Fast Online Adaptive Neural MPC via Meta-Learning

Yu Mei[1], Xinyu Zhou[1], Shuyang Yu[2], Vaibhav Srivastava[1] and Xiaobo Tan[1]

[1] Department of Electrical and Computer Engineering
[2] Department of Computer and Science Engineering

Michigan State University

Oct 07, 2025

MECC 2025

Motivation

Drone delivery [1]

Soft robots [2]

Challenges

Dynamics and/or disturbances that are unknown, difficult-to-model:

- Drone delivery: unknown weights, unmodeled aerodynamic effects
- Soft robots: Hyperelasticity, viscoelasticity and hysteresis

[1] Saunders et al., JFR '24
[2] Haggerty, Science Robotics '23

Learning-based Optimal Control

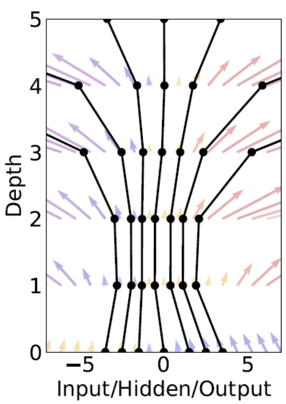
A general continuous-time, partial-known nonlinear dynamic system:

$$\begin{aligned}\dot{x}(t) &= f(x(t), u(t)),\\ &= \underbrace{f_{\text{nom}}(x(t), u(t))}_{\text{known nominal dynamics}} + \underbrace{f_{\text{res}}(x(t), u(t))}_{\text{Unknown residual dynamics}}\end{aligned}$$

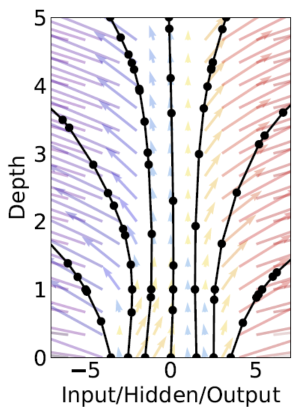
Neural ODEs [6]

Instead of specifying a discrete sequence of system, it parameterizes the derivative of the system dynamics using a neural network.

$$x_{k+1} = x_k + f_\theta(x_k)$$



$$\frac{dx(t)}{dt} = f_\theta(x(t), t)$$



[6] Chen et al., NeurIPS '19

Learning-based MPC

Real-time Neural MPC [7]

Real-time Neural ODE + MPC based on *L4CasADi*

Robotic system with state $x = [x_1, x_2]^T$:

$$\dot{x}(t) = f_{\text{nom}}(x(t), u(t)) + f_{\text{res}}(x(t), u(t))$$
$$= \begin{bmatrix} x_2 \\ \hat{\dot{x}}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ f_{\text{NN}}(x, u; \theta) \end{bmatrix}$$
$$= \begin{bmatrix} x_2 \\ \hat{\dot{x}}_2 + f_{\text{NN}}(x, u; \theta) \end{bmatrix}$$

collect and learn offline

Nonlinear MPC:

$$\begin{aligned} \min_{\mathbf{u}_{0:N-1}} \quad & \sum_{k=0}^{N-1} \ell(x_k, u_k) + m(x_N) \\ \text{s.t.} \quad & x_{k+1} = \phi(x_k, u_k, f, \delta t), \quad \forall k = 0, \dots, N-1, \\ & x_0 = x(t) \\ & g(x_k, u_k) \le 0, \quad \forall k = 0, \dots, N-1 \end{aligned}$$

Rapid integration

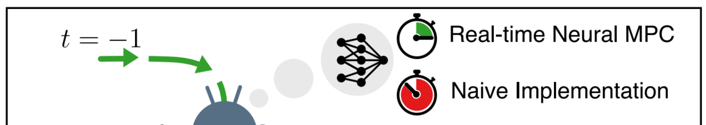


Figure: Real-time Neural MPC

Problems

- data collection and training can be expensive and time-consuming
- may not generalize to unseen environments

[7] Salzmann et al., RAL '23; Alhaddad et al., ICRA '24; Gao et al., TIV '24; ...

Learning-based MPC

Goal

Fast Online adaptive neural MPC, where f_{res} can be learned and refined rapidly using on-the-fly collected data.

