Synthesis of Model Predictive Control and Reinforcement Learning

Rudolf Reiter

University of Zurich

Predict, Control, Learn: Combining Model Predictive Control and Reinforcement Learning
Tutorial Session at ECC 2026, Reykjavík, July 2026

universität freiburg

NTNU | Norwegian University of Science and Technology

University of Zurich[UZH]

ROBOTICS & PERCEPTION GROUP

Tutorial Roadmap

	Talk	*Presenter*	*Duration*
✓	Markov Decision Processes – a unifying perspective on MPC and RL	Jasper Hoffmann	20min
➤	**Synthesis of Model Predictive Control and Reinforcement Learning**	**Rudolf Reiter**	**30min**
	Reinforcement Learning over MPC: Why does it work?	Dirk Reinhardt	30min
	Differentiable NMPC with `acados`	Katrin Baumgärtner	20min
	MPCRL with `leap-c`	Jasper Hoffmann	20min

At the end of each talk there will be time for questions!

Tutorial webpage for slides and more!

Synthesis of MPC and RL

Motivation:

- MPC and RL are complementary approaches for solving MDPs
- Nearly orthogonal advantages: weaknesses of one are strengths of the other
- Growing research interest in combination methods

Goals of this presentation:

- Categorize existing work based on actor-critic RL framework
- Show real world success stories

Synthesis of Model Predictive Control and Reinforcement Learning: Survey and Classification*

Rudolf Reiter[a,**], Jasper Hoffmann[b,**], Dirk Reinhardt[d], Florian Messerer[a], Katrin Baumgärtner[a], Shambhuraj Sawant[d], Joschka Bödecker[a], Moritz Diehl[b,c], Sebastien Gros[d]

[a]Department of Microsystems Engineering, University of Freiburg, Freiburg im Breisgau, 79110, Baden-Württemberg, Germany
[b]Department of Computer Science, University of Freiburg, Freiburg im Breisgau, 79110, Baden-Württemberg, Germany
[c]Department of Mathematics, University of Freiburg, Freiburg im Breisgau, 79104, Baden-Württemberg, Germany
[d]Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim, 7034, Norway

Outline

Outline

Parameterized Nonlinear Program (Optimal Control Problem)

Parameterized NLP with parameters ϕ:

OCP:

$$
\begin{aligned}
\min_{z} \quad & L_\phi(z) \\
\text{s.t.} \quad & x_0 = s, \\
& x_{k+1} = f_\phi^{\text{MPC}}\big(x_k, u_k\big),\ 0 \le k < N, \\
& 0 \le h_\phi^{\text{MPC}}\big(x_k, u_k\big),\ 0 \le k < N, \\
& 0 \le \tilde{h}_\phi^{\text{MPC}}\big(x_N\big),
\end{aligned}
$$

Objective:

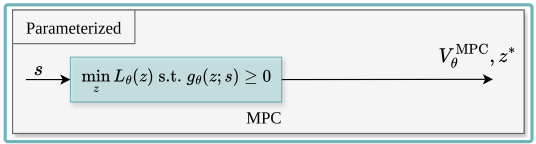
$$
L_\phi(z) := \bar{V}_\phi^{\text{MPC}}(x_N) + \sum_{k=0}^{N-1} l_\phi^{\text{MPC}}\big(x_k, u_k\big).
$$

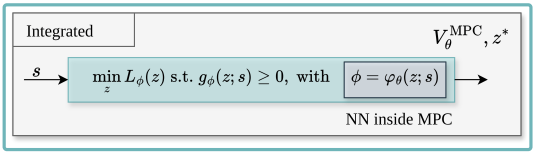
Short notation:

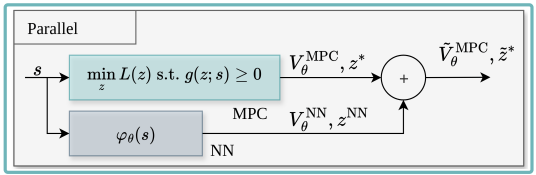
$$
\min_{z} L_\phi(z) \quad \text{s.t.} \quad g_\phi(z; s) \ge 0.
$$

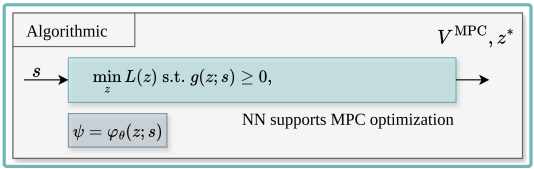
Parameters ϕ can affect: objective L_ϕ, dynamics f_ϕ^{MPC}, and constraints h_ϕ^{MPC}.
Note: g_ϕ encodes both dynamics and constraints.

