UCLA Samueli School of Engineering

SQuAT Plan

Smooth QUadrotor Agile Trajectory PLANning

Aaron John Sabu, Ryan Nemiroff

Class: **Control and Trajectory Planning for Autonomous Aerial Systems**

Instructor: Dr. Brett Lopez

2022 December 1

Motivation

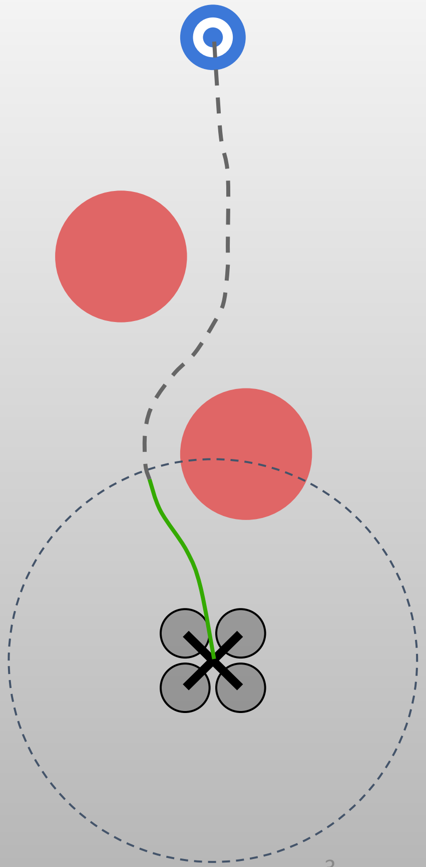
- **Cluttered environments**
  - **Drone delivery in a city**
- **Warehouse Inventory**

https://www.corvus-robotics.com/corvus-one

https://www.aboutamazon.com/news/transportation/amazon-prime-air-prepares-for-drone-deliveries

Problem Statement

- Reach a goal position
  - without *a priori* information about the environment,
  - with a limited sensing horizon,
  - while avoiding obstacles,
  - and remaining in safe input and state ranges.
- Assumptions:
  - Perfect state estimates are available
  - Obstacles can be represented as ellipsoids and cylinders
  - Obstacle perception is given
- Deliverables:
  - Trajectory planner (MPC)
  - Tracking controller



Trajectory Optimizer

- Use Python library called **GEKKO** [1]
  - GEKKO is a front-end for **APMonitor**
  - Nonlinear optimization
  - Built-in methods for solving optimal control problems with dynamics
    - Breaks problem into discrete time steps
    - Discretized dynamics ≠ continuous dynamics :(

[1] Beal, L., Hill, D., Martin, R., & Hedengren, J. (2018). GEKKO Optimization Suite. *Processes, 6(8).*

Problem Formulation 1: Chain of Integrators

$$\min\left(K_p||\mathbf{p}-\mathbf{p}_f||^2+\left(K_s||\mathbf{s}||^2\right)\right)$$

