Deep Reinforcement Learning for Controls – Application to the Path Following Problem

Johannes Ultsch, Jonathan Brembeck, Ricardo de Castro
German Aerospace Center (DLR) - Institute of System Dynamics and Control (SR)

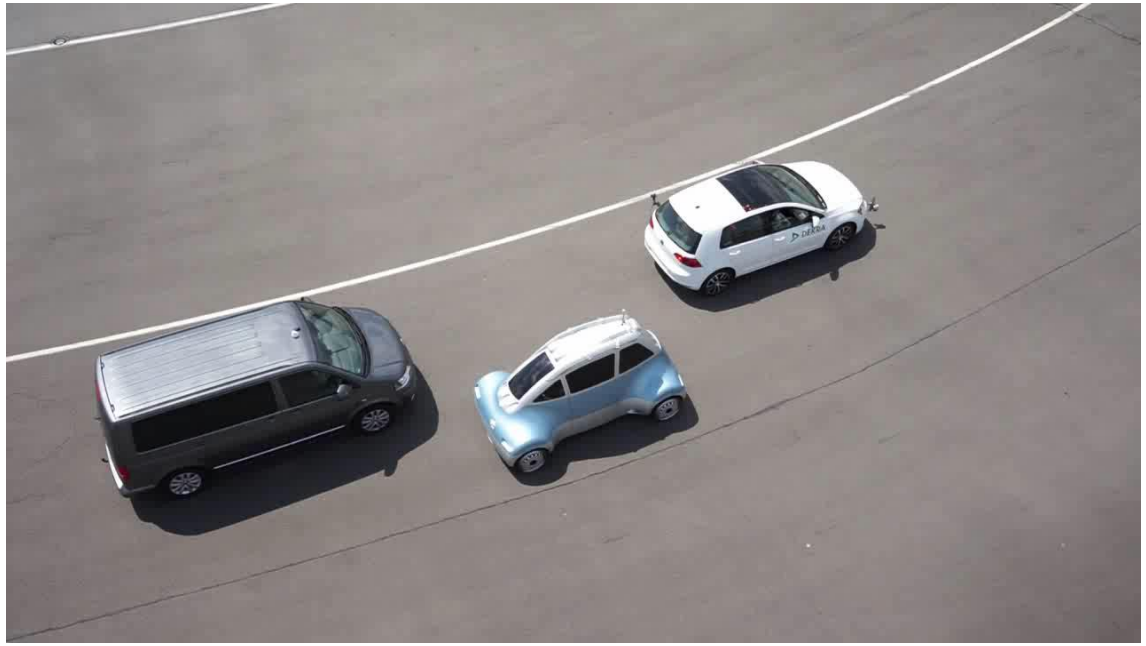
14th ESA Workshop on
Avionics, Data, Control and Software Systems
20-22 October 2020

DLR

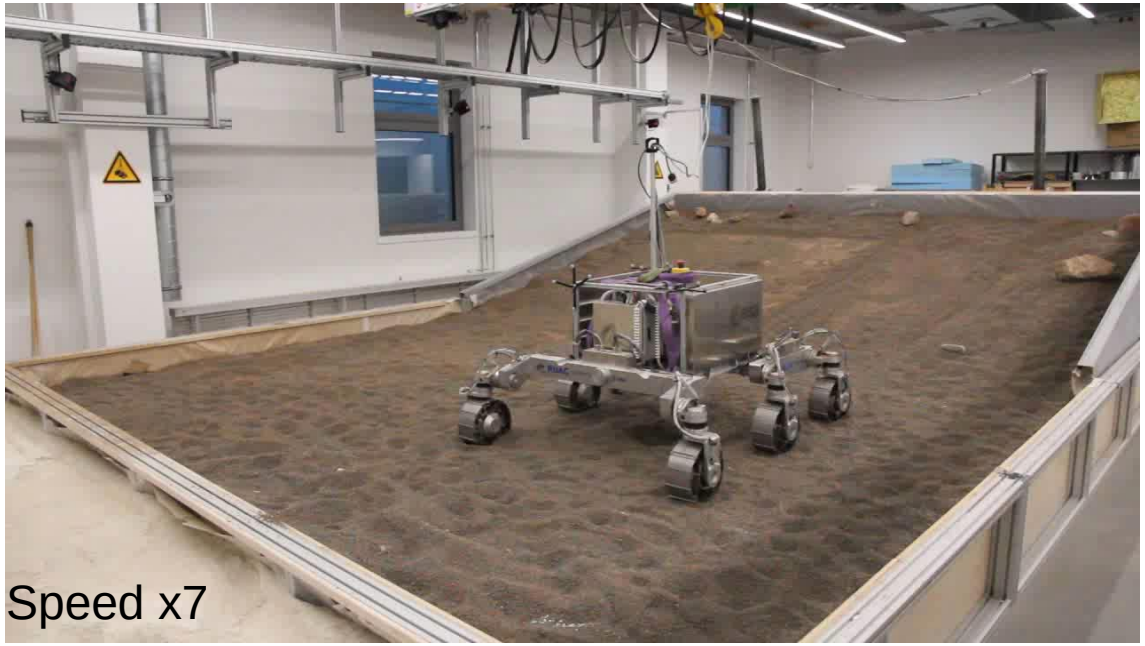
The ROboMObil - Synergies: Transport & Space