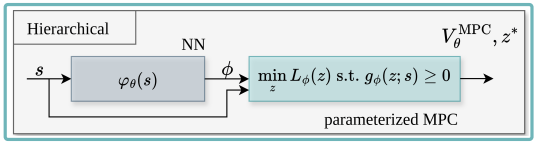
Inference Architectures



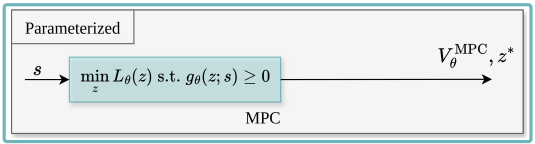


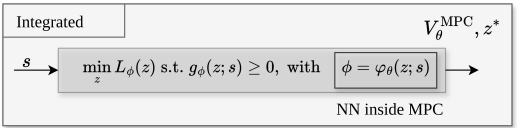


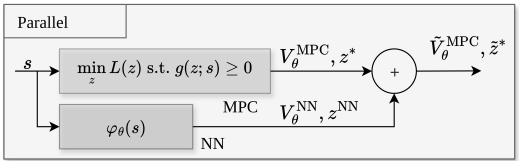


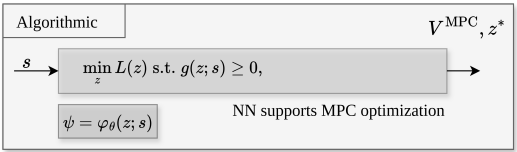


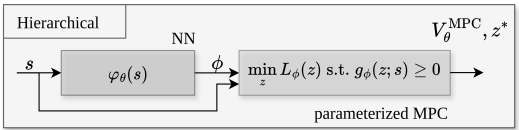
Parameterized Architecture











Parameterized Architecture

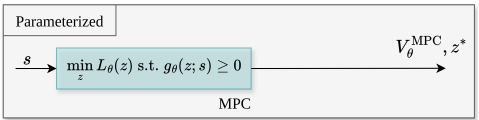
Architecture: Parameters θ are constant and independent of decision variables z or state s

Key characteristics:

- Parameters learned offline, fixed during deployment
- Optimization-friendly MPC formulation
- No evaluation of highly nonlinear functions

Advantages:

- ✓ Preserves MPC problem structure
- ✓ No NN evaluation during inference
- ✓ Suitable for standard MPC solvers
- ✓ Lower computational complexity
- ✗ Less flexibility

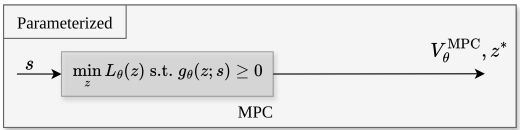


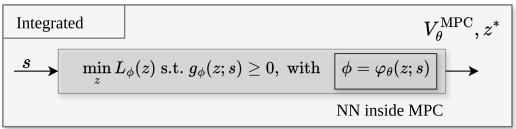
Examples:

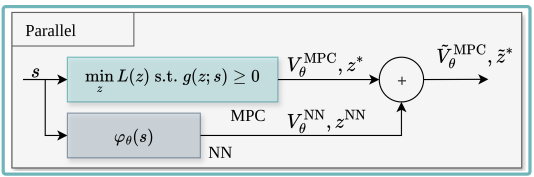
- Simplified models tuned in RL loop for building energy control [a]

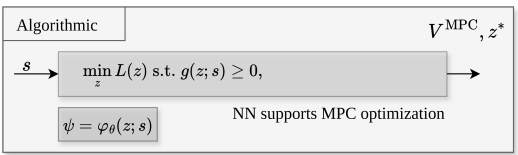
[a] **Cai et al.**(2023), *"A Learning-Based Model Predictive Control Strategy for Home Energy Management Systems"*

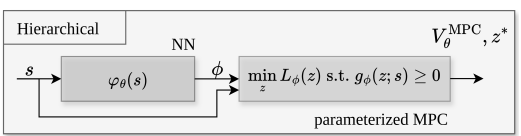
Parallel Architecture











Parallel Architecture

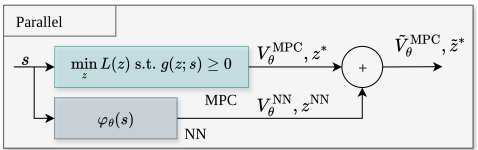
Architecture: MPC problem usually not parameterized; NN evaluated in parallel to correct MPC output

Key characteristics:

- MPC problem typically not parameterized
- NN evaluated in parallel to MPC optimization
- NN output corrects optimal solution or value function
- MPC and NN operate independently

Trade-offs:

- No differentiation through optimization ✓
- Potentially unsafe actions due to perturbation ✗



Examples:

- Parallel architecture for value function approximation[a]

[a] **Bhardwaj, Choudhury, and Boots**(2021), *"Blending MPC & Value Function Approximation for Efficient Reinforcement Learning"*

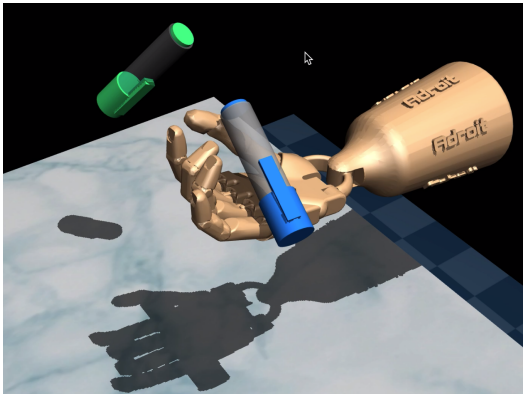
Parallel: Example

BLENDING MPC & VALUE FUNCTION APPROXIMATION FOR EFFICIENT REINFORCEMENT LEARNING

Mohak Bhardwaj[1] **Sanjiban Choudhury**[2] **Byron Boots**[1]

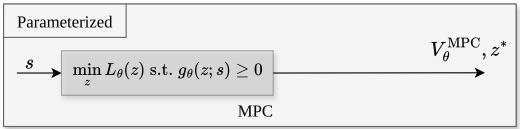
[1] University of Washington [2] Aurora Innovation Inc.

