A Hierarchical Approach for Strategic Motion Planning in Autonomous Racing

Rudolf Reiter, Jasper Hoffmann, Joschka Boedecker and Moritz Diehl

Systems Control and Optimization Laboratory (syscop)

European Control Conference 2023
June 13, 2023

- Task: Strategic planning and control of autonomous race cars $\rightarrow$ blocking of other agents, efficient overtaking

- Idea 1: use optimization-based control: NMPC
- Problem: hard to define strategic decisions (bi-level problem)

- Idea 2: use reinforcement learning
- Problem: can hardly account for safety, many data needed for simple maneuvers

- Our idea: Combine reinforcement learning and NMPC hierarchically
- Questions: Improved performance over pure RL? Faster learning? Meaningful learning? Guaranteed safety?

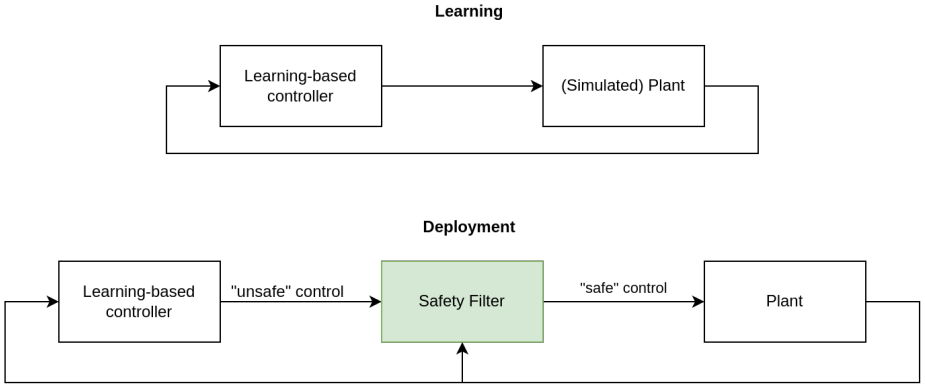
Outline

1. Relation to the safety filter
2. Proposed architecture
3. NMPC Formulation
4. RL Formulation
5. HILEPP Algorithm
6. Evaluation
7. Conclusion and Discussion

Relation to the safety filter[1]

Original

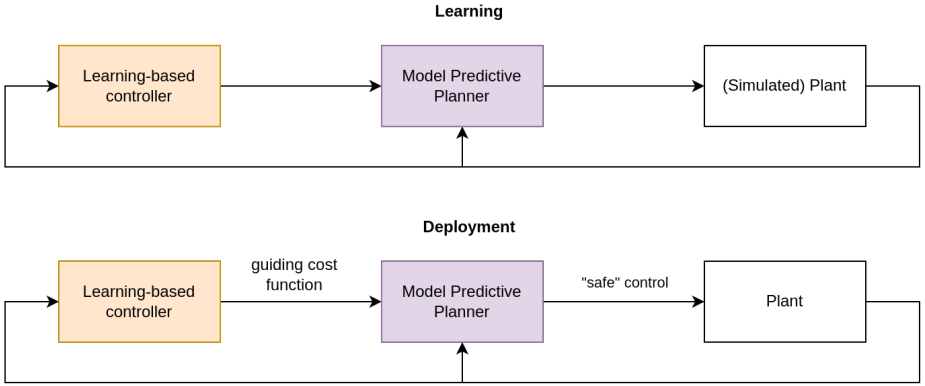
The safety filter uses an NLP to project controls onto safe sets



[1] Kim Peter Wabersich and Melanie N. Zeilinger. "A predictive safety filter for learning-based control of constrained nonlinear dynamical systems". In: *Automatica* 129 (2021), p. 109597. ISSN: 0005-1098. DOI: https://doi.org/10.1016/j.automatica.2021.109597.

Relation to the safety filter[2]

Our approach



[2]Wabersich and Zeilinger, "A predictive safety filter for learning-based control of constrained nonlinear dynamical systems".

