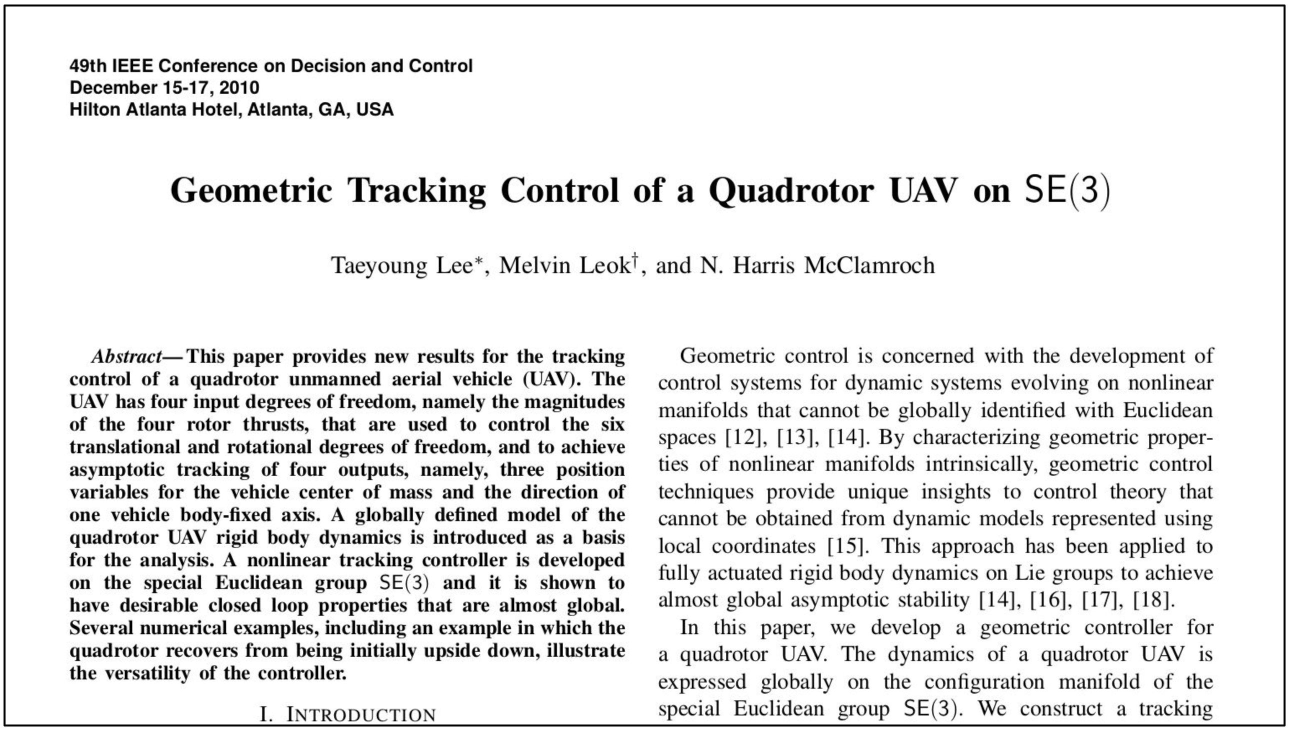


49th IEEE Conference on Decision and Control
December 15-17, 2010
Hilton Atlanta Hotel, Atlanta, GA, USA

Geometric Tracking Control of a Quadrotor UAV on SE(3)

Taeyoung Lee*, Melvin Leok†, and N. Harris McClamroch

***Abstract*— This paper provides new results for the tracking control of a quadrotor unmanned aerial vehicle (UAV). The UAV has four input degrees of freedom, namely the magnitudes of the four rotor thrusts, that are used to control the six translational and rotational degrees of freedom, and to achieve asymptotic tracking of four outputs, namely, three position variables for the vehicle center of mass and the direction of one vehicle body-fixed axis. A globally defined model of the quadrotor UAV rigid body dynamics is introduced as a basis for the analysis. A nonlinear tracking controller is developed on the special Euclidean group SE(3) and it is shown to have desirable closed loop properties that are almost global. Several numerical examples, including an example in which the quadrotor recovers from being initially upside down, illustrate the versatility of the controller.**

I. Introduction

Geometric control is concerned with the development of control systems for dynamic systems evolving on nonlinear manifolds that cannot be globally identified with Euclidean spaces [12], [13], [14]. By characterizing geometric properties of nonlinear manifolds intrinsically, geometric control techniques provide unique insights to control theory that cannot be obtained from dynamic models represented using local coordinates [15]. This approach has been applied to fully actuated rigid body dynamics on Lie groups to achieve almost global asymptotic stability [14], [16], [17], [18].