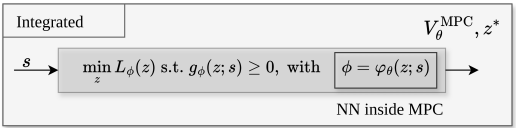
- Use MPC for local value estimates
- NN provides corrections
- Blend both estimates via weighting factor λ
- Improves data efficiency and adds structure
- Evaluated on 24DOF dexterous hand (fit pose) among others

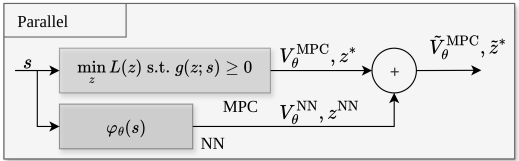


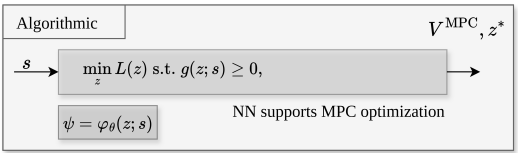
[Image: Bhardwaj et al. 2020]

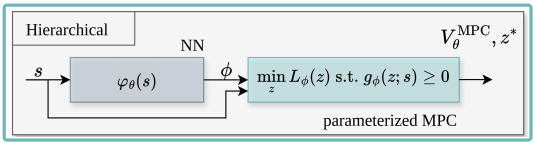
Hierarchical Architecture











Hierarchical Architecture

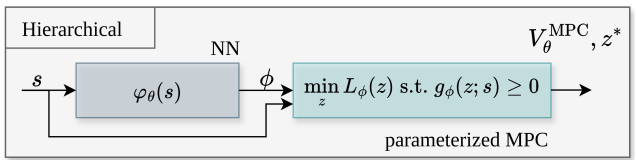
Architecture: NN provides input to MPC, parameter ϕ depends on state via $\phi = \varphi_\theta(s)$

Key characteristics:

- NN evaluated before solving optimization problem
- Function $\varphi_\theta(s)$ evaluated independently of MPC
- Nonlinearity does not affect optimization problem

Tradeoffs:

- ✓ Optimization not harder to solve
- ✓ Gradient-Based MPC solvers work
- ✗ MPC theory may not hold



Examples: Modification of:

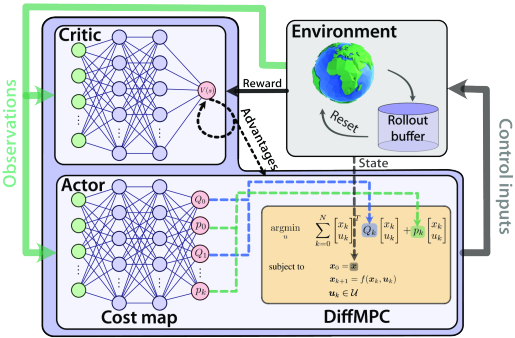
- Reference trajectory [a]
- Cost [b] [c]
- Constraints [d]

[a] **Brito et al.**(2021), *"Where to go Next: Learning a Subgoal Recommendation Policy for Navigation in Dynamic Environments"*
[b] **Reiter, Hoffmann, et al.**(2023), *"A Hierarchical Approach for Strategic Motion Planning in Autonomous Racing"*
[c] **Romero, Song, and Scaramuzza**(2024), *"Actor-Critic Model Predictive Control"*
[d] **Jacquet and Alexis**(2024), *N-MPC for Deep Neural Network-Based*

Hierarchical: Example

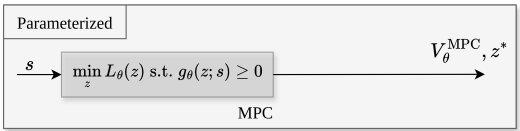
Actor-Critic Model Predictive Control

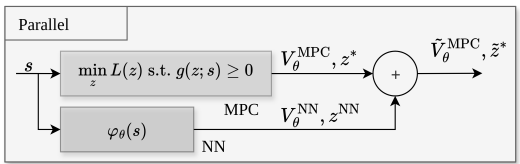
Angel Romero, Yunlong Song, Davide Scaramuzza

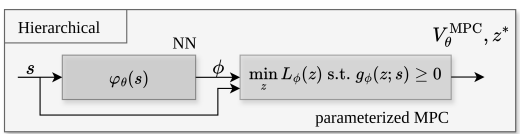


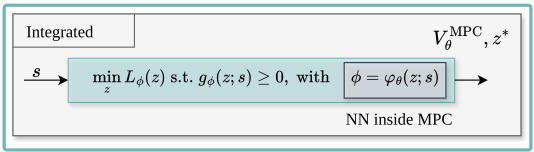
[Image: Romero et al. 2024]

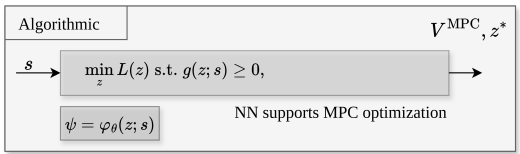
Integrated Architecture











Integrated Architecture

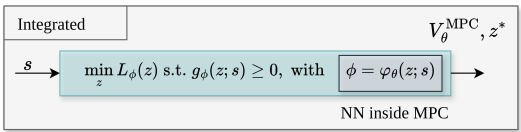
Architecture: Parameters depend on both state s and optimization variables