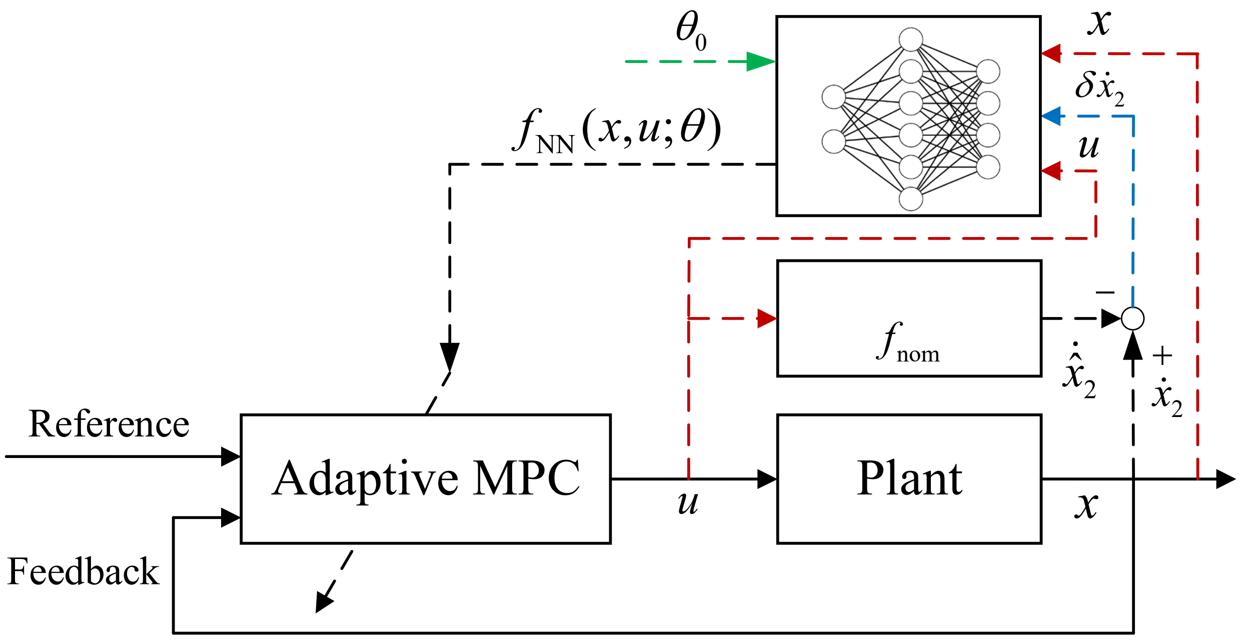


Figure: Diagram of online adaptive neural MPC

- Loss function for training:

$$\mathcal{L}_{\mathrm{MAE}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left| f_{\mathrm{NN}}(x_i, u_i; \theta) - \delta\dot{x}_{2,i} \right|$$

Fast Adaptation via Model-Agnostic Meta-Learning (MAML)

Problem

Training to convergency is time-consuming, especially for deep or large NN.

Model-Agnostic Meta-Learning (MAML) [8]

A strategy to pretrain the NN parameters, which effectively adapts to previously unseen dynamics, using only a small number of data.

(*few-shot learning*)

