Model Predictive Control-based Trajectory Planning for Quadrotors

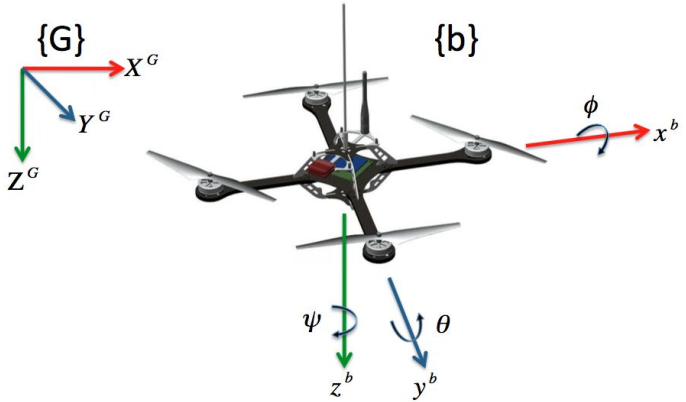
ME231A, 2023 Fall

Mintae Kim
Qiayuan Liao
Faris Tulbah
Ruiting Wang

Content

- **Introduction and system dynamics**
- Trajectory optimization and MPC
- Neural MPC
- Conclusion

Introduction



Problem

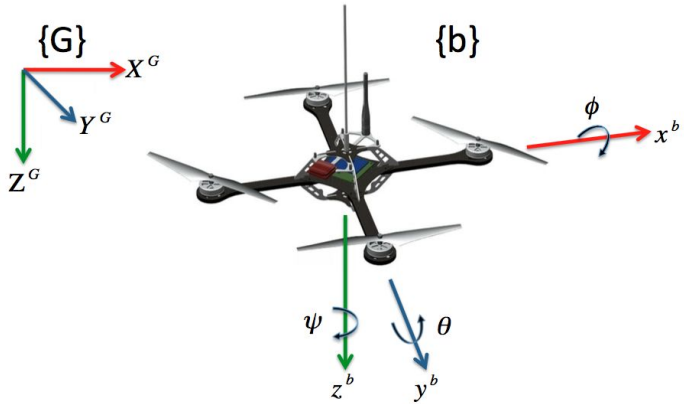
Trajectory planning for quadrotors. In this project, we focus on finding trajectory from a certain initial point to a given final point. After reaching the final spot, the quadrotors stop and hover without rotation.

Method

Trajectory optimization methods in MPC with knowledge of system dynamics

Neural MPC finds control input as a function of states, through neural networks

Dynamics



$\mathbf{x} \in \mathcal{R}^{12}$

- Position
- Row/Pitch/Yaw
- Velocity
- RPY rate

$\mathbf{u} \in \mathcal{R}^4$ — 4 motors speed

$$\ddot{\mathbf{p}} = \left(\mathbf{R}[0, 0, k_F \sum_{i=0}^{3} \omega_i^2] - [0, 0, mg]\right) m^{-1}$$

$$\ddot{\boldsymbol{\psi}} = \mathbf{J}^{-1} \left(\tau(l, k_F, k_T, [\omega_0^2, \omega_1^2, \omega_2^2, \omega_3^2]) - \dot{\boldsymbol{\psi}} \times \left(\mathbf{J}\dot{\boldsymbol{\psi}}\right)\right)$$

Content

- Introduction and system dynamics
- **Trajectory optimization and MPC**
- Neural MPC
- Conclusion

Problem formulation

The general trajectory optimization problem with continuous system dynamics

$$\min_{\mathbf{x}(t),\mathbf{u}(t)} \; p(\mathbf{x}(t_0+t_h)) + \int_{t_0}^{t_0+t_h} q(\mathbf{x}(t),\mathbf{u}(t),t)\,dt$$

$$\begin{aligned}
\text{s.t. } & \mathbf{x}(t_0) = \hat{\mathbf{x}}, \\
& \dot{\mathbf{x}}(t) = \mathbf{f}_c(\mathbf{x}(t),\mathbf{u}(t),t), \\
& \mathbf{g}(\mathbf{x}(t),\mathbf{u}(t),t) = \mathbf{0}, \\
& \mathbf{h}(\mathbf{x}(t),\mathbf{u}(t),t) \geq \mathbf{0},
\end{aligned}$$