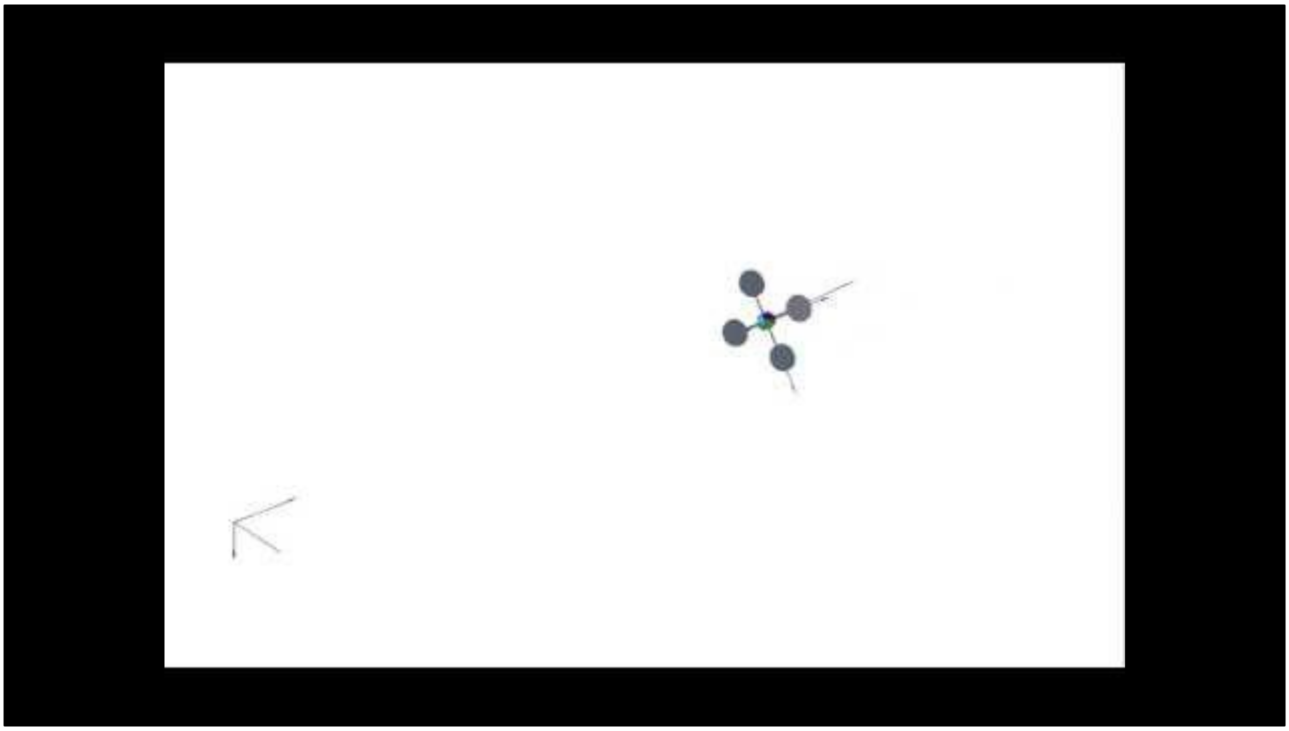
In this paper, we develop a geometric controller for a quadrotor UAV. The dynamics of a quadrotor UAV is expressed globally on the configuration manifold of the special Euclidean group SE(3). We construct a tracking

