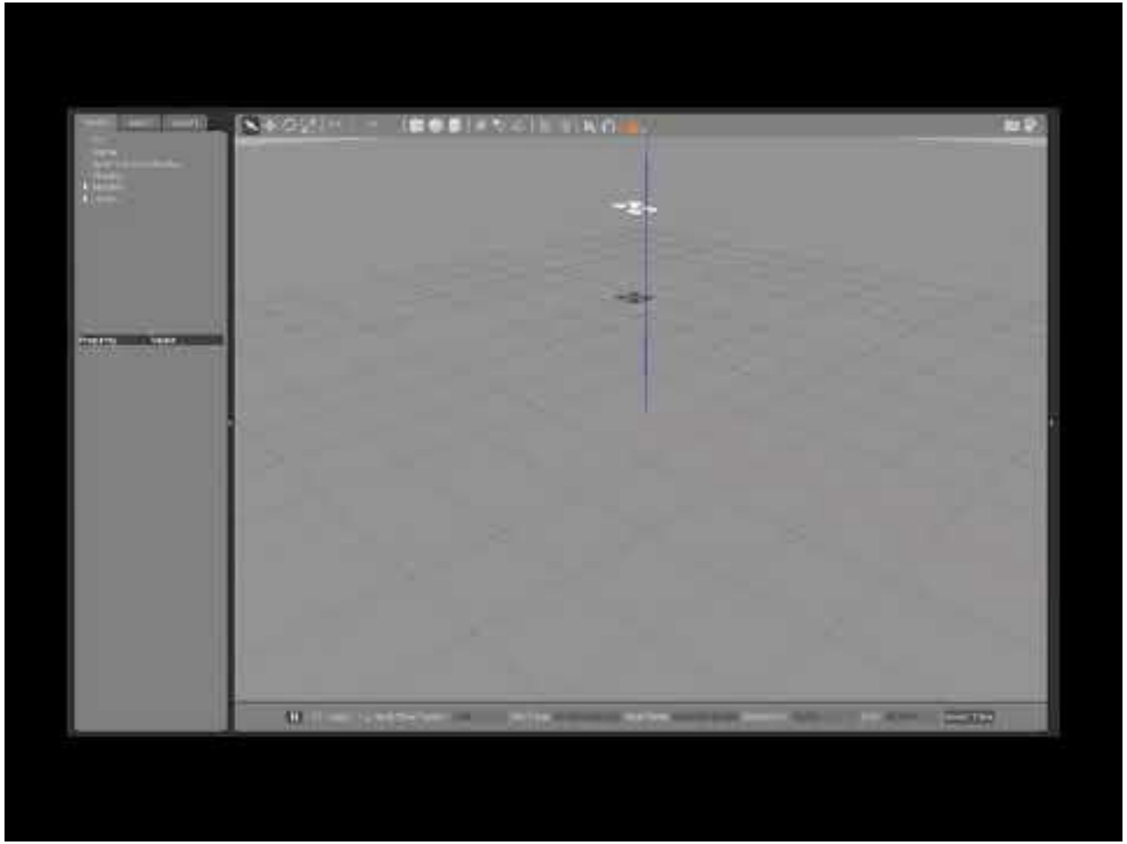
Results

I implemented this paper in ROS and Gazebo based off of the MAGICC Lab’s ROScopter simulation