Key characteristics:

- NN is embedded in the optimization problem
- Cannot be evaluated separately
- May output value V_θ^{MPC} or optimal decision variables $z^\star$

Tradeoffs:

- ✓ Highly expressive
- ✗ Highly nonlinear, possibly nonconvex
- ✗ Computationally expensive
- ✗ Requires $\nabla_z \varphi_\theta(z; s)$



Examples:

- Latent space MPC [a] [b]
- Learned nonlinear residual dynamics [c]
- Cost function [d]

[a] **Lowrey et al.**(2019), *"Plan Online, Learn Offline: Efficient Learning and Exploration via Model-Based Control"*
[b] **N. A. Hansen, Su, and Wang**(2022), *"Temporal Difference Learning for Model Predictive Control"*
[c] **Salzmann et al.**(2023), *Learning for CasADi: Data-driven models in numerical optimization*
[d] **Seel et al.**(2022), *"Convex Neural Network-Based Cost Modifications for Learning Model Predictive Control"*

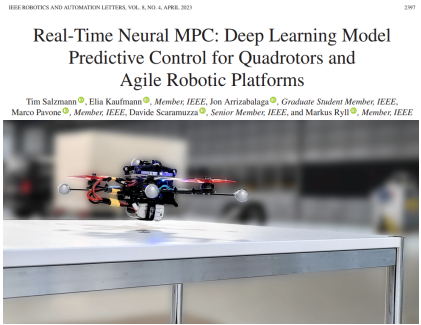
Integrated: Example

Real-Time Neural MPC:

- Learn NN residual from data
- Use model for MPC planning
- Real-time feasible
- Tracking error reduced by 82%

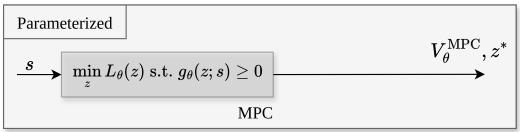
Anyone wants to try? Wrappers for acados and CasADi available:

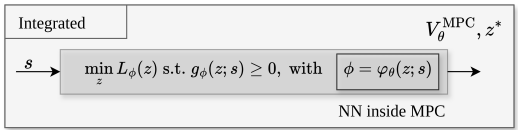
[l4casadi] [l4acados]

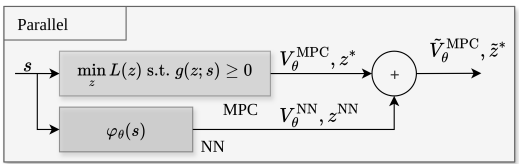


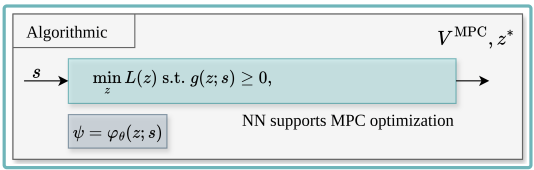
[Image: Salzmann et al. 2023]

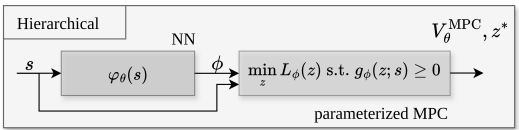
Algorithmic Architecture











Algorithmic Architecture

Architecture: RL policy guides the MPC optimization algorithm

Key characteristics:

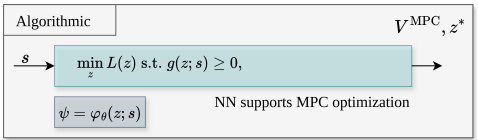
- RL policy alters solver hyperparameters ψ
- Optimization problem remains unchanged
- Examples of hyperparameters:
  - Initial guess (MPPI/SQP)
  - Active-set predictions
  - Horizon length

Tradeoffs:

✓ Improves online computation time

✓ Can help escape local minima

✗ Increased complexity



Examples:

- MPPI Warm-starting [a]
- SQP-RTI Warm-starting [b]

[a] **N. Hansen, Su, and Wang**(2024), *"TD-MPC2: Scalable, Robust World Models for Continuous Control"*

[b] **Reiter, Ghezzi, et al.**(2025), *"AC4MPC: Actor-Critic Reinforcement Learning for Guiding Model Predictive Control"*

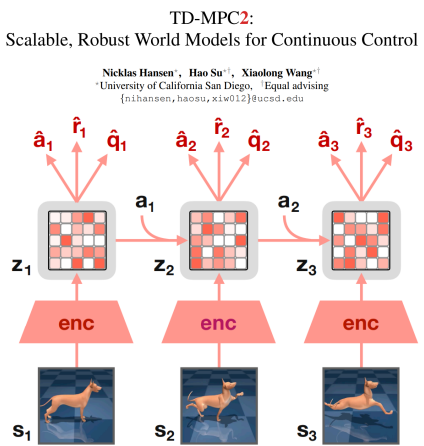
TD-MPC(2): Example of Algorithmic + Integrated

Idea:

- MPC in latent space (MPPI)
- Learned: Encoder, Decoder, Model, Policy, Latent Value Function
- Model learned only via reward decoding - no reconstruction loss
- Learns policy in latent space used as initializer in deployment

Architectures:

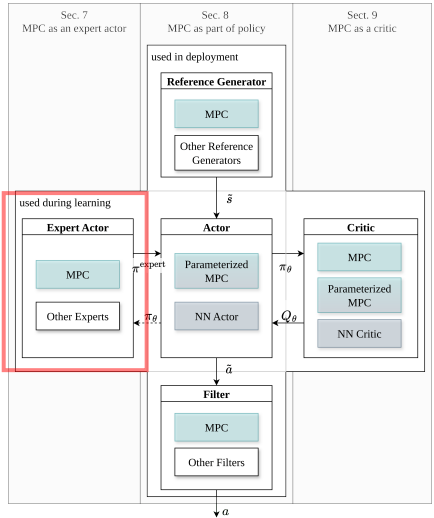
- Initializing policy: algorithmic
- Learned MPC components: integrated



[Image: Hansen et al. 2023]

Outline

MPC+RL: Taxonomy of Combination Approaches



MPC as an Expert Actor

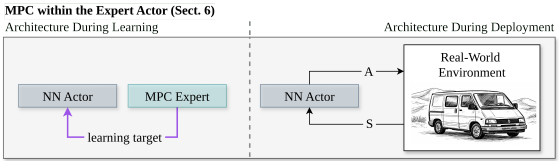
MPC used to generate training data for RL policy

Characteristics:

- MPC generates state-action pairs
- RL policy learns to mimic MPC behavior
- Can use behavioral cloning or DAgger
- See also [Karg and Lucia 2020]

Properties:

- ✓ Faster inference than MPC
- ✓ Leverages domain knowledge
- ✓ Sample-efficient learning
- ✗ Limited to MPC performance



Examples:

- Drone acrobatics [a]
- Autonomous driving [b]
- Robot manipulation [c]

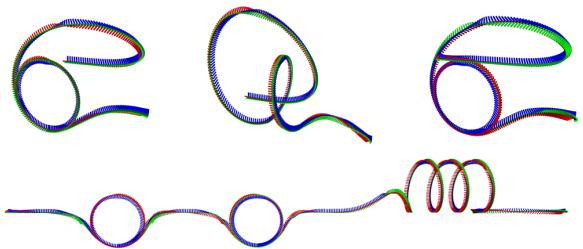
[a] **Kaufmann et al.**(2020), *"Deep Drone Acrobatics"*
[b] **Hoffmann et al.**(2024), *"PlanNetX: Learning an efficient neural network planner from MPC for longitudinal control"*
[c] **Mamedov et al.**(2024), *"Safe Imitation Learning of Nonlinear Model Predictive Control for Flexible Robots"*

MPC as an Expert Actor: Example

Deep Drone Acrobatics

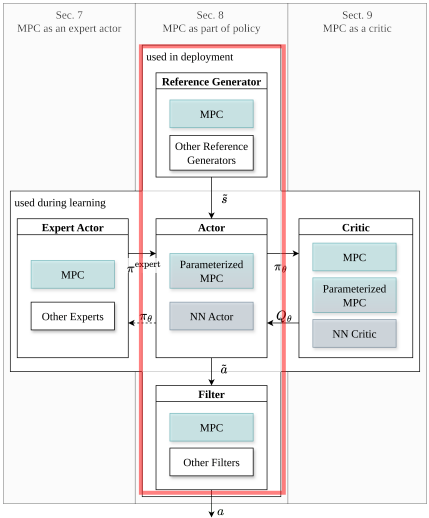
Elia Kaufmann*‡, Antonio Loquercio*‡, René Ranftl†, Matthias Müller†, Vladlen Koltun†, Davide Scaramuzza‡

- Use a privileged MPC expert actor
- At deployment visual abstractions only (no state)
- Zero-shot transfer



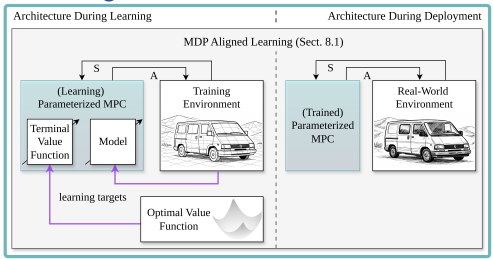
[Image: Kaufmann et al., RSS 2020,]

Taxonomy

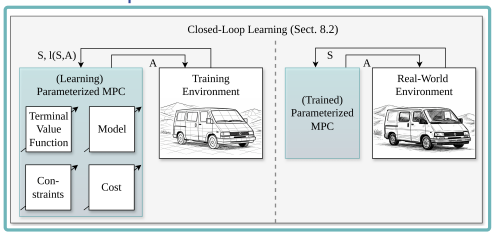

Sec. 7
MPC as an expert actor
Sec. 8
MPC as part of policy
Sect. 9
MPC as a critic
used in deployment
Reference Generator
MPC
Other Reference Generators
$\tilde{s}$
used during learning
Expert Actor
MPC
Other Experts
π^{expert}
π_θ
Actor
Parameterized MPC
NN Actor
π_θ
Q_θ
Critic
MPC
Parameterized MPC
NN Critic
$\tilde{a}$
Filter
MPC
Other Filters
a