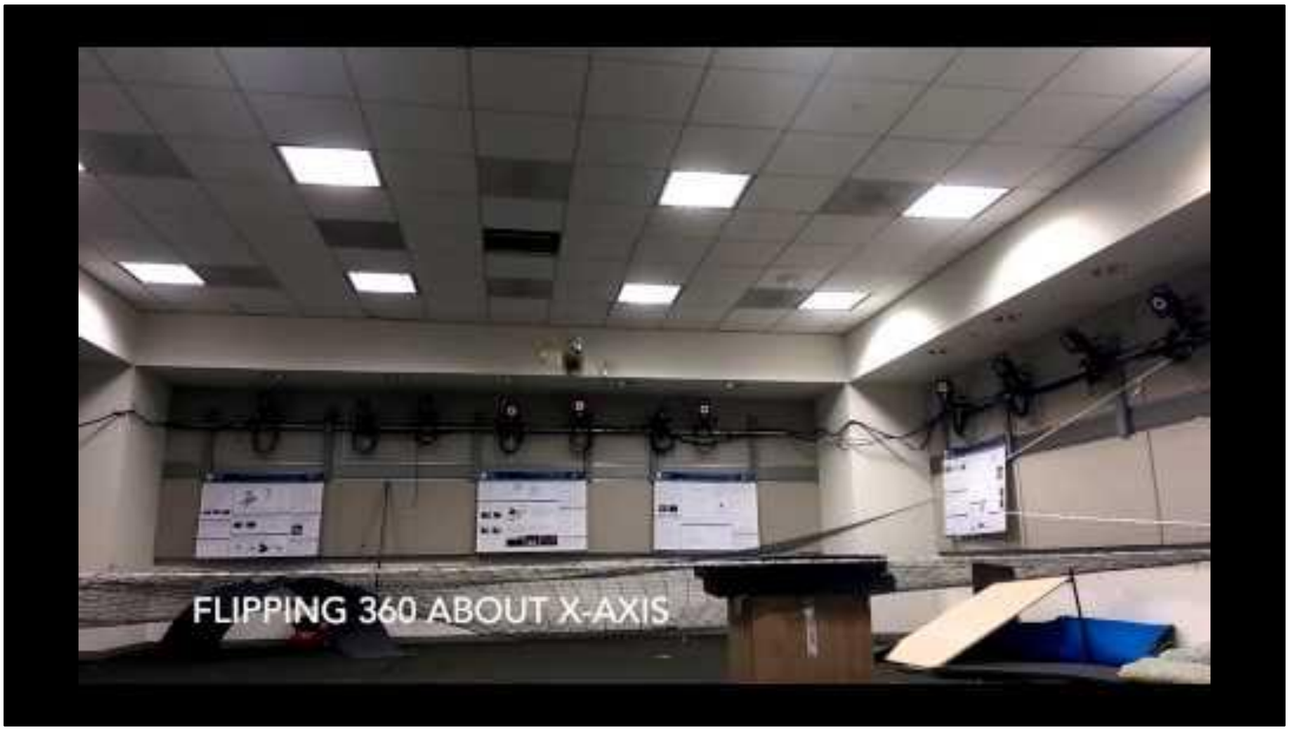
Problems with parameterizations: https://en.wikipedia.org/wiki/Charts_on_SO(3)

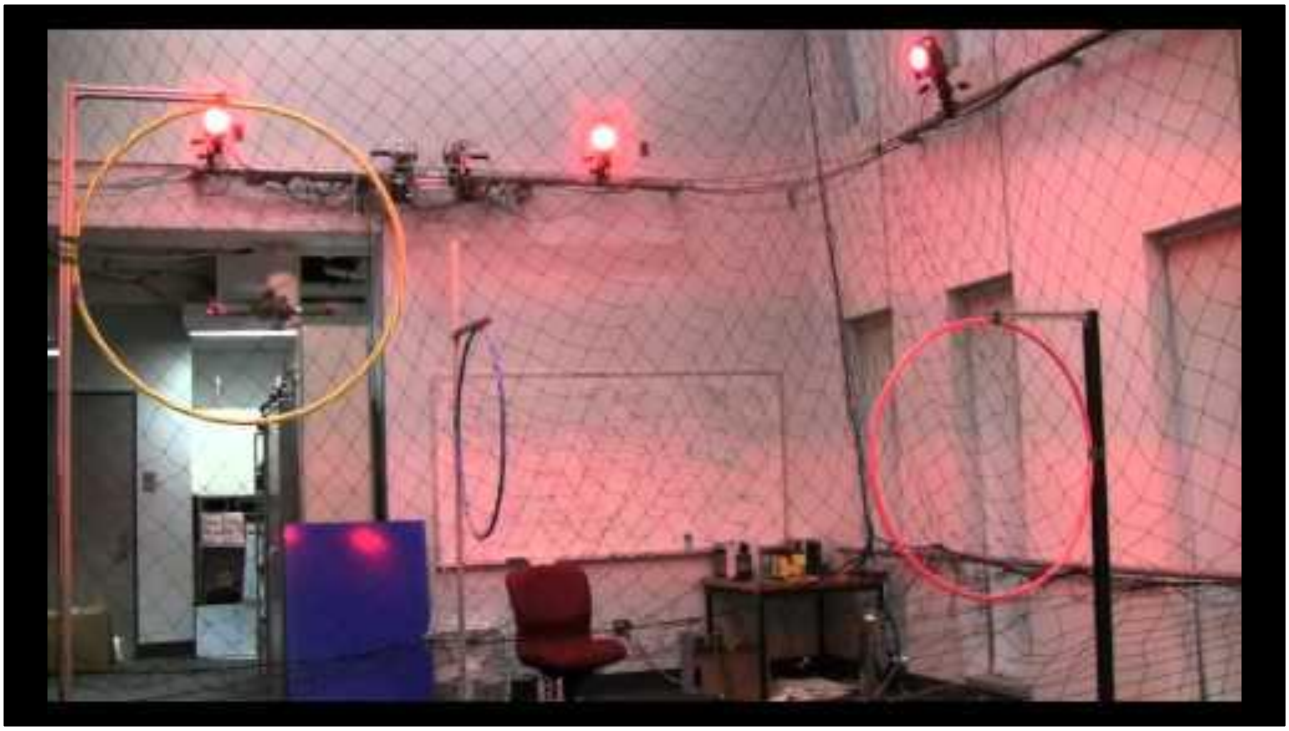
http://lucaballan.altervista.org/pdfs/RigidTransformations.pdf

“Geometric control is concerned with the development of control systems for dynamical systems evolving on nonlinear manifolds that cannot be globally identified with Euclidean spaces.”