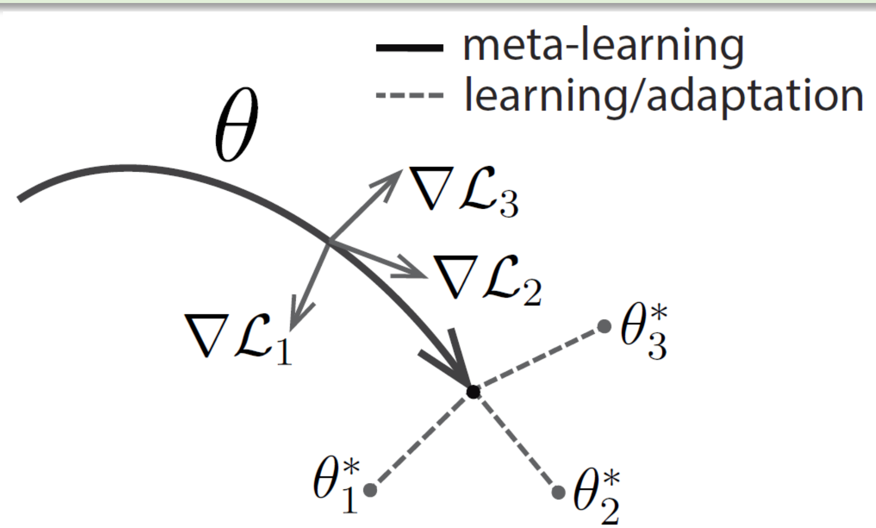
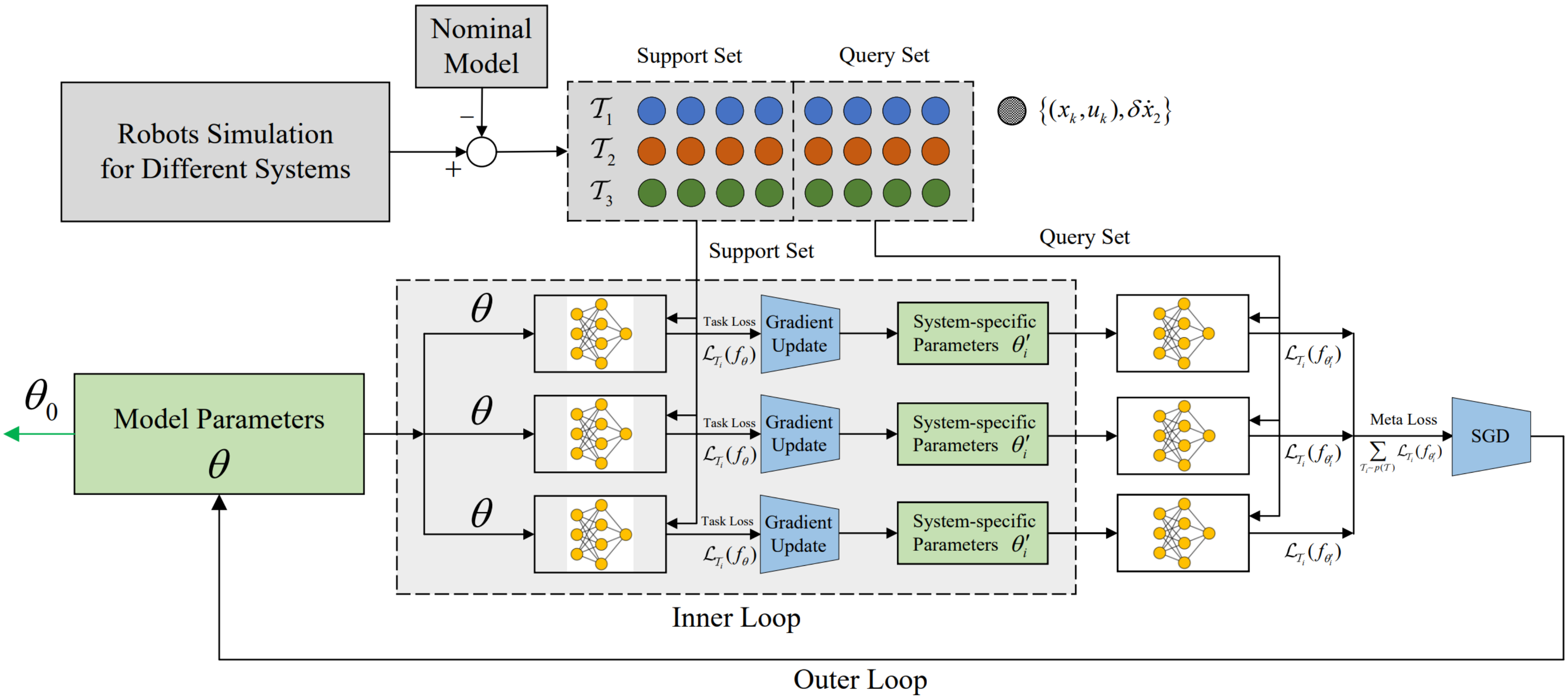


Figure: Diagram of MAML

[8] Finn et al., ICML '17; Shi et al., NeurIPS '21; Richards et al., IJRR ‘23.