Synergies of operation modes

ROboMObil



ExoMars rover

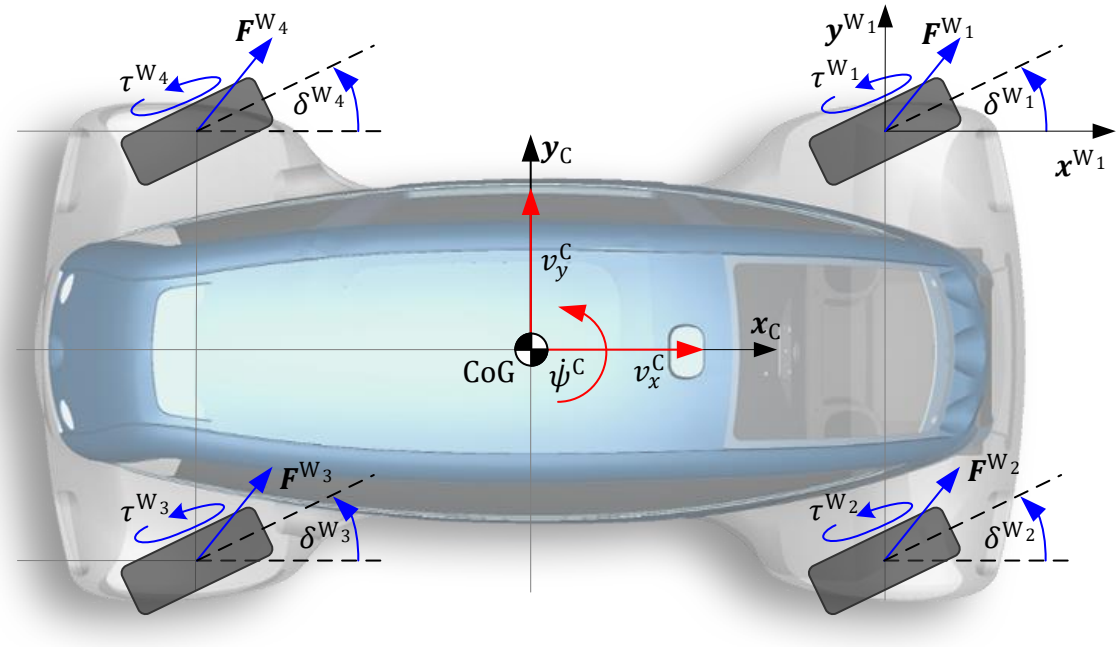


The ROboMObil - The Robotic Electric Vehicle

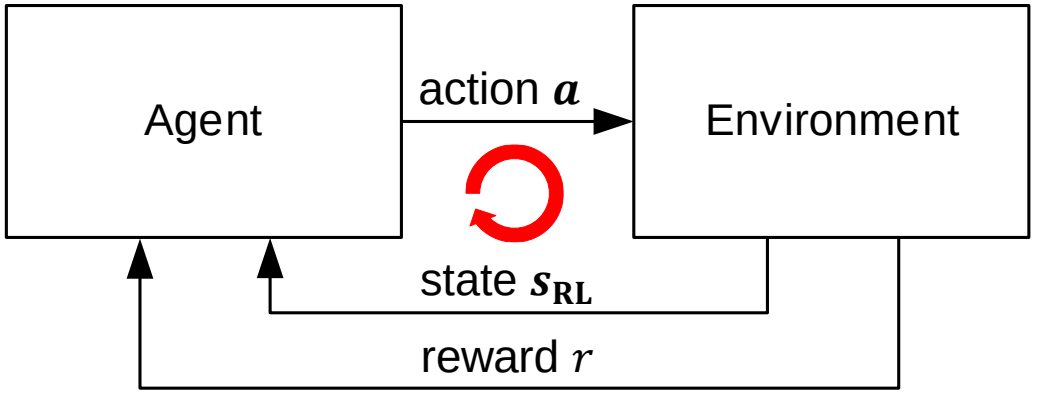
- Electric X-by-Wire vehicle
- 13 kWh Li-ion battery @ 350V

- 4 Steer-by-wire actuators
- 4 In-wheel traction motors
- 4 Semi-active dampers
- 2 Electro-hydraulic brake-by-wire actuators for friction brake

- <u>Control inputs:</u>
  - Desired steering angle: $\delta^{W_1}, \delta^{W_2}, \delta^{W_3}, \delta^{W_4}$