Lee et al., *Control of Complex Maneuvers for a Quadrotor UAV using Geometric Methods on SE(3)*, 2010.




FLIPPING 360 ABOUT X-AXIS



Interesting perspective of differential flatness that I didn't realize -- it seems to be directly related to manifolds and the idea of finding a mapping from the curvey SE(3) space onto a flat space: https://youtu.be/cx1WugXzlFM?t=10m51s

Definitions of SO(3) and SE(3)

$$\mathsf{SO}(3) \triangleq \{R \in \mathbb{R}^{3\times 3} \mid R^\top R = I,\ \mathsf{det}\ R = 1\}$$

$$\mathsf{SE}(3) \triangleq \{(R, \mathbf{p}) \mid R \in \mathsf{SO}(3), \mathbf{p} \in \mathbb{R}^3\}$$

Note: These are also Lie Groups because they are groups that are also differentiable manifolds. This allows us to use some nice results from differential geometry and Lie theory (e.g., tangent spaces and the exponential mapping).

Quick Review: Manifolds

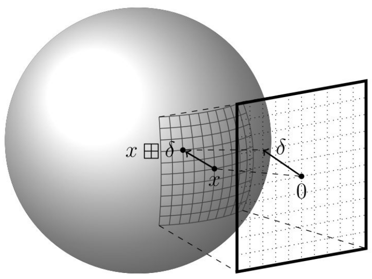


Figure 1: Mapping a local neighborhood in the state space (here: on the unit sphere S^2) into $\mathbb{R}^n$ (here: the plane) allows for the use of standard sensor fusion algorithms without explicitly encoding the global topological structure.

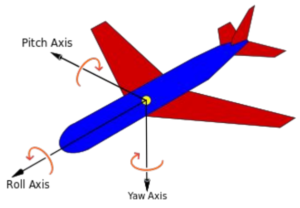
Hertzberg et al., *Integrating Generic Sensor Fusion Algorithms with Sound State Representations Through Encapsulation of Manifolds*, 2013.

IF THE EARTH WAS FLAT
CATS WOULD HAVE PUSHED EVERYTHING OFF IT BY NOW

Other Popular Representations of Orientation

Euler Angles

- Attempt to globally cover SO(3)
- Have to deal with wrapping and singularities
- Intuitive to understand



Quaternions

- Hypercomplex representation of rotation
- Double covers SO(3):
  - **q** and -**q** represent the same orientation
- Computationally efficient