Fast Adaptation via Model-Agnostic Meta-Learning (MAML)



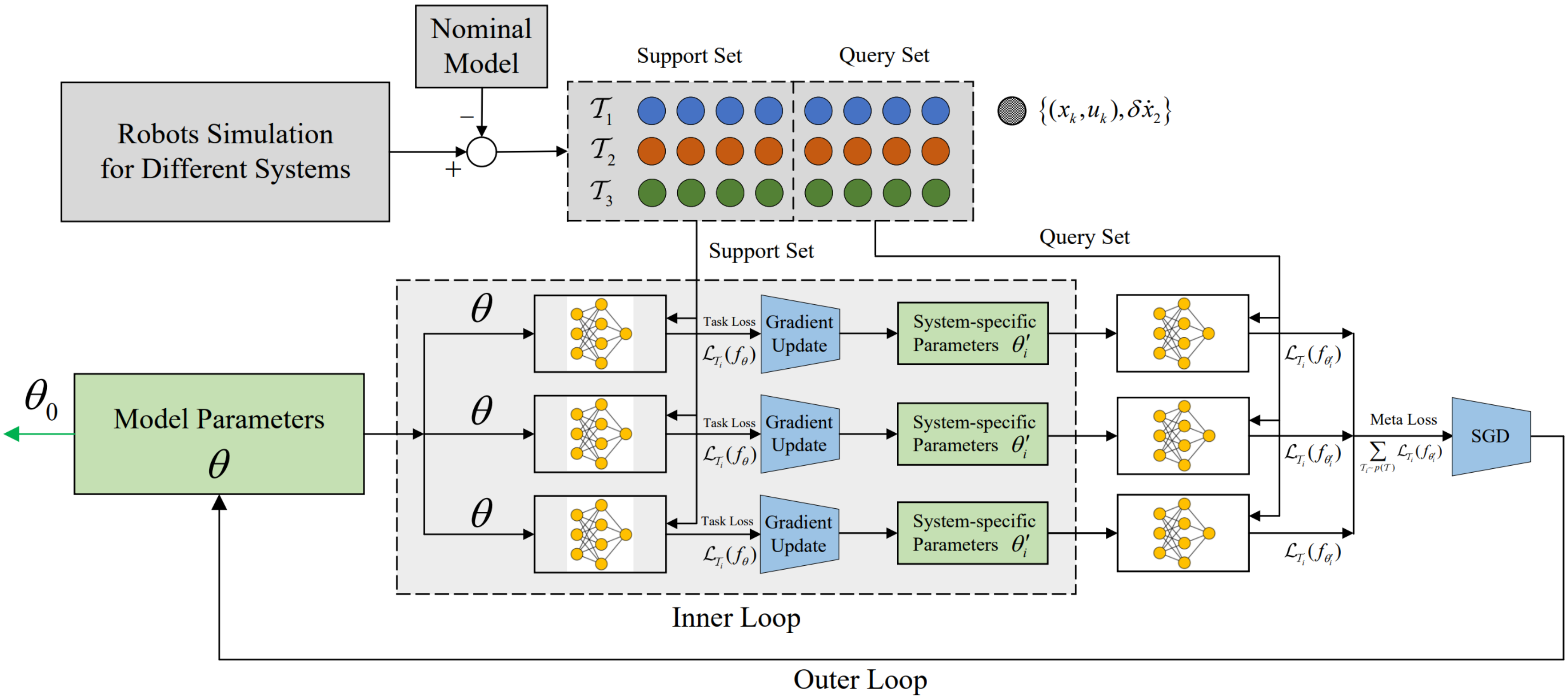
Procedures:

a) Generate dataset for subsystem $\mathcal{T}_i$ (only needs 2K samples for K-shot learning)

b) Inner loop performs single gradient descent step:

$$\theta_i' = \theta - \alpha\nabla_\theta\mathcal{L}_{\mathcal{T}_i}^{\text{support}}(f_\theta)$$

Fast Adaptation via Model-Agnostic Meta-Learning (MAML)



Procedures:

c) Outer loop performs sum-up of gradients across subsystems via SGD:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}^{\text{query}}(f_{\theta_i'})$$