Results

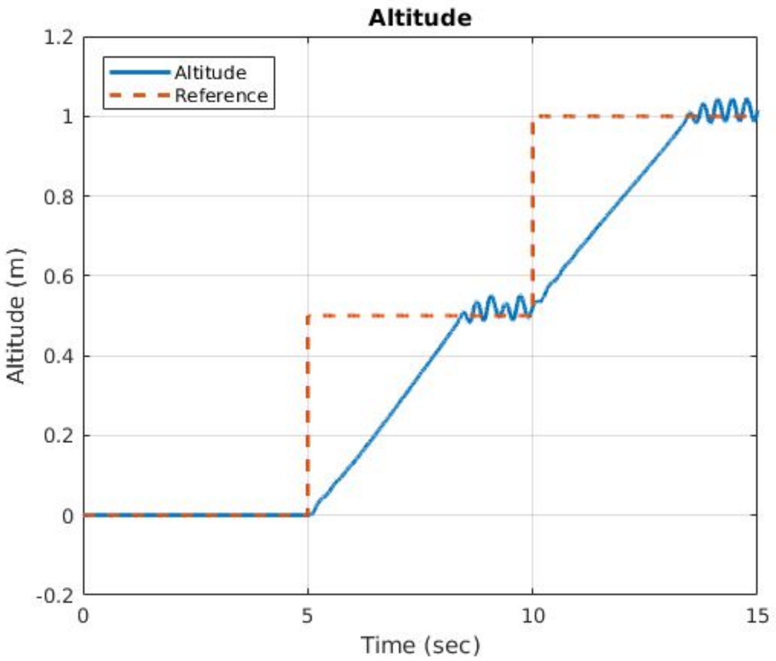


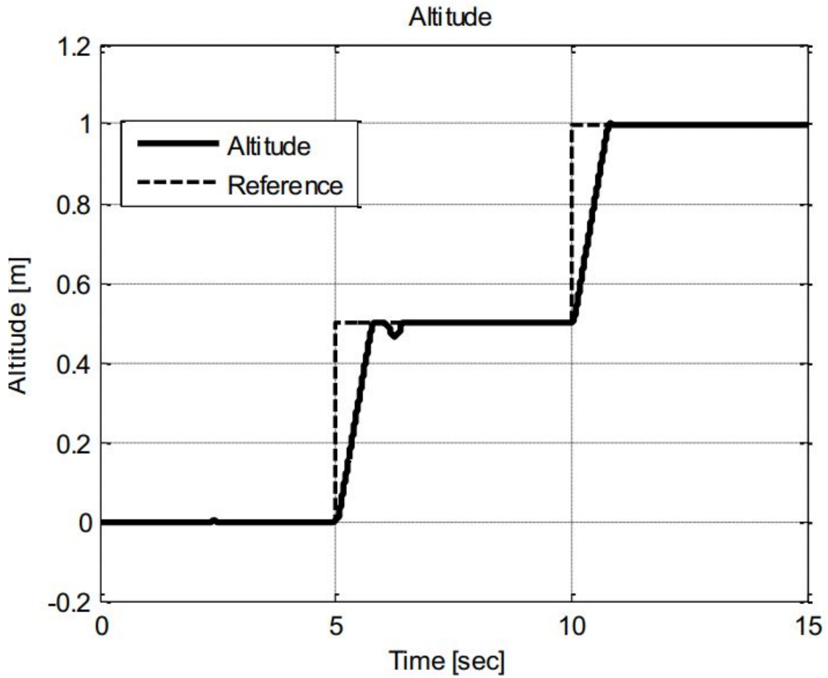
Results (after some tuning)



Altitude Control

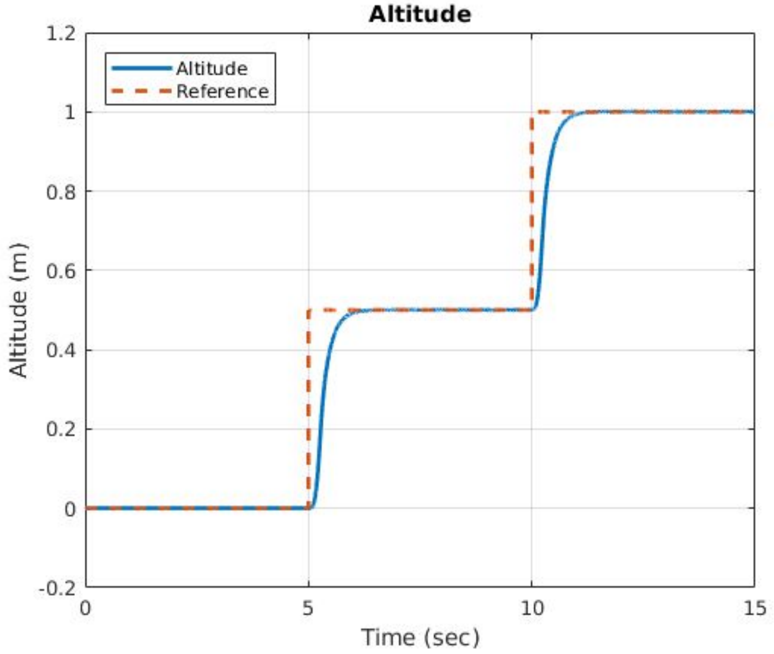
Using the paper’s gains:

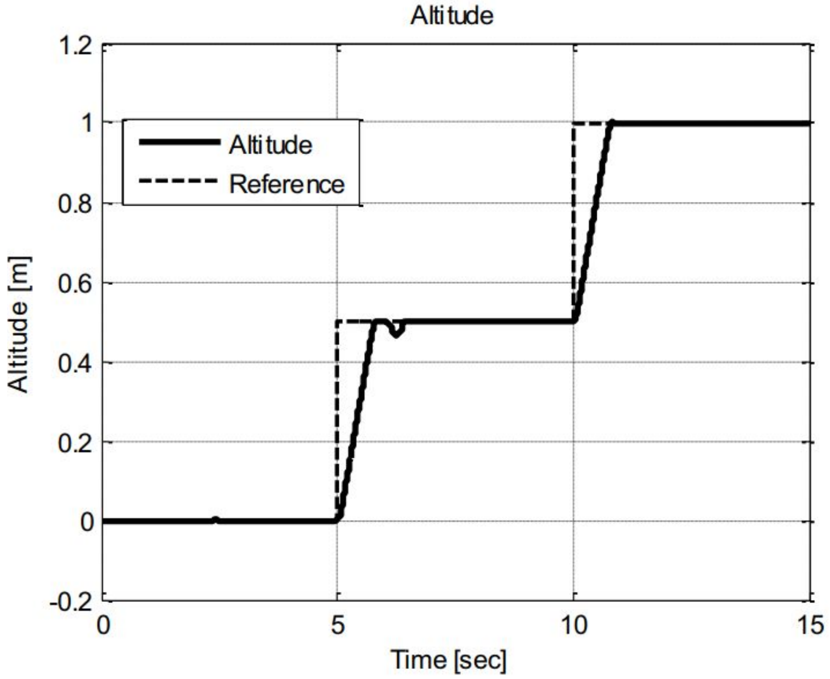




Altitude Control

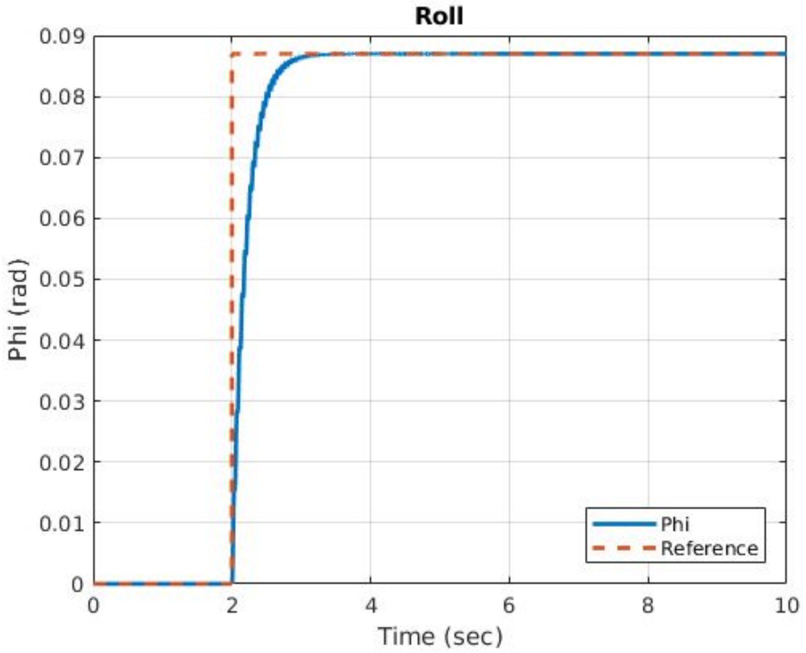
After some tuning… :