$$S^3 = \{\mathbf{q} \in \mathbb{H} \mid ||\mathbf{q}|| = 1\}$$

$$\mathbf{q} = q_0 + q_x i + q_y j + q_z k$$

The equations of motion of the quadrotor UAV can be written as

$$\dot{x} = v, \tag{2}$$

$$m\dot{v} = mge_3 - fRe_3, \tag{3}$$

$$\dot{R} = R\hat{\Omega}, \tag{4}$$

$$J\dot{\Omega} + \Omega \times J\Omega = M, \tag{5}$$

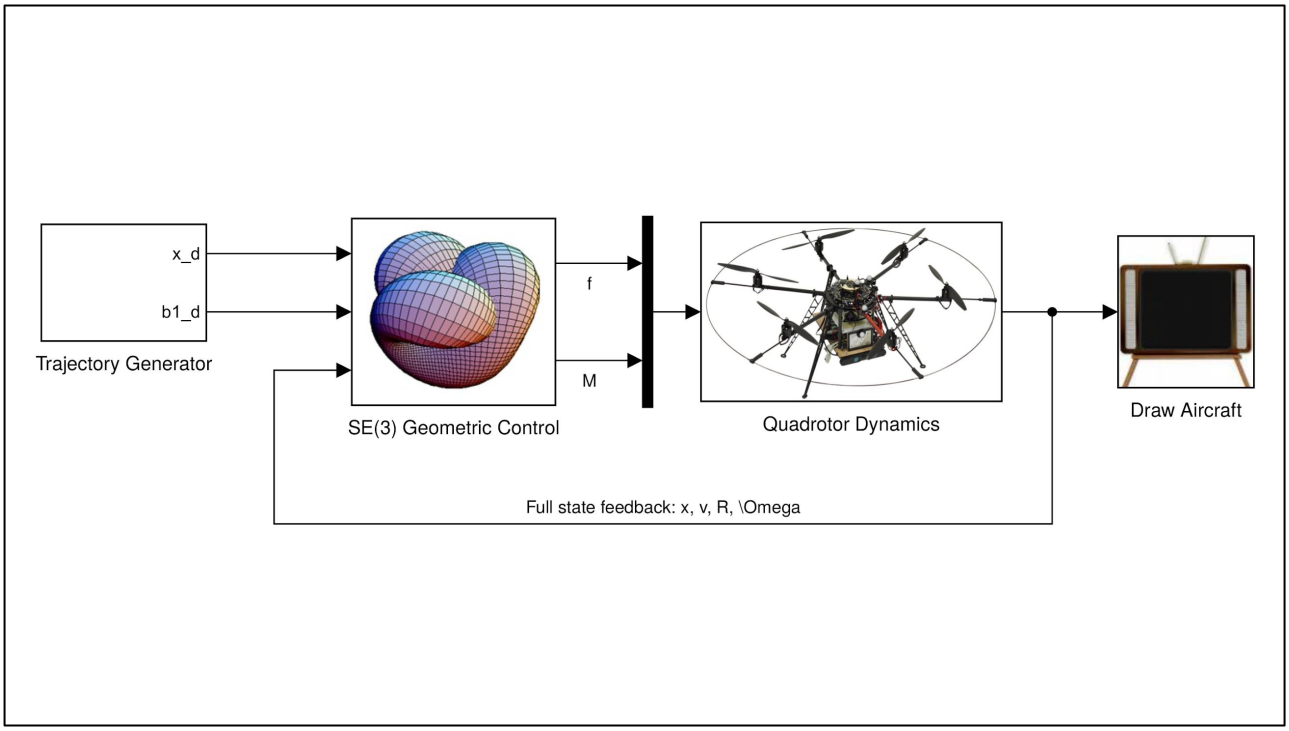
where the *hat map* $\hat{\cdot} : \mathbb{R}^3 \to \mathfrak{so}(3)$ is defined by the condition that $\hat{x}y = x \times y$ for all $x, y \in \mathbb{R}^3$ (see Appendix A).

A. Properties of the Hat Map

The hat map $\hat{\cdot} : \mathbb{R}^3 \to \mathfrak{so}(3)$ is defined as

$$\hat{x} = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix} \tag{55}$$

for $x = [x_1; x_2; x_3] \in \mathbb{R}^3$. This identifies the Lie algebra $\mathfrak{so}(3)$ with $\mathbb{R}^3$ using the vector cross product in $\mathbb{R}^3$. The inverse of



Maybe do a slide on group theory

http://www.roboticsproceedings.org/rss11/p06.pdf

Rotation:
https://cwzx.wordpress.com/2013/12/16/numerical-integration-for-rotational-dynamics/
http://www.cs.unc.edu/~lin/COMP768-F07/LEC/rbd1.pdf

Python quadrotor simulation: https://github.com/hbd730/quadcopter-simulation

30,000 Foot View

- We would like to control position and heading.
- Our Lyapunov-based control recipe will be:
  - Analyze the attitude dynamics and design a tracking controller for an input of desired attitude.
  - Design a tracking controller for an input of desired position.
  - Extend the position controller to allow a specification of heading
  - Make stability claims along the way

Attitude Tracking Controller

- An arbitrary, smooth attitude command $R_d(t) \in \mathsf{SO(3)}$ is given.
- Define an attitude error function that evolves on the tangent bundle of $\mathsf{SO(3)}$: $\Psi(R, R_d) = \frac{1}{2}\mathrm{tr}[I - R_d^T R]$.
- Using Lyapunov-based control design, it is shown that the attitude controller given below achieves **exponential stability** to the desired attitude -- *assuming attitude errors < 180°.*

$$M = -k_R e_R - k_\Omega e_\Omega + \Omega \times J\Omega - J(\hat{\Omega} R^T \underline{R_d \Omega_d - R^T R_d \dot{\Omega}_d})$$

Position Tracking Controller

- An arbitrary, smooth position command $x_d(t) \in \mathbb{R}^3$ is given.
- The below position and attitude control can be shown to give **exponential stability** to the desired position command -- *assuming attitude errors < 90°*.
- However, it can be shown that this controller achieves **almost global exponential attractiveness** for *attitude errors < 180°*.