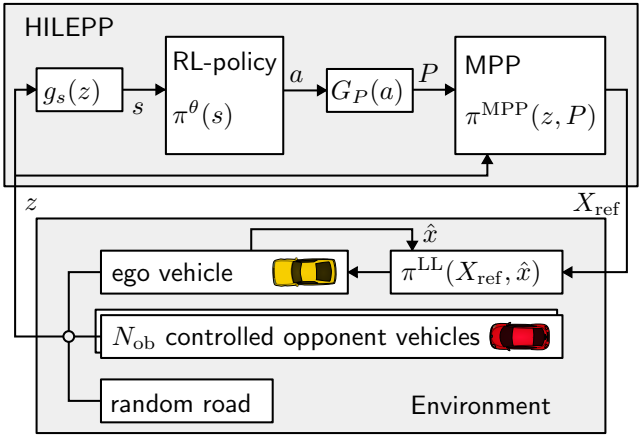
Safety Filter

$$
\begin{aligned}
\min_{X,U} \quad & \|u_0 - \bar{a}\|_R^2 \\
\text{s.t.} \quad & x_0 = \bar{x}_0, \quad x_N \in \mathcal{S}^{\text{t}} \\
& x_{i+1} = F(x_i, u_i), \quad i = 0, \dots, N-1 \\
& x_i \in \mathcal{X}, u_i \in \mathcal{U}, \quad i = 0, \dots, N-1
\end{aligned} \tag{1}
$$

HILEPP (ours)

$$
\begin{aligned}
\min_{X,U} \quad & L(X, U, a) \\
\text{s.t.} \quad & x_0 = \hat{x}_0, \quad x_N \in \mathcal{S}^{\text{t}} \\
& x_{i+1} = F(x_i, u_i), \quad i = 0, \dots, N-1 \\
& x_i \in \mathcal{X}, u_i \in \mathcal{U}, \quad i = 0, \dots, N-1,
\end{aligned} \tag{2}
$$

[3] Wabersich and Zeilinger, "A predictive safety filter for learning-based control of constrained nonlinear dynamical systems".



Invariance pre-conditioning function $g_s(z)$ sets inputs s to RL policy $a = \pi^{\Theta}(s)$. Function $G_P(a)$ transforms RL actions a to MPP parameters P. Policy $\pi^{\mathrm{MPP}}(z, P)$ solves NLP and outputs safe reference X^{ref}.

- MPP is a NMPC used as planner
- Kinematic vehicle model in Frenet coordinate frame. States $x^\top = [\zeta, n, \alpha, v, \delta]$
- Obstacle avoidance with ellipses - circles[4]
- Obstacle prediction in two modes (Defined according to racing rules):
  - *Follower: generously* assuming straight linear motion in Frenet coordinate frame
  - *Leader: evasively* allowing only decelerating linear motion
- Cost parameterization through RL actions:

$$G_P(a) : a \rightarrow \Big(\xi_{\mathrm{ref},0}(a), \ldots, \xi_{\mathrm{ref},N}(a), Q_{\mathrm{w}}(a)\Big) \tag{3}$$

$$\xi_{\mathrm{ref},k}(a) = [0 \quad n \quad 0 \quad v_x \quad 0]^\top \in \mathbb{R}^{n_x} \tag{4}$$

$$Q_{\mathrm{w}}(a) = \mathrm{diag}([0 \quad w_n \quad 0 \quad w_v \quad 0]) \tag{5}$$

[4] Rudolf Reiter et al. *Frenet-Cartesian Model Representations for Automotive Obstacle Avoidance within Nonlinear MPC*. 2023. arXiv: 2212.13115 [eess.SY].

Cost parameterization through RL actions:

$$G_P(a) : a \to \left(\xi_{\mathrm{ref},0}(a), \ldots, \xi_{\mathrm{ref},N}(a), Q_{\mathrm{w}}(a)\right) \tag{6}$$

$$\xi_{\mathrm{ref},k}(a) = [0 \quad n \quad 0 \quad v_x \quad 0]^\top \in \mathcal{R}^{n_x} \tag{7}$$

$$Q_{\mathrm{w}}(a) = \mathrm{diag}([0 \quad w_n \quad 0 \quad w_v \quad 0]) \tag{8}$$

NMPC (MPP) parameterized cost:

$$\begin{aligned} L(X, U, a, \Xi) = \sum_{k=0}^{N-1} \|x_k - \xi_{\mathrm{ref},k}(a)\|^2_{Q_{\mathrm{w}}(a)} + \|u_k\|^2_R \\ + \|x_N - \xi_{\mathrm{ref},N}(a)\|^2_{Q^t} + \sum_{k=0}^{N} \|\sigma_k\|^2_{Q_{\sigma,2}} + |q^\top_{\sigma,1}\sigma_k|. \end{aligned} \tag{9}$$