  - Desired traction torque: $\tau^{W_1}, \tau^{W_2}, \tau^{W_3}, \tau^{W_4}$



The Reinforcement Learning (RL) Approach



- <u>Goal:</u>

Find a policy $\pi(\boldsymbol{a}_k|\boldsymbol{s}_{\mathbf{RL},k})$, which maximizes the *expected discounted return* G_k:

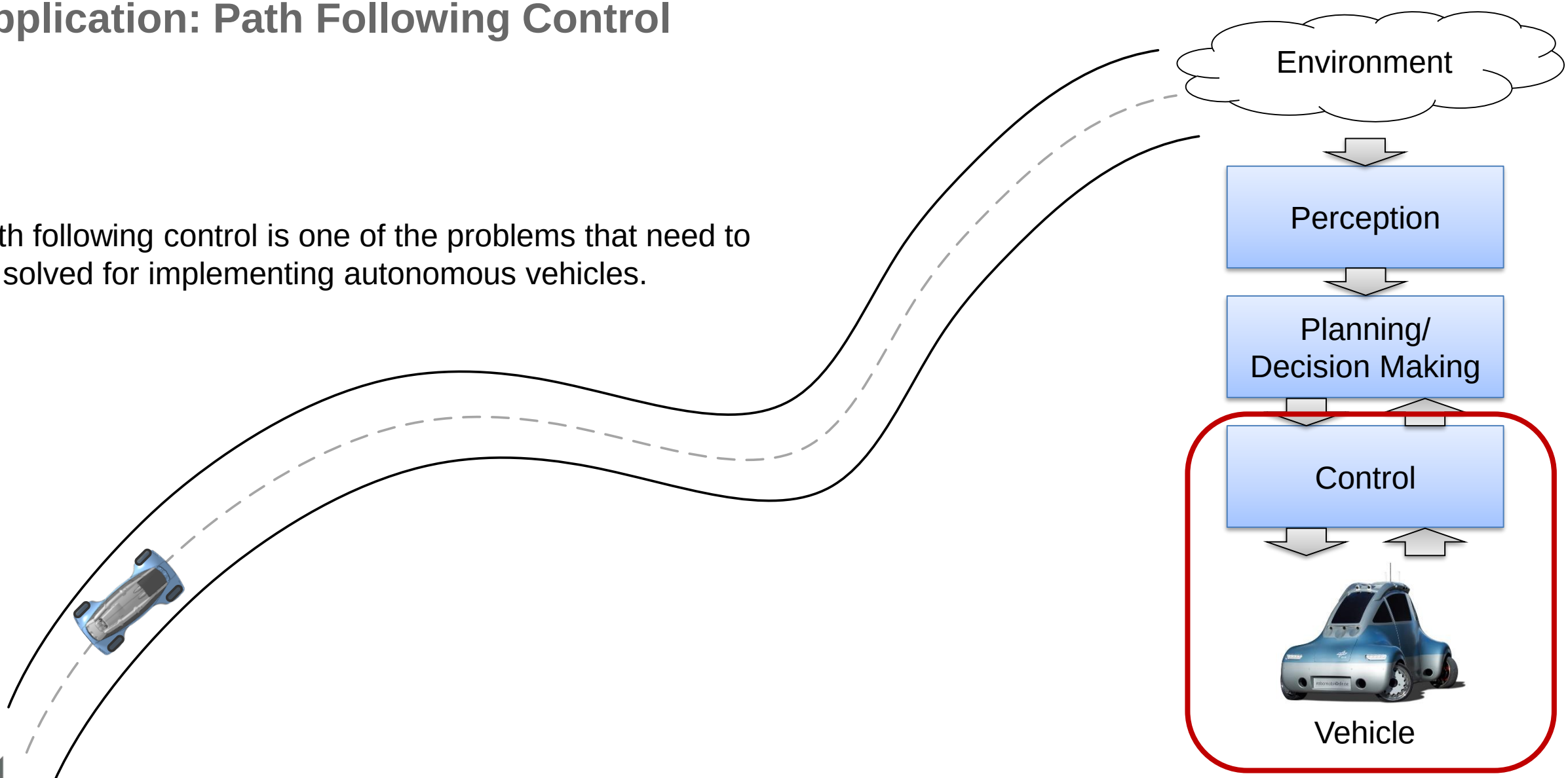
$$G_k = r_{k+1} + \gamma\, r_{k+2} + \gamma^2\, r_{k+3} + \ \ldots = \sum_{i=0}^{\infty} \gamma^i\, r_{k+i+1} \ \text{ mit } \gamma \in [0,1]$$