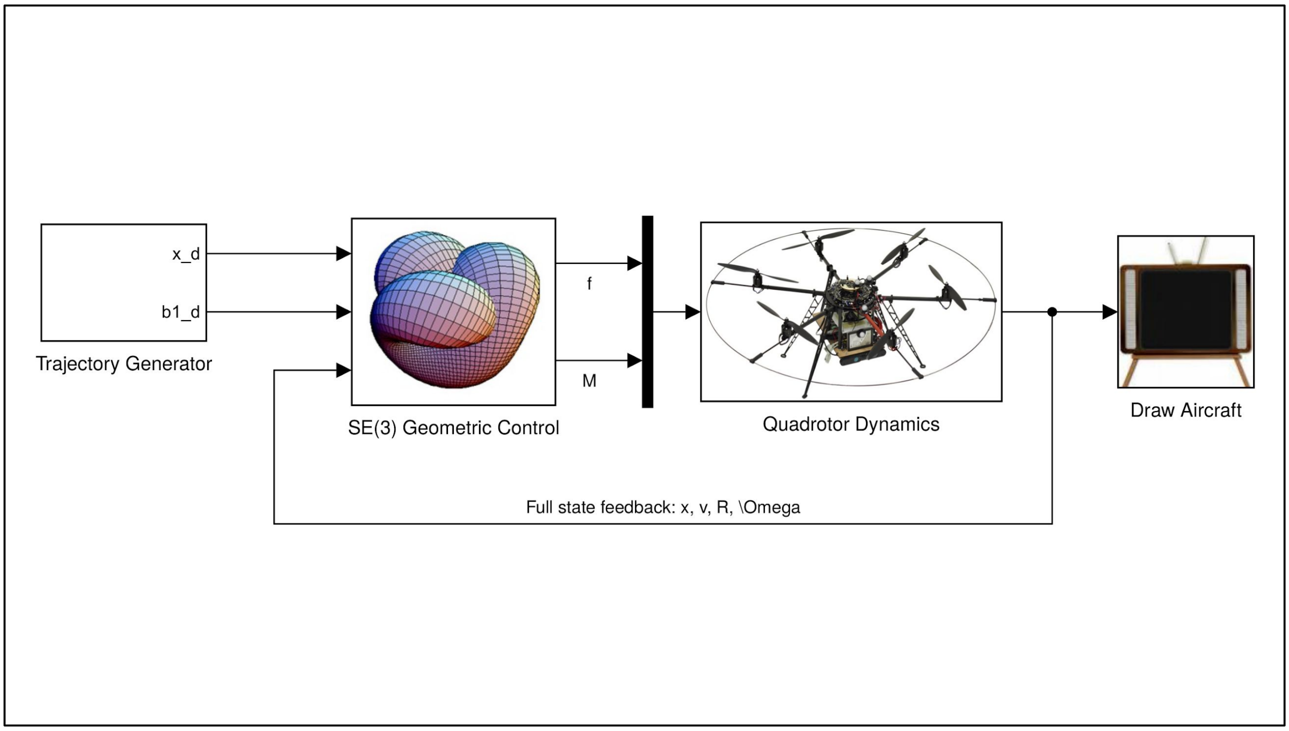
$$f = (k_x e_x + k_v e_v + mge_3 - m\ddot{x}_d) \cdot Re_3$$
$$M = -k_R e_R - k_\Omega e_\Omega + \Omega \times J\Omega - J(\hat{\Omega} R^T \underline{R_c \Omega_c - R^T R_c \dot{\Omega}_c})$$

Position Tracking Controller with Desired Heading

- An arbitrary, smooth position command $x_d(t) \in \mathbb{R}^3$ and a desired heading direction $b_{1_d} \in \mathsf{S}^2$ are given.
- The given controller continues to achieve **almost global exponential attractiveness** for *attitude errors < 180°.*
- The controller gives **asymptotic stability** in heading direction.

$$f = (k_x e_x + k_v e_v + mge_3 - m\ddot{x}_d) \cdot Re_3$$

$$M = -k_R e_R - k_\Omega e_\Omega + \Omega \times J\Omega - J(\hat{\Omega} R^T \underline{R_c \Omega_c - R^T R_c \dot{\Omega}_c})$$



Maybe do a slide on group theory

http://www.roboticsproceedings.org/rss11/p06.pdf

Rotation:
https://cwzx.wordpress.com/2013/12/16/numerical-integration-for-rotational-dynamics/
http://www.cs.unc.edu/~lin/COMP768-F07/LEC/rbd1.pdf