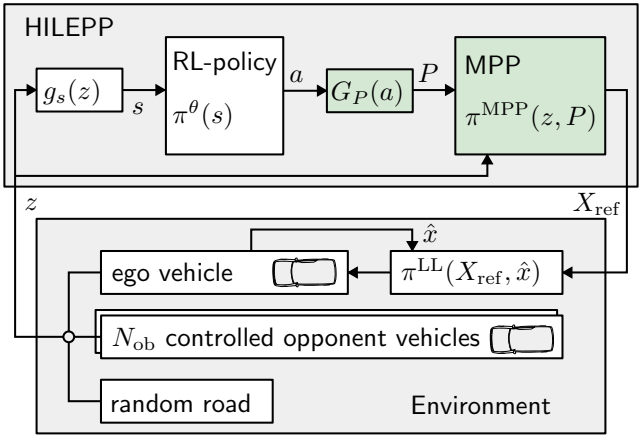
We compare two action vectors (with or without setting weights):

- HILEPP-I: $a_{\mathrm{I}} := [n, v_x]^\top$
- HILEPP-II: $a_{\mathrm{II}} := [n, v_x, w_n, w_v]^\top$

The NLP that is solved for each MPP iteration, can be written as:

$$
\begin{aligned}
\min_{X,U,\Xi} \quad & L(X,U,a,\Xi) \\
\text{s.t.} \quad & x_0 = \hat{x}, \quad \Xi \geq 0, \quad x_N \in \mathcal{S}^{\mathrm{t}} \\
& x_{i+1} = F(x_i, u_i) && i = 0,\dots,N-1 \\
& U_i \in B_u, && i = 0,\dots,N-1 \\
& x_i \in B_x(\sigma_k) \cap B_{\mathrm{lat}}(\sigma_k) && i = 0,\dots,N \\
& x_i \in B_{\mathrm{ob}}(p_i^{\mathrm{ob},j}, \Sigma_i^{\mathrm{ob},j}, \sigma_k) && i = 0,\dots,N \\
& && j = 0,\dots,N_{\mathrm{ob}},
\end{aligned}
\tag{10}
$$

using states X, controls U, slacks Ξ, dynamic integration function $F(\cdot)$, state and acceleration constraints $B_x(\cdot), B_{\mathrm{lat}}(\cdot)$ and obstacle constraints $B_{\mathrm{ob}}(p_i^{\mathrm{ob},j}, \Sigma_i^{\mathrm{ob},j}, \sigma_k)$, depending on prediction $p_i^{\mathrm{ob},j}, \Sigma_i^{\mathrm{ob},j}$ for each obstacle.


HILEPP
$g_s(z)$
s
RL-policy
$\pi^{\theta}(s)$
a
$G_P(a)$
P
MPP
$\pi^{\mathrm{MPP}}(z, P)$
z
X_{ref}
$\hat{x}$
ego vehicle
$\pi^{\mathrm{LL}}(X_{\mathrm{ref}}, \hat{x})$
N_{ob} controlled opponent vehicles
random road
Environment

Reinforcement Learning

General

- Markov assumption, state space $\mathcal{S}$, action space $\mathcal{A}$, looking for policy $\pi^\theta : \mathcal{S} \mapsto \mathcal{A}$, reward function $R : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$
- We use *actor critic policy gradient* algorithm[5] with actor π^θ and a critic Q^ϕ

Specific

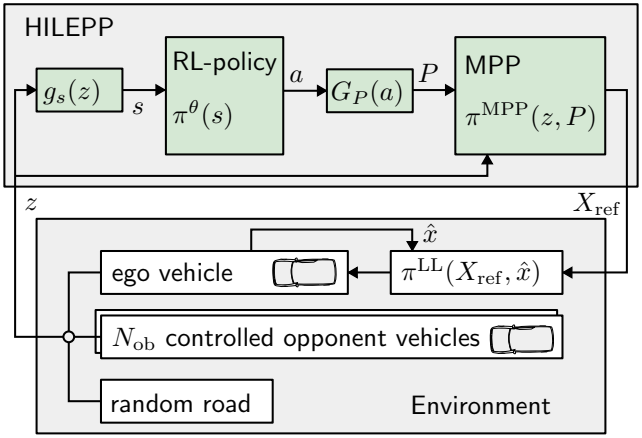
- Pre-processing function from ego state $s = [n, v, \alpha]^\top$, road curvature evaluations $\kappa(\cdot)$ and obstacle states z to (partly) invariant RL states $s_{\text{ob}_i} = [\zeta_{\text{ob}_i} - \zeta, n_{\text{ob}_i}, v_{\text{ob}_i}, \alpha_{\text{ob}_i}]^\top$

$$s_k = g_s(z_k) = [\kappa(\zeta + d_i), \dots, \kappa(\zeta + d_N), s^\top, s_{\text{ob}_1}^\top, \dots, s_{\text{ob}_N}^\top]^\top \tag{11}$$