MPC within the Deployed Policy

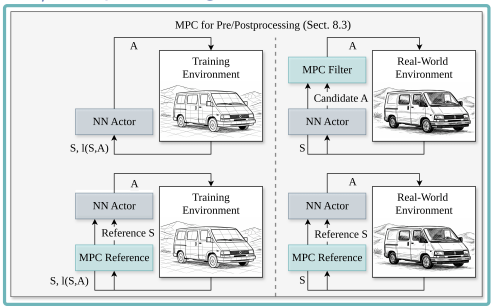
MDP Aligned



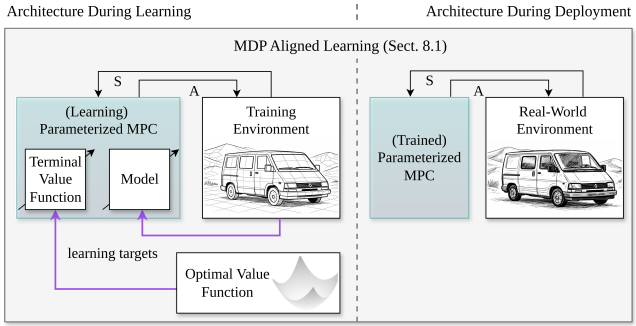
Closed-Loop



Pre/Postprocessing



MDP Aligned Learning

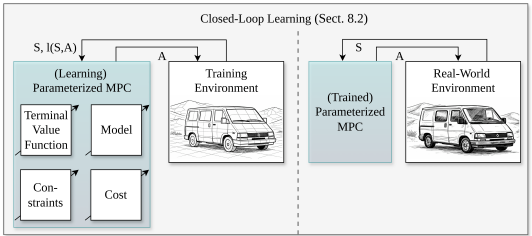


Paradigm: Learn individual components of the MPC controller to approximate the true environment (MDP). This aligns the controller's internal understanding with reality.

Components aligned with MDP:

- Approximate the system's dynamics by learning: $f_\theta^{\mathrm{MPC}} \approx P$
- Approximate the infinite-horizon cost with RL: $\bar{V}_\theta^{\mathrm{MPC}} \approx V^\star$

Closed-Loop Learning



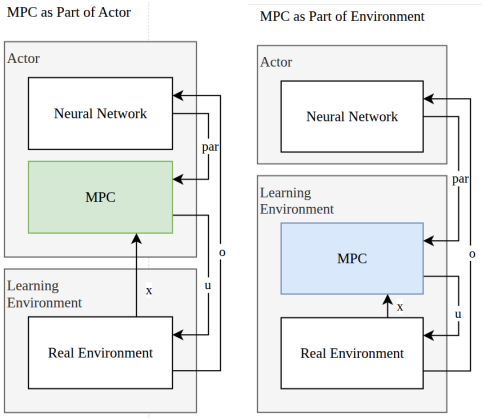
Paradigm:

- The focus is on what works best, not what is most accurate. The internal model or costs may become misaligned.

How is it implemented?

- Learn MPC parameters ϕ to directly optimize the final closed-loop performance $J(\phi)$.
- MPC as part of the Actor: Differentiable MPC as a layer in a policy network.
- MPC as part of the Environment: An outer-loop RL agent learns to choose the best MPC parameters ϕ as its "action", treating the MPC-controlled system as a black box.

Differentiate Through MPC?



A Hierarchical Approach for Strategic Motion Planning in Autonomous Racing

Rudolf Reiter[1], Jasper Hoffmann[2], Joschka Boedecker[2] and Moritz Diehl[1,3]

Actor-Critic Model Predictive Control: Differentiable Optimization meets Reinforcement Learning for Agile Flight

Angel Romero, Elie Aljalbout, Yunlong Song, Davide Scaramuzza

Comparison: Aligned Learning and Closed-Loop Learning

Aligned Learning

Approximate the true Markov Decision Process

Model Learning: Supervised learning on transition data, e.g.,

$$\min_{\theta} \mathcal{L}(S^+ - f_\theta^{\mathrm{MPC}}(S, A))$$

Terminal Value Learning: RL-style updates towards

$$V_\theta^{\mathrm{MPC}} \approx V^\star$$

Tradeoffs:

- ✓ Interpretable model, cost, constraints
- ✓ Division of design into manageable subtasks
- ✓ Safe, generalizes better
- ✗ Sub-optimal closed-loop performance
- ✗ May require complex models

Closed-Loop Learning

Approximate the optimal policy

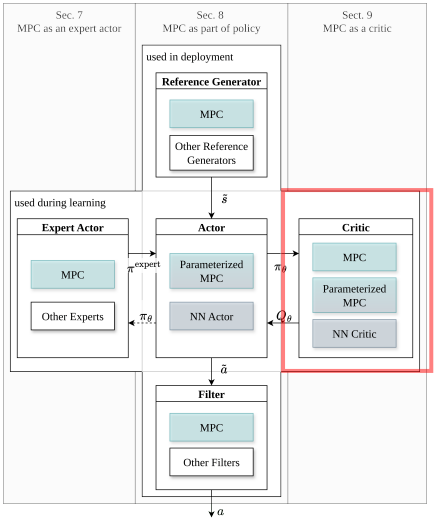
Policy Learning: Directly optimize closed-loop performance, towards