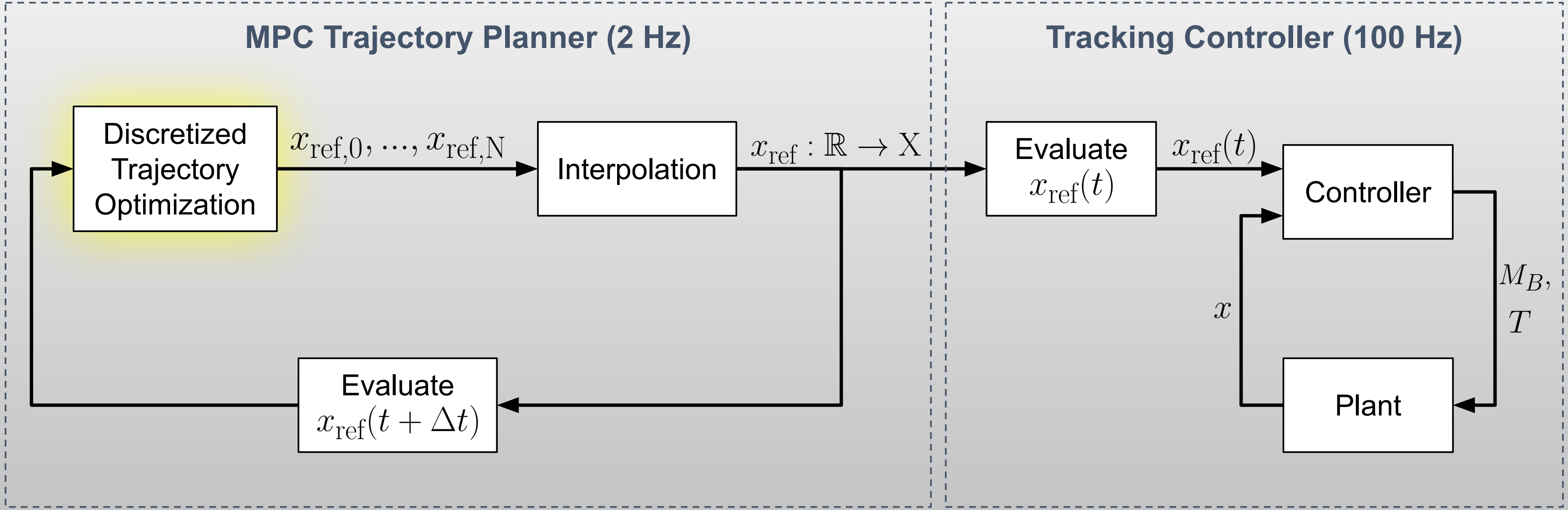
- Dynamics $\dot{\mathbf{p}}=\mathbf{v}$ $\dot{\mathbf{v}}=\mathbf{a}$ $\dot{\mathbf{a}}=\mathbf{j}$ $\dot{\mathbf{j}}=\mathbf{s}$
- Initial States $\mathbf{p}(t=0)=\mathbf{p}_0$ $\mathbf{v}(t=0)=\mathbf{v}_0$ $\mathbf{a}(t=0)=\mathbf{a}_0$ $\mathbf{j}(t=0)=\mathbf{j}_0$
- Terminal States $\mathbf{v}(t=t_f)=\mathbf{v}_f$ $\mathbf{a}(t=t_f)=\mathbf{a}_f$ $\mathbf{j}(t=t_f)=\mathbf{j}_f$
- Input Constraints $\mathbf{s}\in\mathbb{S}$

Problem Formulation 2: Quadrotor Dynamics

$$\min \left(K_p ||\mathbf{p} - \mathbf{p}_f||^2 + \left(K_f ||\mathbf{f}_B||^2 + K_M ||\mathbf{M}_B||^2 \right) \right)$$

- Dynamics $\dot{\mathbf{p}} = \mathbf{v}$ $\quad M\dot{\mathbf{v}} = \mathbf{f}_I + q \otimes \mathbf{f}_B \otimes q$ $\quad \dot{q} = \frac{1}{2} q \otimes \begin{pmatrix} 0 \\ \omega \end{pmatrix}$
  $\mathbf{f}_I = \mathbf{g}$ $\quad J\dot{\omega} = \mathbf{M}_B - \omega \times J\omega$
- Initial States $\mathbf{p}(t=0) = \mathbf{p}_0$ $\quad q(t=0) = q_0$
  $\mathbf{v}(t=0) = \mathbf{v}_0$ $\quad \omega(t=0) = \omega_0$
- Terminal States $\mathbf{v}(t=t_f) = \mathbf{v}_f$ $\quad q(t=t_f) = q_f$
  $\omega(t=t_f) = \omega_f$
- Input Constraints $\mathbf{f}_B \in \mathbb{F}$ $\quad \mathbf{M}_B \in \mathbb{M}$

System Architecture



- Our approach to **recursive feasibility**: Vehicle must be stopped at the end of each planned trajectory.

Tracking Controller – Using Differential Flatness

If using chain of integrators optimization, convert **jerk & snap** to **angular velocity & angular acceleration** [2]:

$$\begin{bmatrix} \omega_{\text{ref}} \\ \dot{\tau}_{ref} \end{bmatrix} = \begin{bmatrix} \tau \mathbf{R}[\mathbf{i}_z]_\times^T & \mathbf{b}_z \\ \mathbf{S} & 0 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{j}_{ref} \\ \dot{\psi}_{ref} \end{bmatrix}$$

$$\begin{bmatrix} \dot{\omega}_{\text{ref}} \\ \ddot{\tau}_{ref} \end{bmatrix} = \begin{bmatrix} \tau \mathbf{R}[\mathbf{i}_z]_\times^T & \mathbf{b}_z \\ \mathbf{S} & 0 \end{bmatrix}^{-1} \left(\begin{bmatrix} \mathbf{s}_{ref} \\ \ddot{\psi}_{ref} \end{bmatrix} - \begin{bmatrix} \mathbf{R}(2\dot{\tau} + \tau[\mathbf{\Omega}]_\times)[\mathbf{i}_z]_\times^T \mathbf{\Omega} \\ \dot{\mathbf{S}}\mathbf{\Omega} \end{bmatrix} \right)$$