Transcription methods

MPC framework

(Non)linear

Programing

Commercial solvers

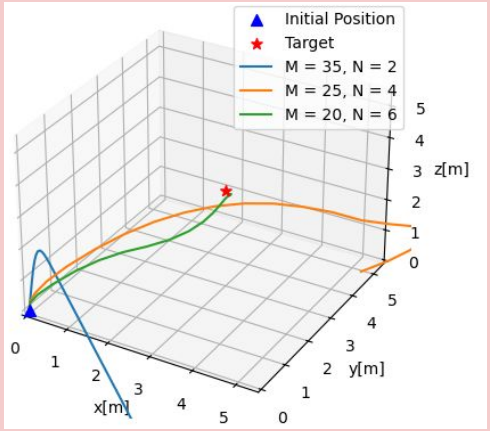
(CasADi + ipopt)

Trajectory Optimization in MPC

Cost function: $J = \alpha_1 J_{\text{input}} + \alpha_2 J_{\text{state}} + \alpha_3 J_{\text{final}}$

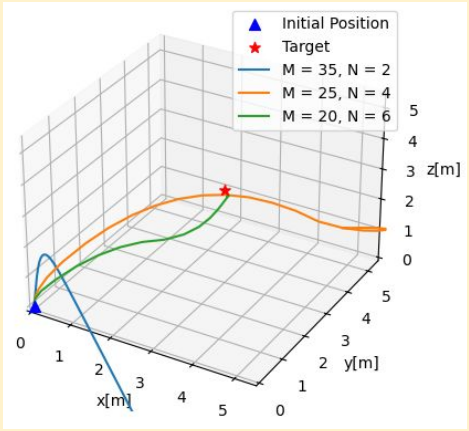
Direct transcription

$$\mathbf{x}_{k+1} = f_d(\mathbf{x}_k, \mathbf{u}_k)$$



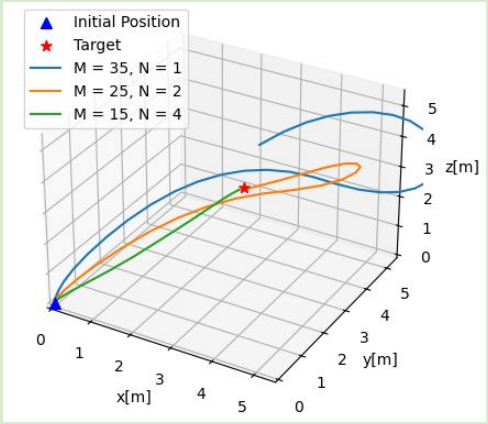
Single shooting

$$J(\mathbf{x}_k, \mathbf{u}_k) -> J(\mathbf{u}_k)$$



Direct collocation

$$\mathbf{u}_{c,k} = \frac{1}{2}(\mathbf{u}_k + \mathbf{u}_{k+1}),$$
$$\mathbf{x}_{c,k} = \frac{1}{2}(\mathbf{x}_k + \mathbf{x}_{k+1}) + \frac{h}{8}(\dot{\mathbf{x}}_k - \dot{\mathbf{x}}_{k+1}),$$
$$\dot{\mathbf{x}}_{c,k} = -\frac{3}{2h}(\mathbf{x}_k - \mathbf{x}_{k+1}) - \frac{1}{4}(\dot{\mathbf{x}}_k + \dot{\mathbf{x}}_{k+1}).$$



Duality-based Collision Avoidance

Minimum Distance Between Polytopes

$$d_i^*(\mathbf{x}) = \min_{\mathbf{y}^{\mathcal{R}}, \mathbf{y}^{\mathcal{O}_i}} \left\| \mathbf{y}^{\mathcal{R}} - \mathbf{y}^{\mathcal{O}_i} \right\|^2,$$
$$\text{s.t.} \quad A_{\mathcal{R}}(\mathbf{x})\mathbf{y}^{\mathcal{R}} \leq \mathbf{b}_{\mathcal{R}}(\mathbf{x}), \; A_{\mathcal{O}_i}\mathbf{y}^{\mathcal{O}_i} \leq \mathbf{b}_{\mathcal{O}_i}.$$