Python quadrotor simulation: https://github.com/hbd730/quadcopter-simulation

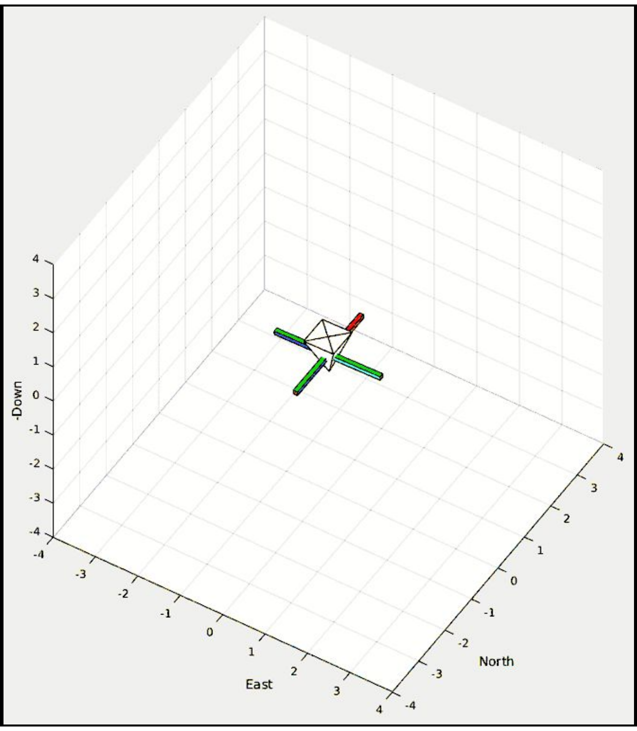
Initially Upside Down Quadrotor

$$x_d(t) = [0, 0, 0], \quad b_{1_d}(t) = [1, 0, 0]$$

$$x(0) = [0, 0, 0], \quad v(0) = [0, 0, 0],$$

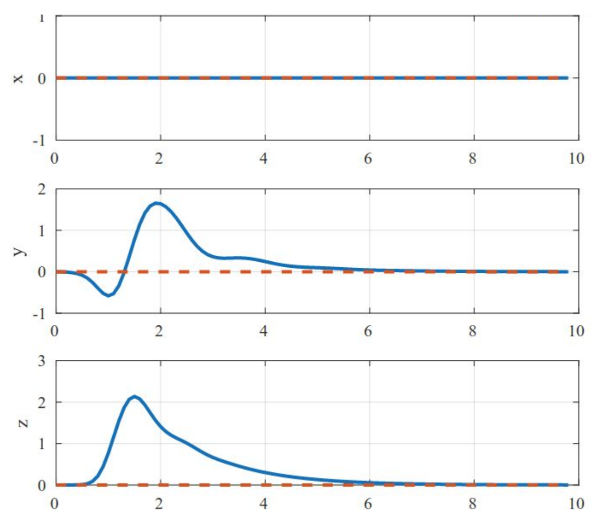
$$R(0) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -0.9995 & -0.0314 \\ 0 & 0.0314 & -0.9995 \end{bmatrix}, \quad \Omega(0) = [0, 0, 0].$$

(178.2° initial attitude error)

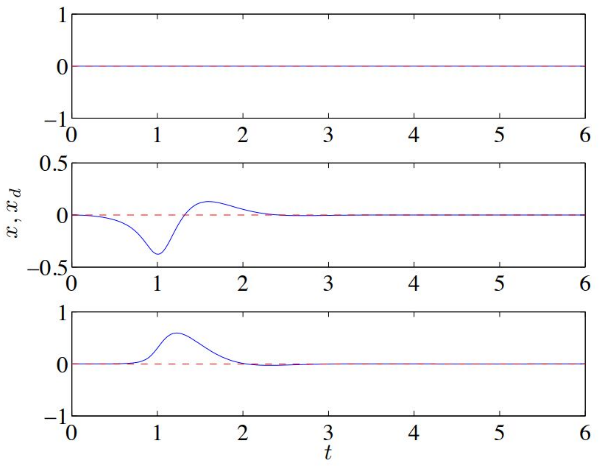

-Down
East
North
4
3
2
1
0
-1
-2
-3
-4

Initially Upside Down Quadrotor

My Implementation



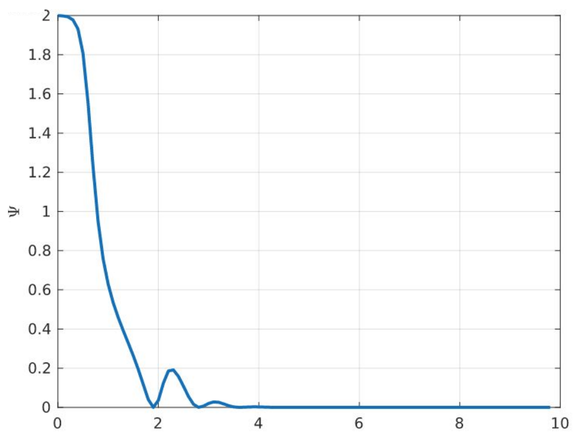
Results from Lee et al.