[2] Tal, E., & Karaman, S. (2021). Accurate Tracking of Aggressive Quadrotor Trajectories Using Incremental Nonlinear Dynamic Inversion and Differential Flatness*. IEEE Transactions on Control Systems Technology, 29(3), 1203-1218.*

Tracking Controller – PD control

1) Position control *(Lec 16)*

$$\mathbf{u} = \mathbf{a}_{\text{ref}} - \mathbf{g}_{\mathcal{I}} - K_p^{\text{pos}} \mathbf{p}_e - K_d^{\text{pos}} \dot{\mathbf{p}}_e$$

$$T = m\|\mathbf{u}\|$$

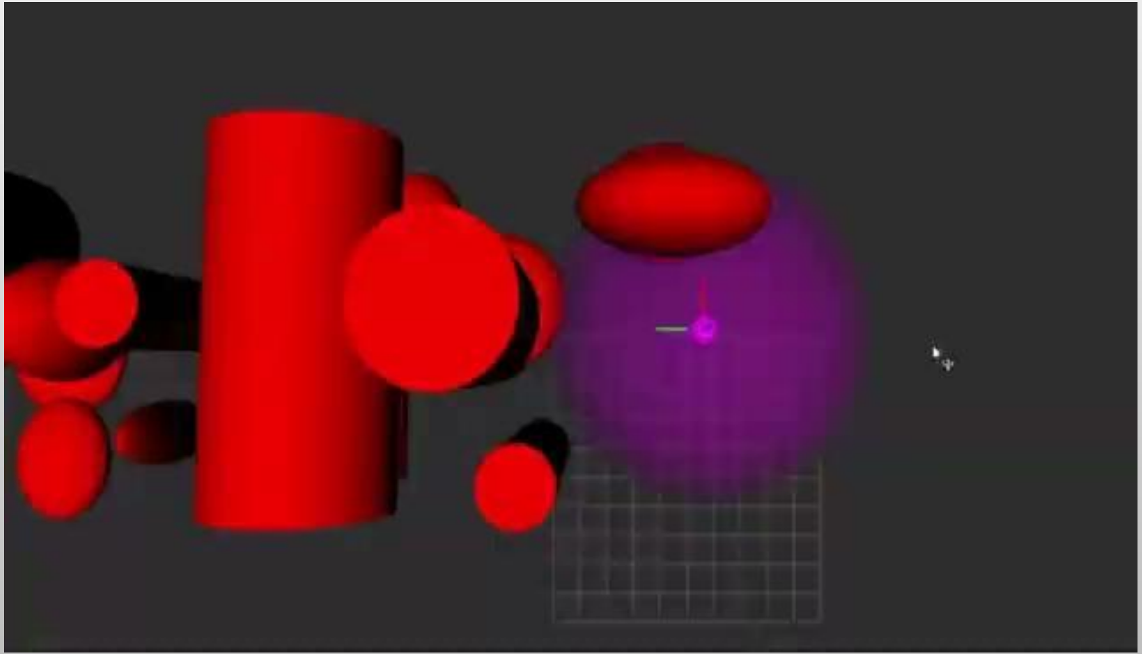
2) Compute q_d *(Lec 16)*

$$q_d = \frac{1}{\sqrt{2(1 + \hat{\mathbf{T}}^T \hat{\mathbf{u}})}} \begin{pmatrix} 1 + \hat{\mathbf{T}}^T \hat{\mathbf{u}} \\ \hat{\mathbf{T}} \times \hat{\mathbf{u}} \end{pmatrix}$$

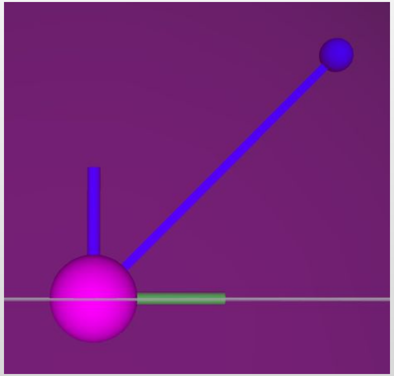
3) Attitude Control *(Lec 15)*

$$\mathbf{M}_B = J\left(\dot{\omega}_{\text{ref}} - K_p^{\text{att}} \operatorname{sgn}(q_e^\circ) \vec{q}_e - K_d^{\text{att}} \omega_e\right)$$