- We use the reward for center line speed $\dot{s}$ and the total rank, with

$$R(s, a) = \frac{\dot{s}}{200} + \sum_{i=1}^{N_{\text{ob}}} 1_{\zeta_k > \zeta_k^{\text{ob}_i}} \tag{12}$$

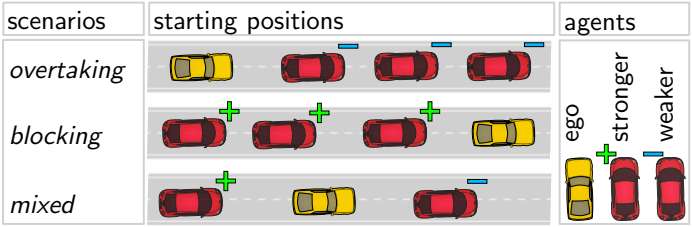
[5] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.


HILEPP
$g_s(z)$
s
RL-policy
$\pi^\theta(s)$
a
$G_P(a)$
P
MPP
$\pi^{\mathrm{MPP}}(z, P)$
z
X_{ref}
$\hat{x}$
ego vehicle
$\pi^{\mathrm{LL}}(X_{\mathrm{ref}}, \hat{x})$
N_{ob} controlled opponent vehicles
random road
Environment

Evaluation

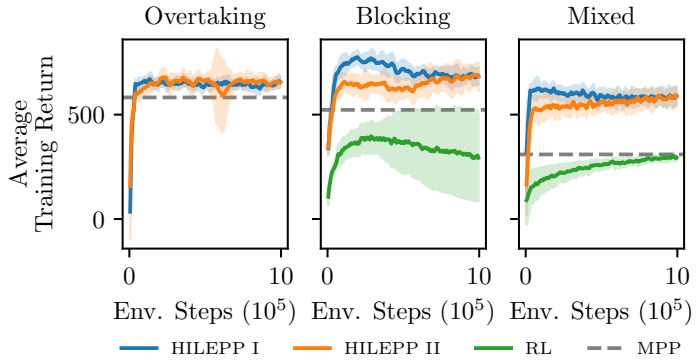
Setup

- Training of $\sim 10^6$ steps in randomized simulated scenarios
- Only the ego agent is trained, opponents only use MPP
- Three different scenario types

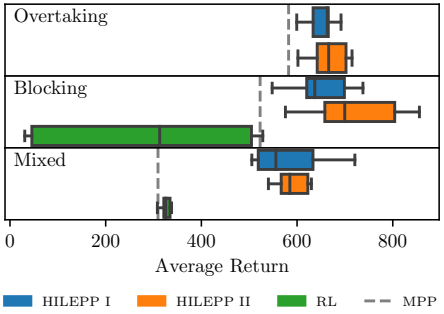


- Comparison of
  - MPP
  - RL
  - HILEPP-I (only reference states)
  - HILEPP-II (reference states and weights)

- pure RL learns slow
- HILEPP very sample efficient
- HILEPP-I learns quicker than HILEPP-II



- pure RL *struggled* to keep up even with MPP
- overtaking does not require much strategy → MPP compared to HILEPP smaller
- HILEPP-II performs better than HILEPP-I



Module	Mean± Std.	Max
MPP	5.45 ± 2.73	8.62
RL policy	0.13 ± 0.01	0.26
HILEPP-I	6.90 ± 3.17	9.56
HILEPP-II	7.41 ± 2.28	9.21

Table: Computation times (ms) of modules.

Evaluation

Examples

- →Play-scenario-blocking
- →Play-scenario-mixed
- →Play-scenario-overtake

Conclusion and Discussion

Questions from the beginning:

- Our idea: Hierarchically combine NMPC and RL
- Questions:´
  Faster learning? ✓(Sample efficiency with HILEPP)
  Meaningful learning? ✓(Outperforming baselines by +100% rewards)
  Guaranteed safety? ✓(w.r.t. NMPC constraints)

Conclusion:

- We can learn strategic decisions
- The final planning times are real-time feasible
- Solutions are safe

Not considered and future work:

- *sim-to-real gap*: Testing the trained policy in real-world scenarios
- Training of other agents (*multi-agent RL*)

Thank you for your attention!

universität freiburg