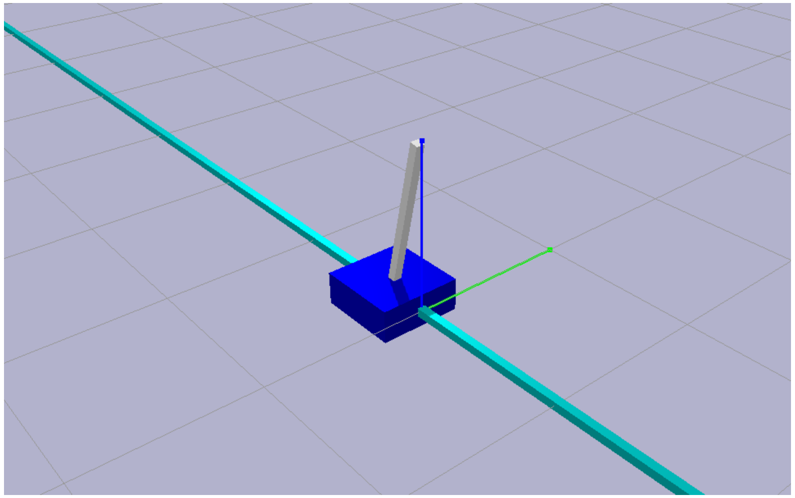
d) Meta-learned initial parameters

Goal:

- Stabilize the cart-pole around the upright position of the pole

Setup:

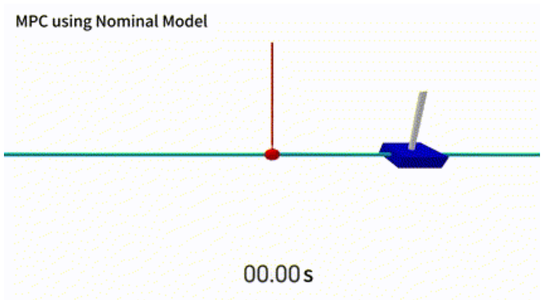
- The model parameters (pole length & mass, cart mass) are inaccurate



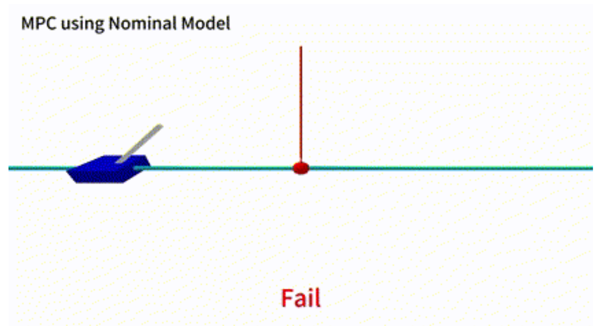
Three comparative controllers:

1. Nominal MPC using only the nominal model
2. Neural MPC with a newly initialized MLP
3. Neural MPC with a meta-learned MLP

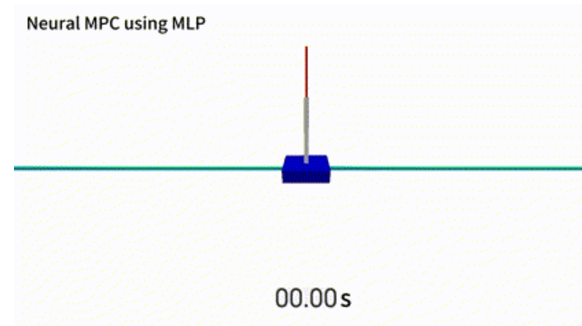
Simulations on Cart-Pole System



Nominal MPC



Nominal MPC + MLP



Nominal MPC + MetaMLP

Table IV. Stabilization success and speed over 100 randomized-initial-state trials. Time-to-stabilize is averaged over successful trials only.