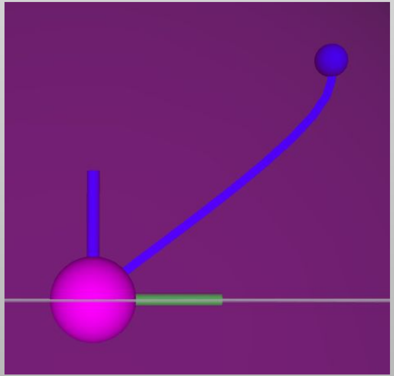
Results – Animations



Ex: Chain of integrators

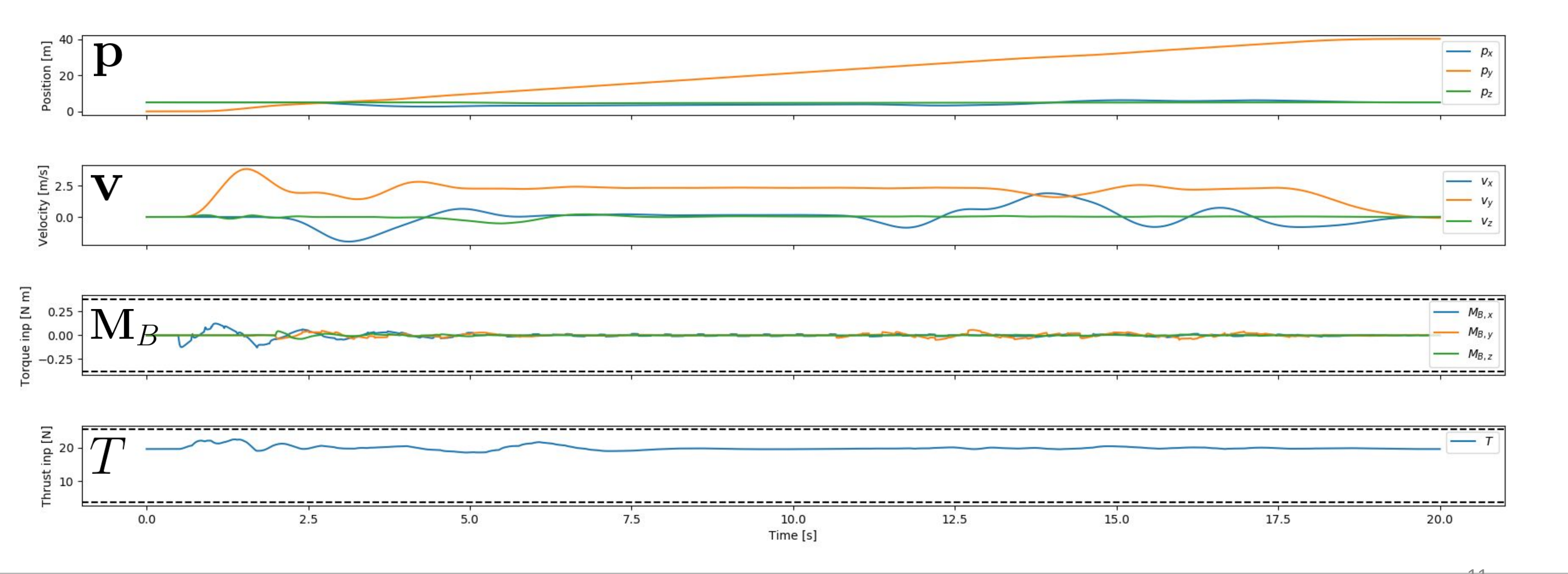


Ex: Quadrotor dynamics



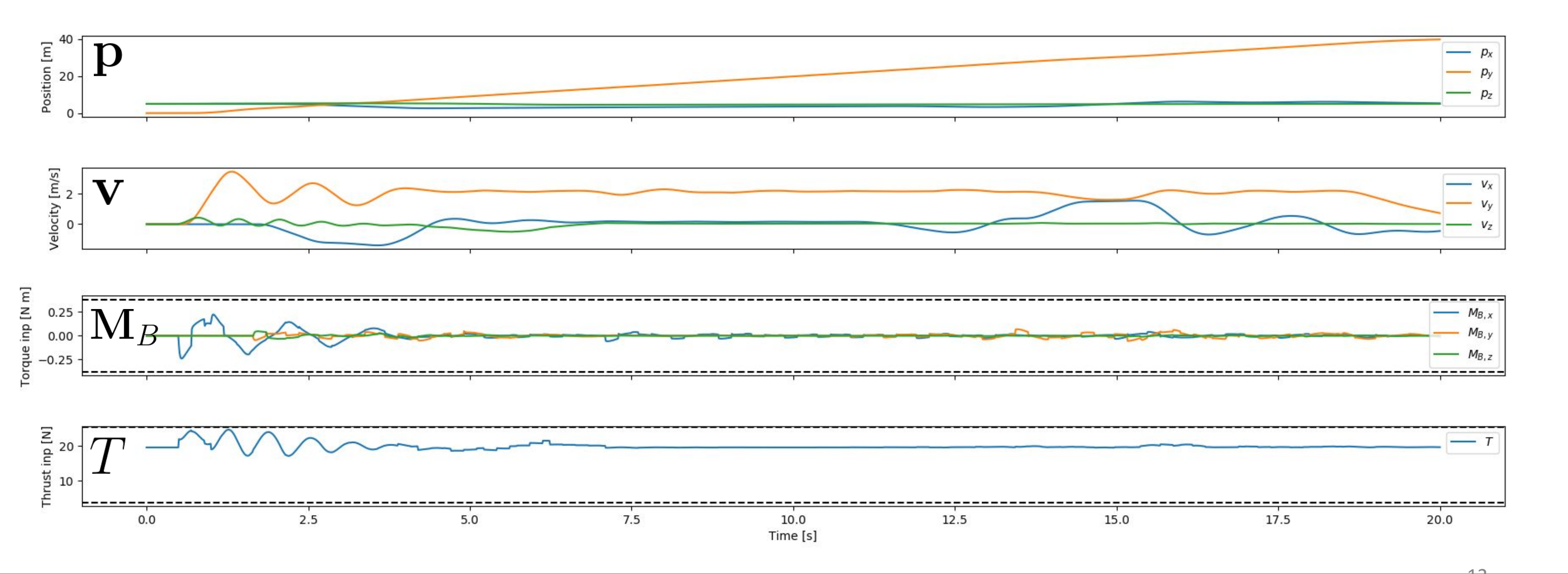
Results – State and Input Plots

Chain of integrators



Results – State and Input Plots

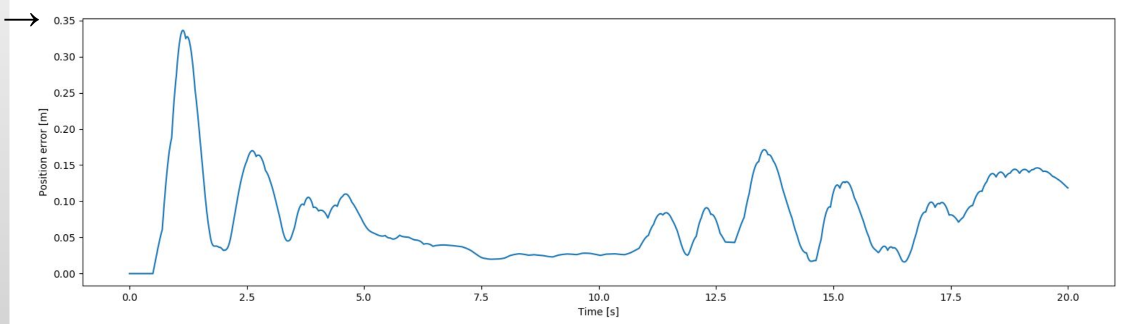
Quadrotor dynamics