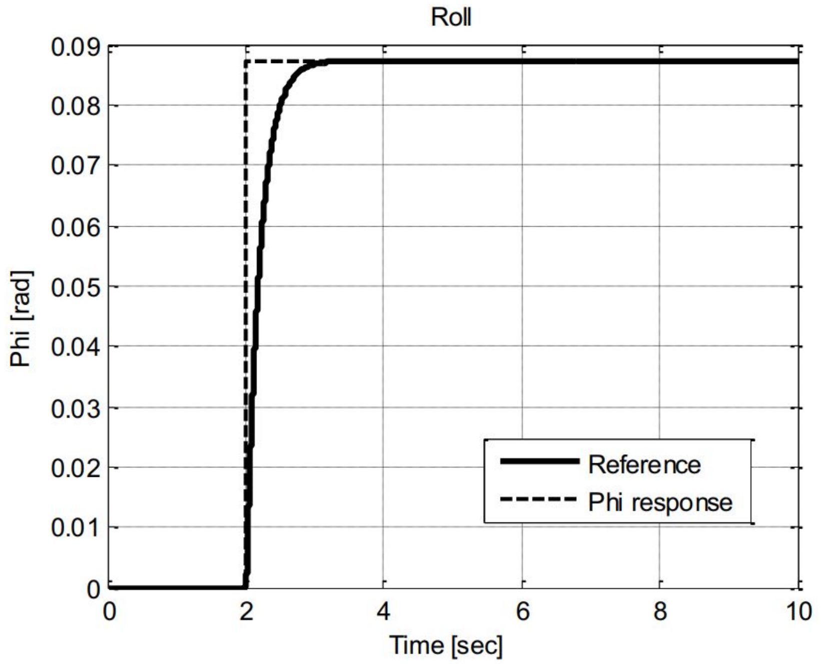




Roll Control

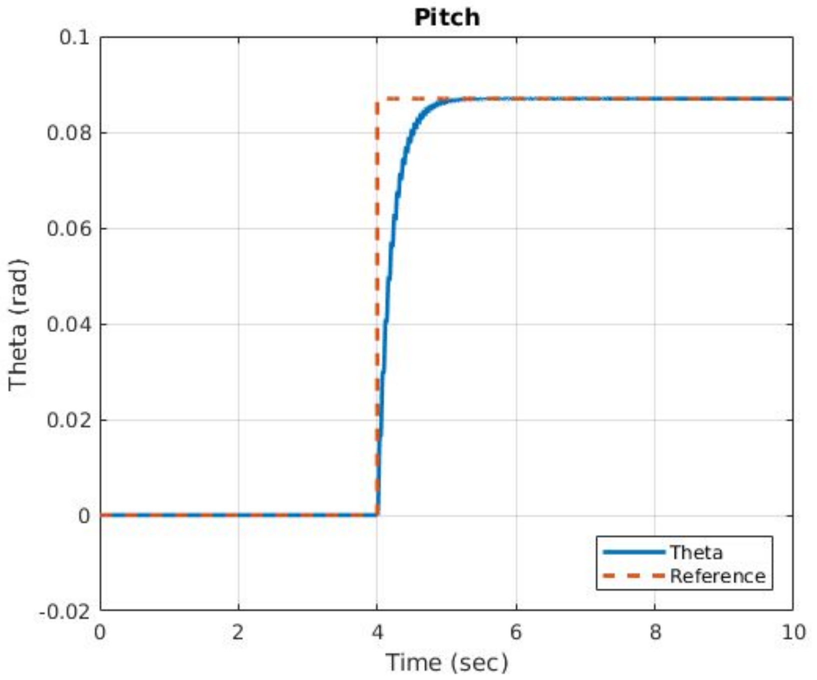
Using the paper's gains:

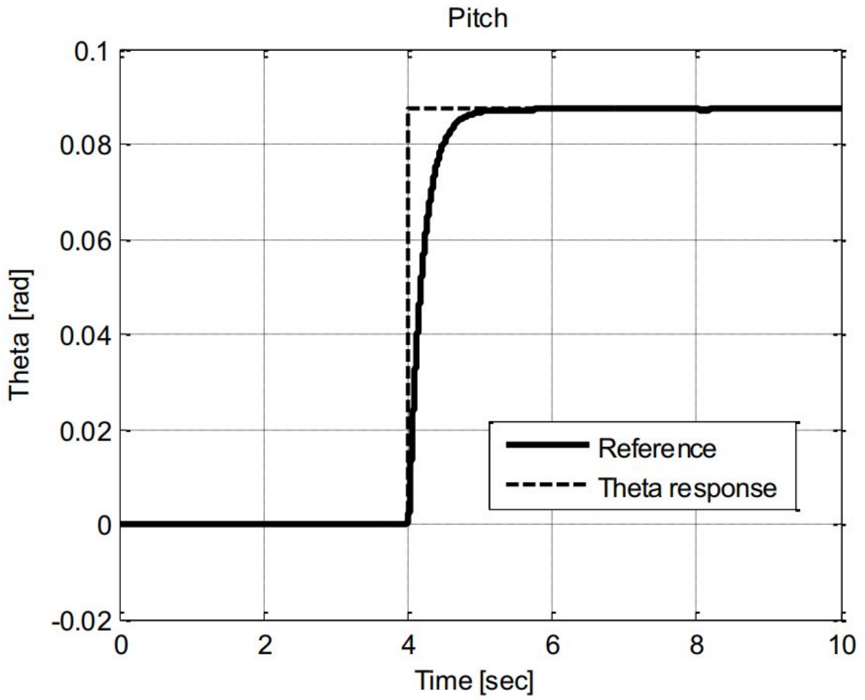




Pitch Control

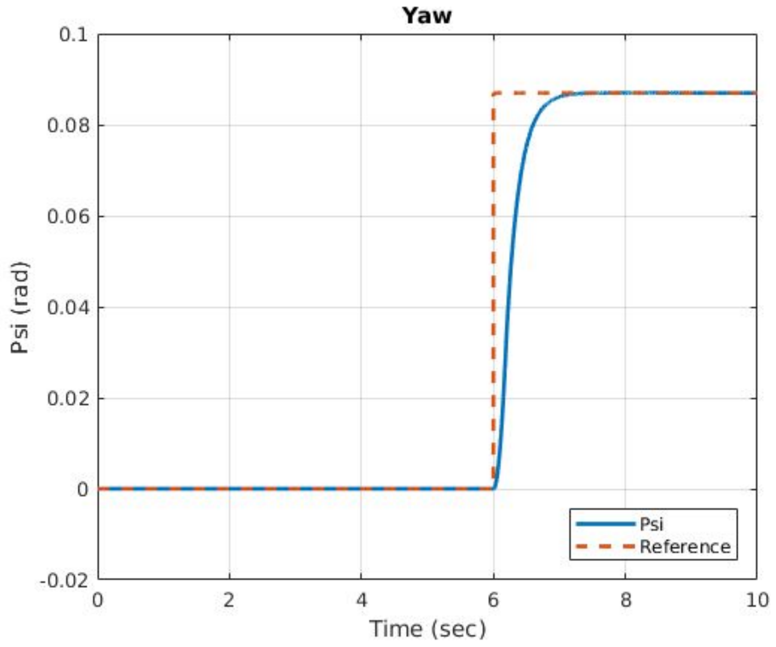
Using the paper's gains:

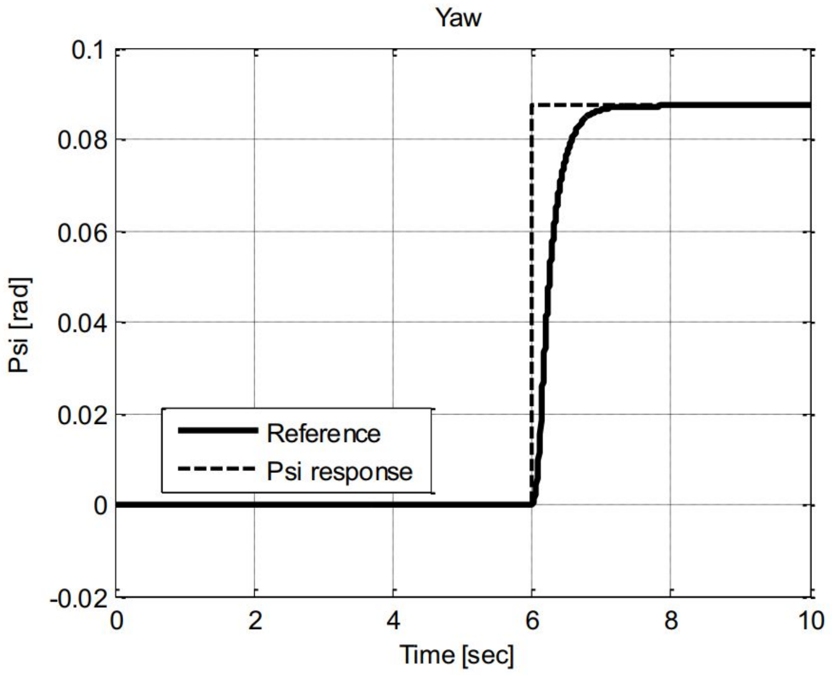




Yaw Control

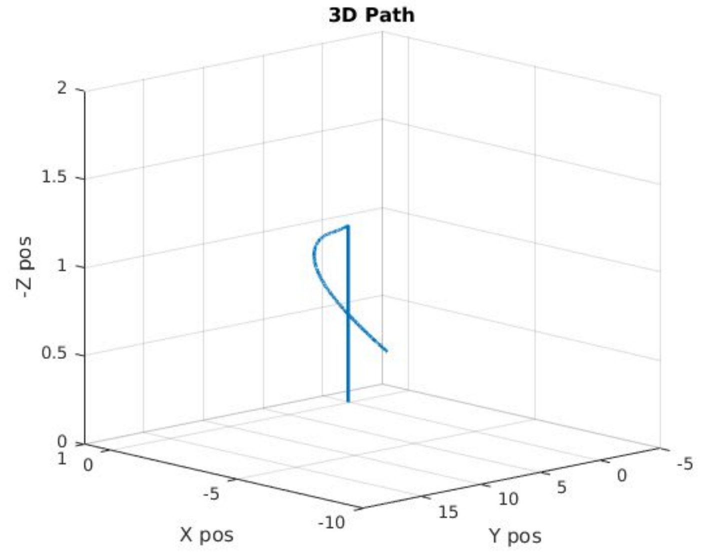
Using the paper’s gains:

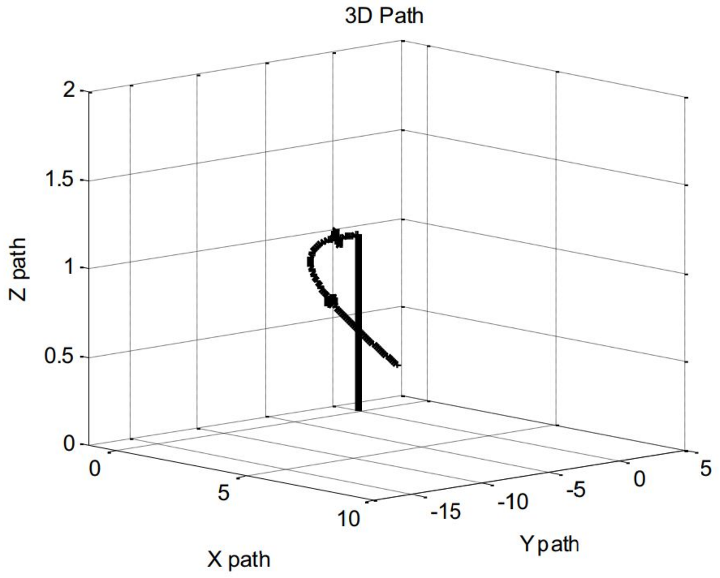




Trajectory

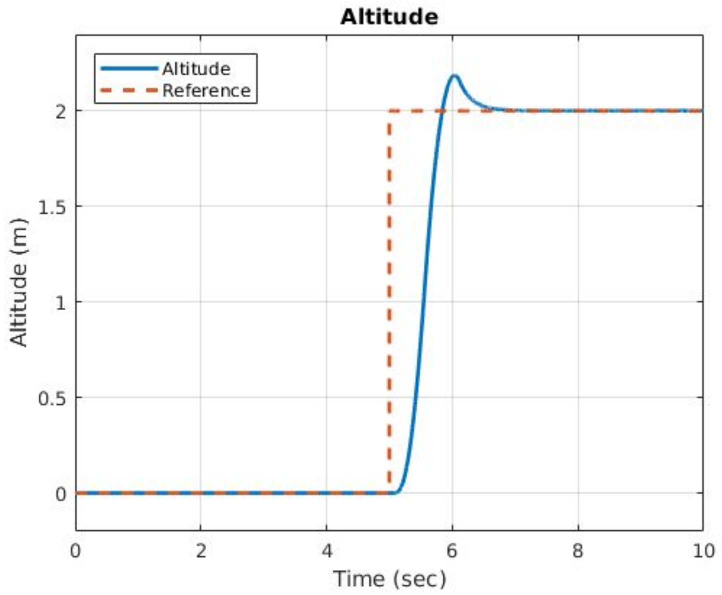
Performing these commands in sequence...





Robustness Check

To test the robustness claims of SMC I doubled gravity and tested the altitude response:



Overshoot but still stabilizes very nicely

Conclusions

Sliding mode control seems like it actually can be a good fit for the quadrotor case.

- Implementation in code is fairly straightforward
- I still feel like I need to understand more about selection of the sliding mode surface
- Overall it was a good experience and an enjoyable project

Questions?

Video Links

First Demo: https://youtu.be/BPn8gN1QOzE

Second Demo: https://youtu.be/d-zb6jNipHY

Results 1 (chattery): https://youtu.be/nT4dZZIfBW0

Results 2 (tuned): https://youtu.be/ZboHQZvNh7Q