$$\mu_\theta^{\mathrm{MPC}} \approx \mu^\star$$

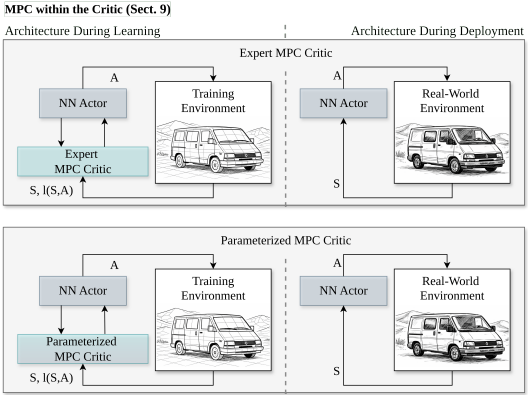
Tradeoffs:

- ✓ Direct optimization of actual objective
- ✓ Can correct for model errors through learning
- ✓ Flexibility in model choice
- ✗ Limited generalization to new scenarios
- ✗ Not representing true system
- ✗ Reduced adaptability when requirements change

Taxonomy


Sec. 7
MPC as an expert actor
Sec. 8
MPC as part of policy
Sect. 9
MPC as a critic
used in deployment
Reference Generator
MPC
Other Reference Generators
$\tilde{s}$
used during learning
Expert Actor
MPC
Other Experts
π^{expert}
π_θ
Actor
Parameterized MPC
NN Actor
π_θ
Q_θ
Critic
MPC
Parameterized MPC
NN Critic
$\tilde{a}$
Filter
MPC
Other Filters
a

MPC as the Critic



$$
\begin{aligned}
Q^{\mathrm{MPC}}(s,a) = \min_{z} \quad & \sum_{k=0}^{N-1} l^{\mathrm{MPC}}(x_k, u_k) + \bar{V}^{\mathrm{MPC}}(x_N) \\
\text{s.t.} \quad & x_0 = s, \\
& u_0 = a, \\
& x_{k+1} = f^{\mathrm{MPC}}(x_k, u_k),\ 0 \le k < N, \\
& 0 \le h^{\mathrm{MPC}}(x_k, u_k),\ 1 \le k < N, \\
& 0 \le h_N^{\mathrm{MPC}}(x_N).
\end{aligned}
$$

- We can use MPC to derive a Q-function Q^{MPC}, which can be used as a critic!
- We constrain the initial control u_0 to be a.

Example: Learnable Critic

Instead of using a fixed MPC critic, we can further learn to improve the critic:

- In [Bhardwaj et al. 2020], the author combine sampling-based MPC with a learned value function using a parallel architecture solving multiple robot manipulation tasks. A simplified version of the critic is given by:

$$Q(s,a) = (1-\lambda)\ \underbrace{Q_w(s,a)}_{\text{learned critic}} + \lambda\ \underbrace{Q^{\text{MPC}}(s,a)}_{\text{model-based}},$$

  where Q_w is a learned critic and Q^{MPC} is the MPC-based Q-function.
- The parameter λ balances the two Q-functions and is decayed over time.
- (Remember: This was the parallel architecture example)

Example: Learnable Critic from parameterized MPC

- [Anand et al. 2023] proposed a simple actor-critic scheme, where a single MPC scheme is used for both actor and critic.
- **Idea:** Close-to-optimal MPC parameters ϕ allow approximation of the optimal Q-function $Q^\star$.

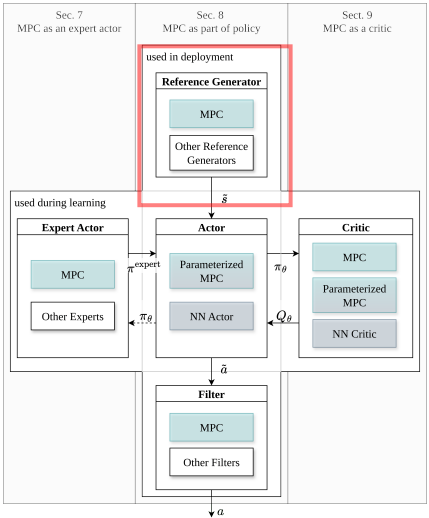
Local Q-function approximation:

$$Q_w(s,a) = Q_\phi^{\mathrm{MPC}}(s,a) + \nabla_\phi Q_\phi^{\mathrm{MPC}}(s,a)^\top w$$

MPC Q-Function

$$\begin{aligned} Q_\phi^{\mathrm{MPC}}(s,a) = \min_z \quad & \sum_{k=0}^{N-1} l_\phi^{\mathrm{MPC}}(x_k,u_k) + \bar{V}_\phi^{\mathrm{MPC}}(x_N) \\ \text{s.t.} \quad & x_0 = s, \\ & u_0 = a, \\ & x_{k+1} = f_\phi^{\mathrm{MPC}}(x_k,u_k), \ , \\ & 0 \le h_\phi^{\mathrm{MPC}}(x_k,u_k), , \\ & 0 \le h_{N_\phi}^{\mathrm{MPC}}(x_N). \end{aligned}$$