$\longrightarrow$ X $\longrightarrow$ $d^*(\mathbf{x}) \geq \alpha,$

Dual Problem

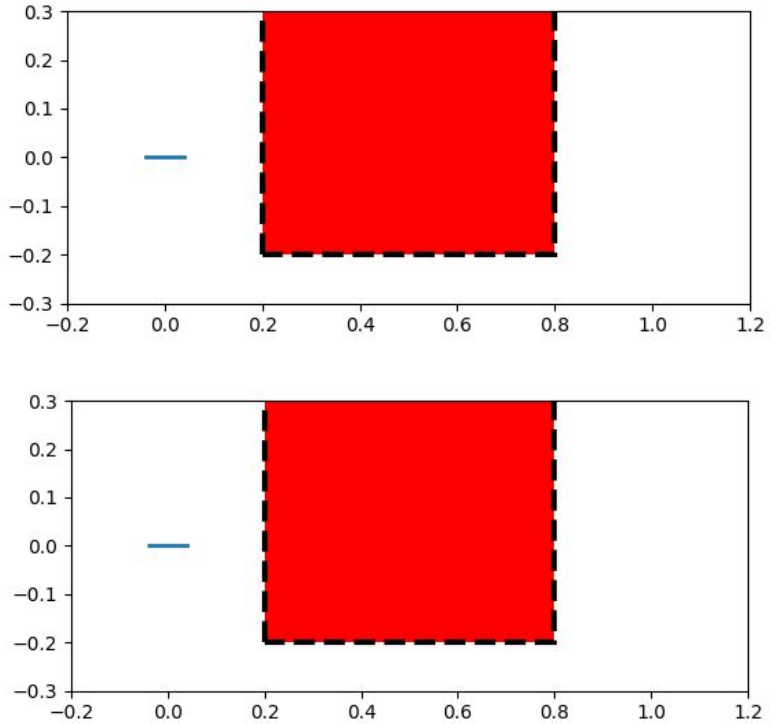
$$d^{\boxtimes}(\mathbf{x}) = \max_{\boldsymbol{\lambda}^{\mathcal{R}}, \boldsymbol{\lambda}^{\mathcal{O}_i}} -(\boldsymbol{\lambda}^{\mathcal{R}})^T \mathbf{b}_{\mathcal{R}}(\mathbf{x}) - (\boldsymbol{\lambda}^{\mathcal{O}_i})^T \mathbf{b}_{\mathcal{O}_i}$$
$$\text{s.t.} \quad A_{\mathcal{R}}^T(\mathbf{x})\boldsymbol{\lambda}^{\mathcal{R}} + A_{\mathcal{O}_i}^T \boldsymbol{\lambda}^{\mathcal{O}_i} = 0,$$
$$\left\| A_{\mathcal{O}_i}^T \boldsymbol{\lambda}^{\mathcal{O}_i} \right\|_2 \leq 1, \; \boldsymbol{\lambda}^{\mathcal{R}} \geq 0, \; \boldsymbol{\lambda}^{\mathcal{O}_i} \geq 0,$$

Don't need to Solve

$$d^{\square}(\mathbf{x}) = \max_{\boldsymbol{\lambda}^{\mathcal{R}}, \boldsymbol{\lambda}^{\mathcal{O}_i}} -(\boldsymbol{\lambda}^{\mathcal{R}})^T \mathbf{b}_{\mathcal{R}}(\mathbf{x}) - (\boldsymbol{\lambda}^{\mathcal{O}_i})^T \mathbf{b}_{\mathcal{O}_i}$$
$$\text{s.t.} \quad A_{\mathcal{R}}^T(\mathbf{x})\boldsymbol{\lambda}^{\mathcal{R}} + A_{\mathcal{O}_i}^T \boldsymbol{\lambda}^{\mathcal{O}_i} = 0,$$
$$\left\| A_{\mathcal{O}_i}^T \boldsymbol{\lambda}^{\mathcal{O}_i} \right\|_2 \leq 1, \; \boldsymbol{\lambda}^{\mathcal{R}} \geq 0, \; \boldsymbol{\lambda}^{\mathcal{O}_i} \geq 0,$$

Duality based Collision Avoidance

$$
\begin{aligned}
\min_{\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\lambda}_k^{\mathcal{R}}, \boldsymbol{\lambda}_k^{\mathcal{O}_i}} \quad & q(\mathbf{x}_N) + \sum_{k=0}^{N-1} p(\mathbf{x}_k, \mathbf{u}_k) \\
\text{s.t.} \quad & \mathbf{x}_{k+1} = \mathbf{f}_d(\mathbf{x}_k, \mathbf{u}_k), \quad \forall k \in [0, N-1], \\
& \mathbf{x}_0 = \mathbf{x}(0), \\
& -(\boldsymbol{\lambda}_k^{\mathcal{R}})^T \mathbf{b}_{\mathcal{R}}(\mathbf{x}) - (\boldsymbol{\lambda}_k^{\mathcal{O}_i})^T \mathbf{b}_{\mathcal{O}_i} \geq \alpha, \\
& A_{\mathcal{R}}^T(\mathbf{x}) \boldsymbol{\lambda}_k^{\mathcal{R}} + A_{\mathcal{O}_i}^T \boldsymbol{\lambda}_k^{\mathcal{O}_i} = 0, \\
& \left\| A_{\mathcal{O}_i}^T \boldsymbol{\lambda}_k^{\mathcal{O}_i} \right\|_2 \leq 1, \\
& \boldsymbol{\lambda}_k^{\mathcal{R}} \geq 0, \ \boldsymbol{\lambda}_k^{\mathcal{O}_i} \geq 0,
\end{aligned}
$$