Controller	**Success rate**	**Time-to-stabilize [s]**
Nominal MPC	0%	—
Neural MPC + MLP	98%	3.72 ± 1.36
Neural MPC + MetaMLP	100%	$\boxed{2.55 \pm 0.49}$

Goal:

- Stabilize the rotor in one point

Setup:

- The model parameters (mass, moment of inertia) are inaccurate

Simulations on 2D Quadrotor System Stabilization

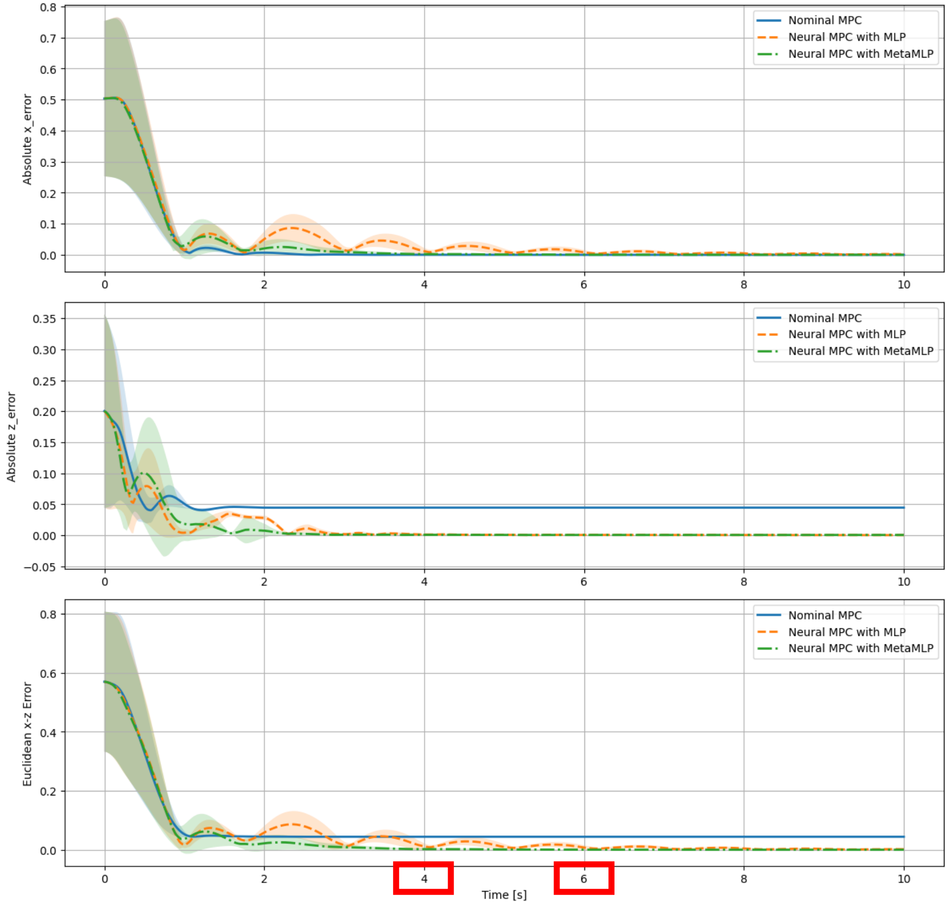


Figure: Averaged errors over 20 trials

Results:

- Our method achieves the fastest stabilization without steady-state errors compared to others:
  - ➢ Nominal MPC suffers from steady errors
  - ➢ Nominal MPC with MLP exhibits larger oscillations

Simulations on 2D Quadrotor Tracking

Goal:

- Track the reference trajectory

Simulations on 2D Quadrotor Tracking

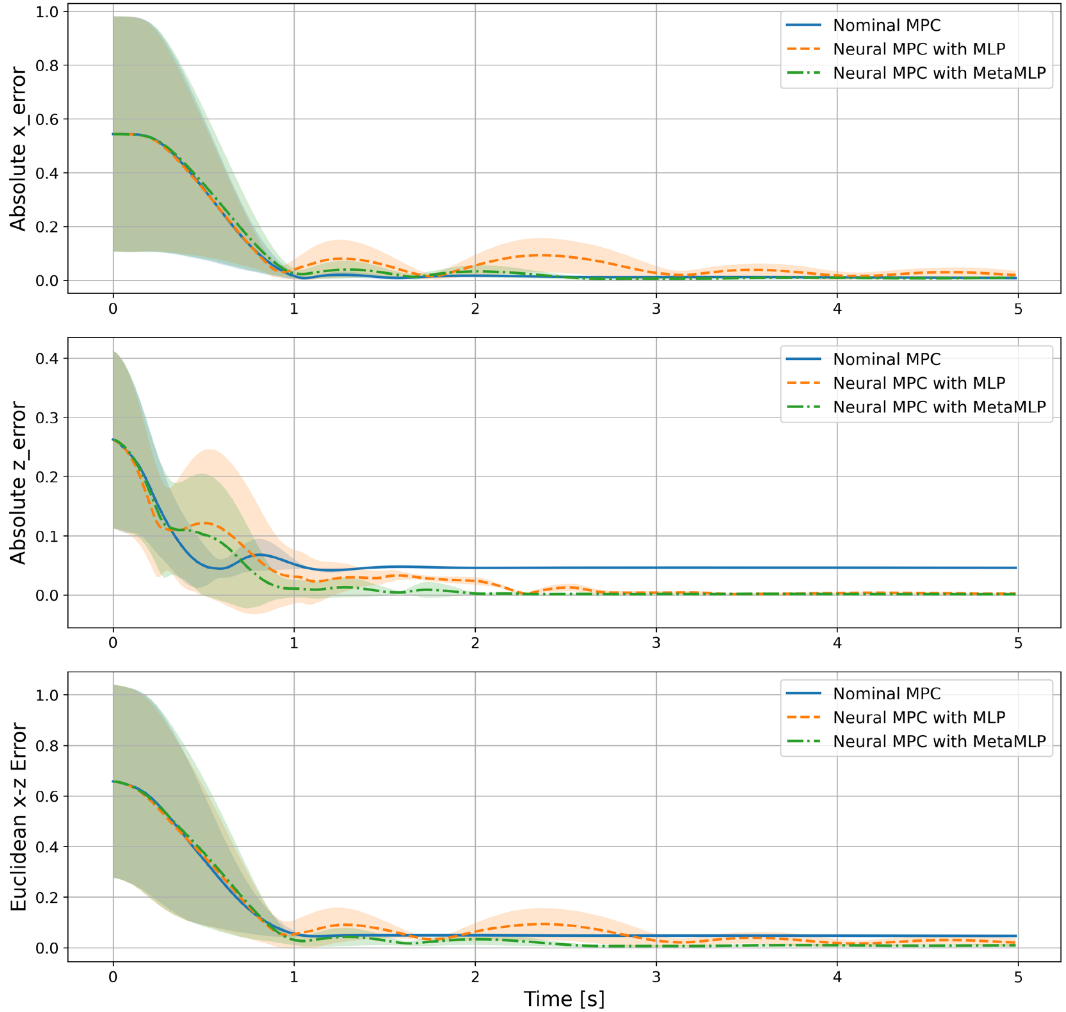


Figure: Averaged errors over 20 trials

Same comparison results.

Computational efficiency

Both the Cart-Pole and 2D quadrotor systems demonstrate that Neural MPC with either an MLP or a MetaMLP can sustain a control frequency of 45-50Hz.

Conclusions and Future Work

Conclusions

A fast online adaptive neural MPC framework for robotic systems using Model-Agnostic Meta-Learning (MAML) to enable efficient few-shot learning.

Future Work

1. Continual learning across time-varying uncertainty
2. Real robots application

Thank you!

Code

Video

Acknowledgements:

This research was supported in part by National Science Foundation awards (CNS 2237577 and CMMI 1940950).

State of the art

Robust Control [3]

- ❑ Guarantees closed-loop stability for a uncertain plant, But
- ➢ conservative since assuming worst-case disturbances

Adaptive Control [4]

- ❑ Estimates uncertainty and compensates control input, But
- ➢ cannot handle unmodeled uncertainty

Learning-based Control [5]

- ❑ Collects data offline and train the data-driven models:
  - Neural networks (MLP, LSTM)
  - Gaussian process regression
  - Koopman operator, But
- ➢ data collection and training can be expensive and time-consuming
- ➢ may not generalize to unseen environments

[3] Wang et al., TMech '19; Yang et al., TNNLS '19; Patterson., RAL '22; …
[4] Chowdhary et al., TNNLS '15; Yang et al., TNNLS '19; Patterson., RAL '22; …
[5] Torrente et al., RAL '21; Salzmann et al., RAL '23; Bruder et al., TRO '21, IJRR '25; Liu et al., IEEE Ctrl Sys '18.…