Taxonomy


Sec. 7
MPC as an expert actor
Sec. 8
MPC as part of policy
Sect. 9
MPC as a critic
used in deployment
Reference Generator
MPC
Other Reference Generators
$\tilde{s}$
used during learning
Expert Actor
MPC
Other Experts
π^{expert}
π_θ
Actor
Parameterized MPC
NN Actor
π_θ
Q_θ
Critic
MPC
Parameterized MPC
NN Critic
$\tilde{a}$
Filter
MPC
Other Filters
a

MPC for Preprocessing

MPC used in deployed policy, but not during training

Preprocessing: Reference Generator

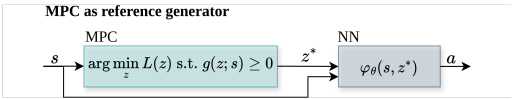
Purpose: Provide reference trajectory for RL policy

Characteristics:

- MPC outputs planned trajectory
- RL policy uses reference as input

Benefit:

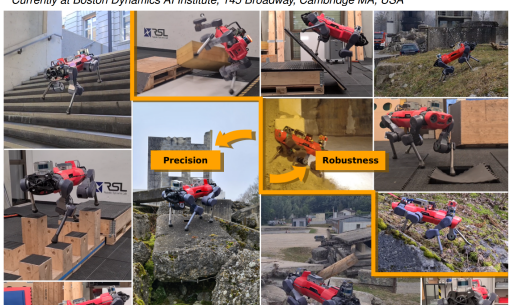
- ✓ Interpretable reference trajectory
- ✓ Bandwidth and learning at lower level



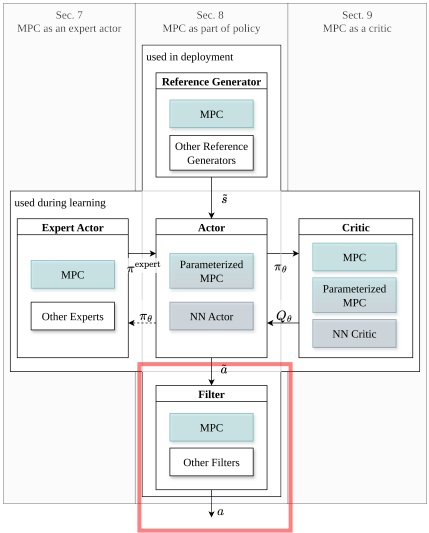
DTC: Deep Tracking Control

Fabian Jenelten,[1*] Junzhe He,[1] Farbod Farshidian,[2] and Marco Hutter[1]

[1] *Robotic Systems Lab, ETH Zurich, 8092 Zurich, Switzerland.*
[2] *Currently at Boston Dynamics AI Institute, 145 Broadway, Cambridge MA, USA*



Taxonomy


Sec. 7
MPC as an expert actor
Sec. 8
MPC as part of policy
Sect. 9
MPC as a critic
used in deployment
Reference Generator
MPC
Other Reference Generators
$\tilde{s}$
used during learning
Expert Actor
MPC
Other Experts
π^{expert}
π_θ
Actor
Parameterized MPC
NN Actor
π_θ
Q_θ
Critic
MPC
Parameterized MPC
NN Critic
a
Filter
MPC
Other Filters
a

MPC for Postprocessing

MPC used in deployed policy, but not during training

Postprocessing: Safety Filter

Purpose: Ensure safety w.r.t. model and constraints

Characteristics:

- MPC filters RL policy output
- Projects actions to safe set

Benefit:

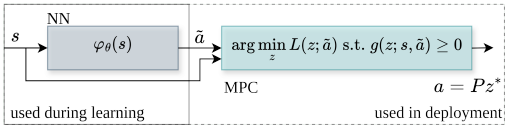
✓ Provides safety guarantees w.r.t. model and constraints

Issue:

✗ Policy unaware of filter during training

✗ May lead to suboptimal actions

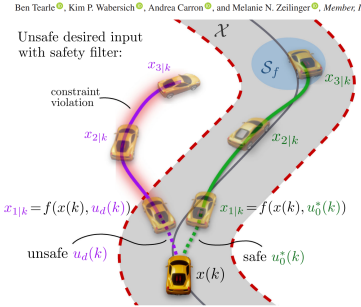
MPC for postprocessing



IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 6, NO. 4, OCTOBER 2021

A Predictive Safety Filter for Learning-Based Racing Control

Ben Tearle, Kim P. Wabersich, Andrea Carron, and Melanie N. Zeilinger, *Member, IEEE*



Outline

Next Steps and Current Challenges

- **Computational efficiency:** Scaling to massive parallelization
- **Algorithms:** Slow solving of complex problems (MPPI) vs. fast solving of simple optimization problems (SQP)
- **Expanding frameworks:** Generalization from MPC to broader planning (combinatorial optimization, mixed-integer, stochastic multi-stage programming)
- **Multi-agent systems:** Coordination, communication, and distributed decision-making largely unresolved
- **Applications:** Practical demonstrations beyond robotics (energy, healthcare, logistics)

Summary

Having heard this lecture, you should have a better understanding of...

- Identify strengths and weaknesses of MPC and RL
- Recognize potential benefits of combining both approaches
- MPC+NN: Distinguish between parameterized MPC architectures:
  - Integrated, hierarchical, parallel, parameterized
- MPC+RL Design MPC-RL algorithms based on:
  - MPC as expert actor
  - MPC within deployed policy
  - MPC as critic
  - MPC as reference generator
  - MPC as a safety filter

Thank you for your attention!