Content

- Introduction and system dynamics
- Trajectory optimization and MPC
- **Neural MPC**
- Conclusion

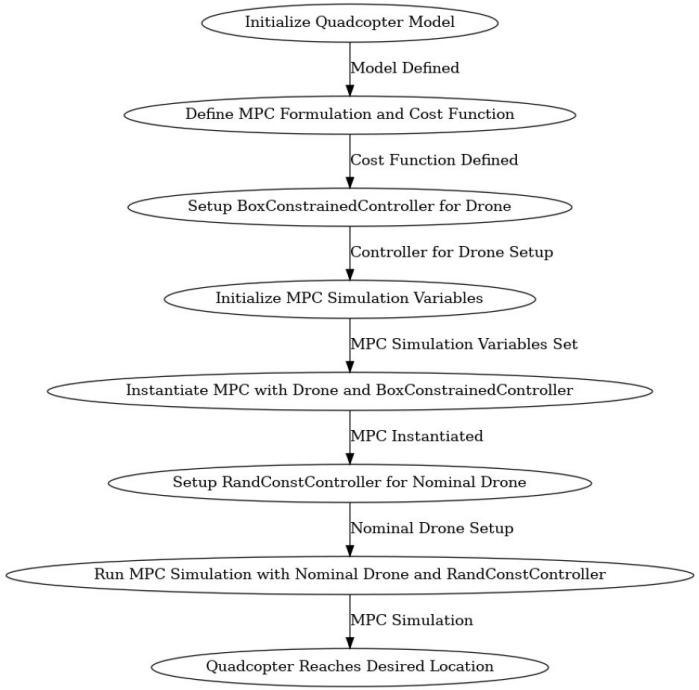
Neural Controllers in MPC for Improved Drone Navigation

Box Constrained Controller:

- Architecture: Utilizes a multi-layer neural network with Softplus and Tanh activations, ensuring bounded control inputs.
- Functionality: This controller is pivotal for handling complex, nonlinear control tasks, especially in fluctuating environments. It scales inputs and outputs to align with drone operational parameters.
- MPC Integration: Key component in MPC, optimizing drone trajectories under dynamic conditions.

RandConstController:

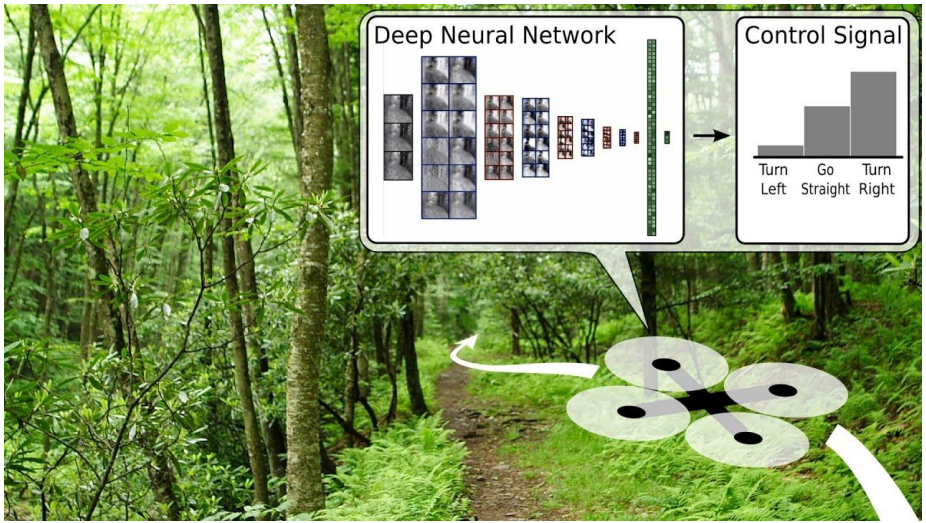
- Simplicity: Generates a constant control signal, designed for scenarios requiring steady, unchanging inputs.
- Usage: Ideal for residual propagation and forward simulation steps in MPC, particularly in more predictable or controlled settings.
- Implementation: Provides a uniform control signal within specified bounds, simplifying the control mechanism.