(b) Position (x:solid, x_d:dotted, (m))

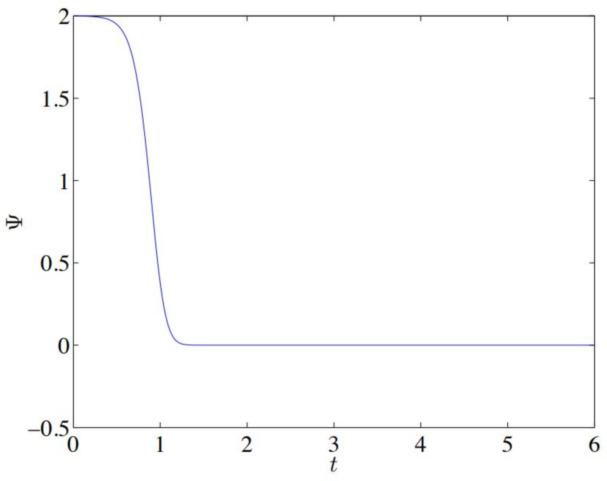
Initially Upside Down Quadrotor

My Implementation



$$\Psi(R, R_d) = \frac{1}{2}\mathrm{tr}[I - R_d^T R]$$

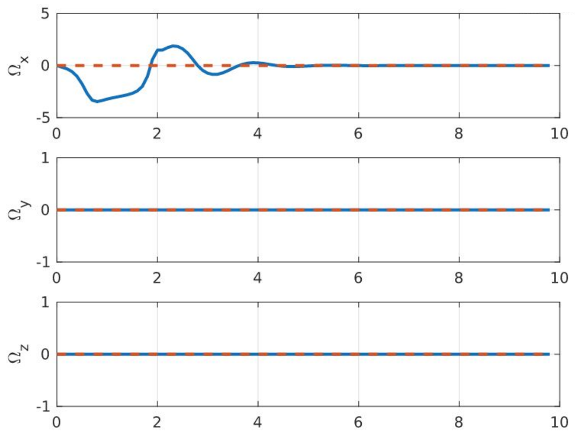
Results from Lee et al.



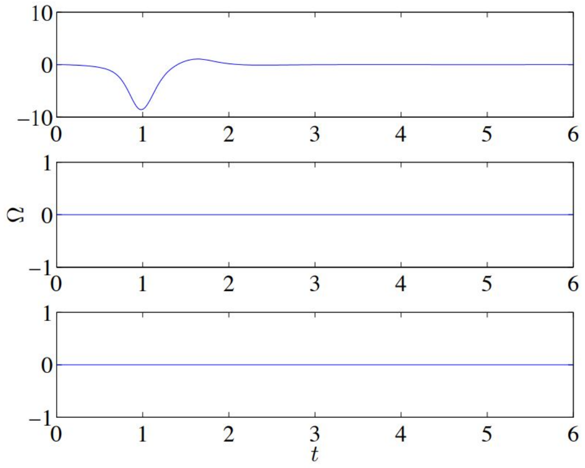
(a) Attitude error function Ψ

Initially Upside Down Quadrotor

My Implementation



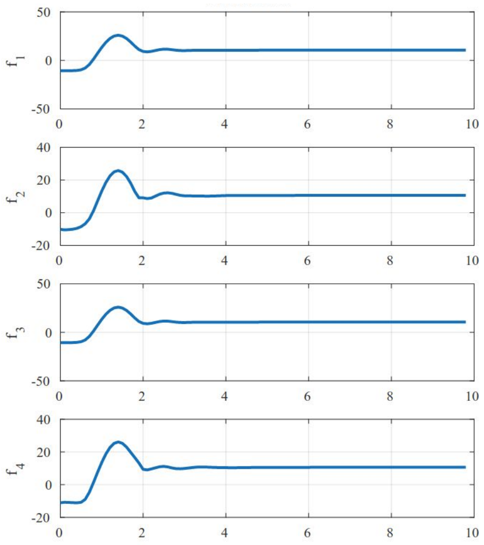
Results from Lee et al.



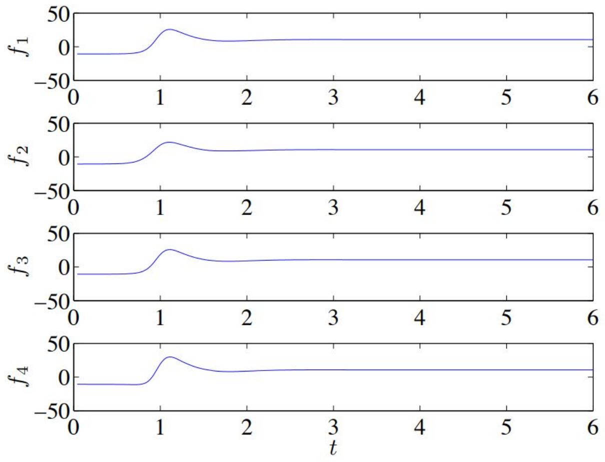
(c) Angular velocity (rad/sec)

Initially Upside Down Quadrotor

My Implementation



Results from Lee et al.



(d) Thrust of each rotor (N)

Implementation Thoughts

- Lots of high-order numerical derivatives
- Nice not having to worry about Euler angles
    - Wrapping, singularities
- Lots of operations (cross, dot, norm), but fairly easy to implement
- Math is different, less intuitive to debug