Results – Position Error

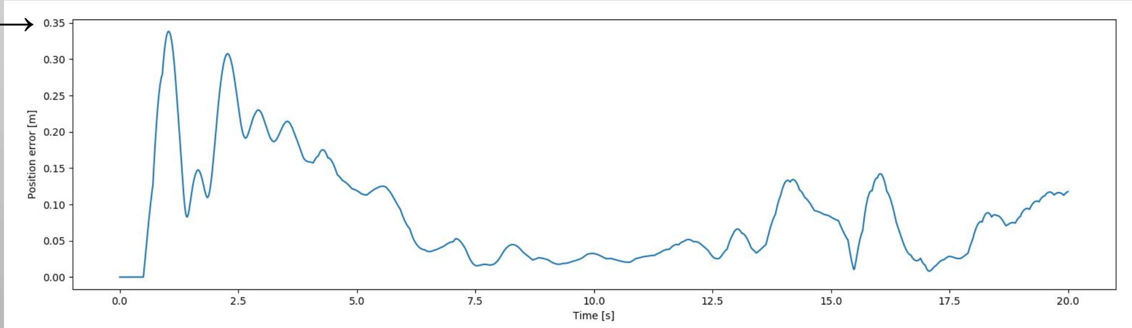
0.35m →

Chain of integrators



0.35m →

Quadrotor dynamics



Possible Improvements

- More thorough collision checking/constraints
- Theoretically guaranteed “tube” for tracking
- Properly integrated dynamics in trajectory optimization
- Plan of action in dead ends / mapping