Learning Dynamics in MPC with Neural Network Controllers

BoxConstrainedController Learning Process:

- Role in MPC: The controller generates control inputs based on the current state of the drone. These inputs are optimized through the MPC framework to achieve desired future states.
- Learning Mechanism:
    - The neural network learns to predict control actions that minimize a defined cost function over the MPC's planning horizon.
    - The Adam optimizer is used to iteratively adjust the neural network weights, reducing the difference between the predicted states and desired trajectory.
- Handling Complex Dynamics: The controller's ability to adapt its outputs based on feedback allows for efficient handling of environmental perturbations and dynamic flight conditions.



https://www.youtube.com/watch?v=umRdt3zGgpU

RandConstController Application in MPC:

- This controller is typically used for simpler control scenarios within MPC, where the objective is to maintain a steady state or for initial testing phases.
- It can be particularly useful in scenarios where the environment is relatively stable and predictable.

Quadcopter Navigation Amidst Perturbations

Mass Fluctuations:

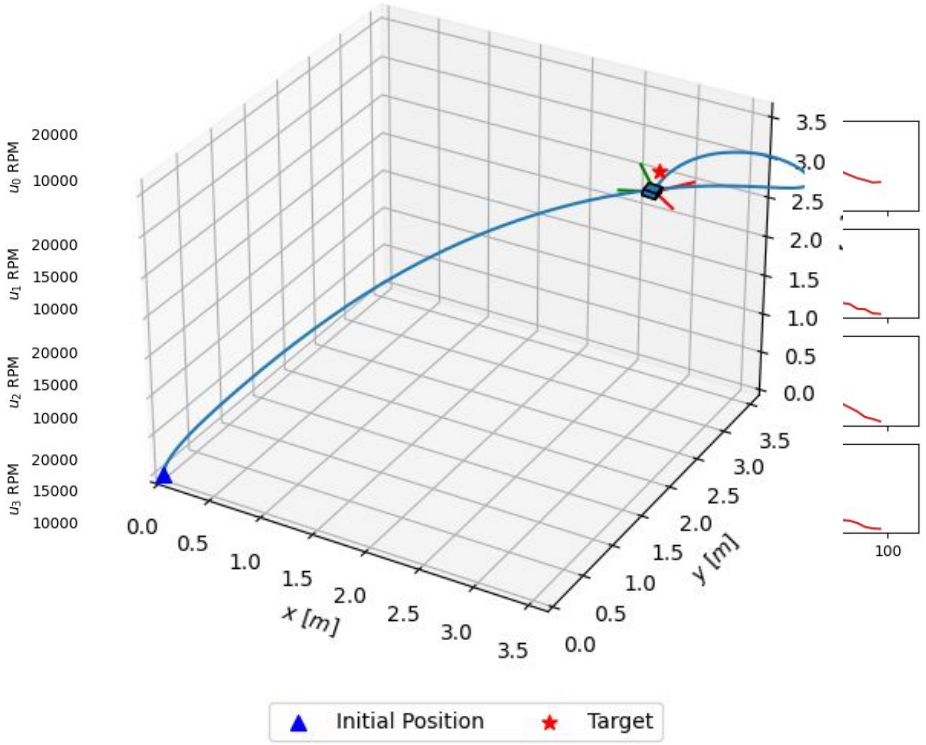
- Perturbations simulate a 5% random mass fluctuation to reflect changes in payload or fuel weight.

Motor Efficiency Fluctuations:

- Motor efficiency factors are varied between 95% to 100% to represent real-world motor performance discrepancies.

Wind Gust Simulation:

- Velocity perturbations with a standard deviation of 0.1 are introduced to mimic the effect of wind gusts on the drone.



Content

- Introduction and system dynamics
- Trajectory optimization and MPC
- Neural MPC
- **Conclusion**

Conclusion

- Classic MPC framework is implemented and evaluated for autonomous quadrotor trajectory planning
  - 3 transcription methods
  - Collision avoidance

- We proposed a novel Neural MPC framework
  - Neural MPC finds control input as a function of states, through neural networks
  - No explicit system dynamics information are used
  - The model performs well with perturbations