- The policy is adapted based on interaction with the system (*training*)
  - **Offline** training (simulation): safe, fast and scalable
  - **Online** training (real world): no reality gap

- Central design degrees of freedom
  - Selection of the **reward function** r which encodes the control goal
  - Selection of **RL algorithm** and hyperparameters
  - Selection of the **feedback state** $\boldsymbol{s}_{\mathbf{RL}}$

Application: Path Following Control

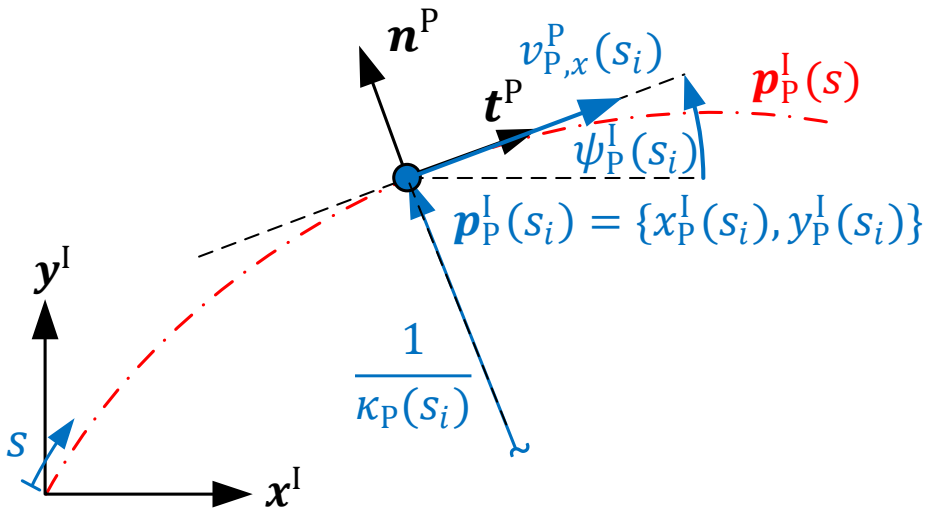
Path following control is one of the problems that need to be solved for implementing autonomous vehicles.



Motion Demand Representation

- Control goal: The vehicle should follow a predetermined motion demand $\boldsymbol{\lambda}$ as close as possible
- The motion demand is parametrized by the arc length s:

$$\boldsymbol{\lambda}(s) = \left[x_{\mathrm{P}}^{\mathrm{I}}(s), y_{\mathrm{P}}^{\mathrm{I}}(s), \psi_{\mathrm{P}}(s), \kappa_{\mathrm{P}}(s), v_{\mathrm{P},x}^{\mathrm{P}}(s)\right]^{\mathrm{T}}$$



- In contrast to trajectory tracking control there is no explicit time dependency in the motion demand related to path following control

The Path Following Problem

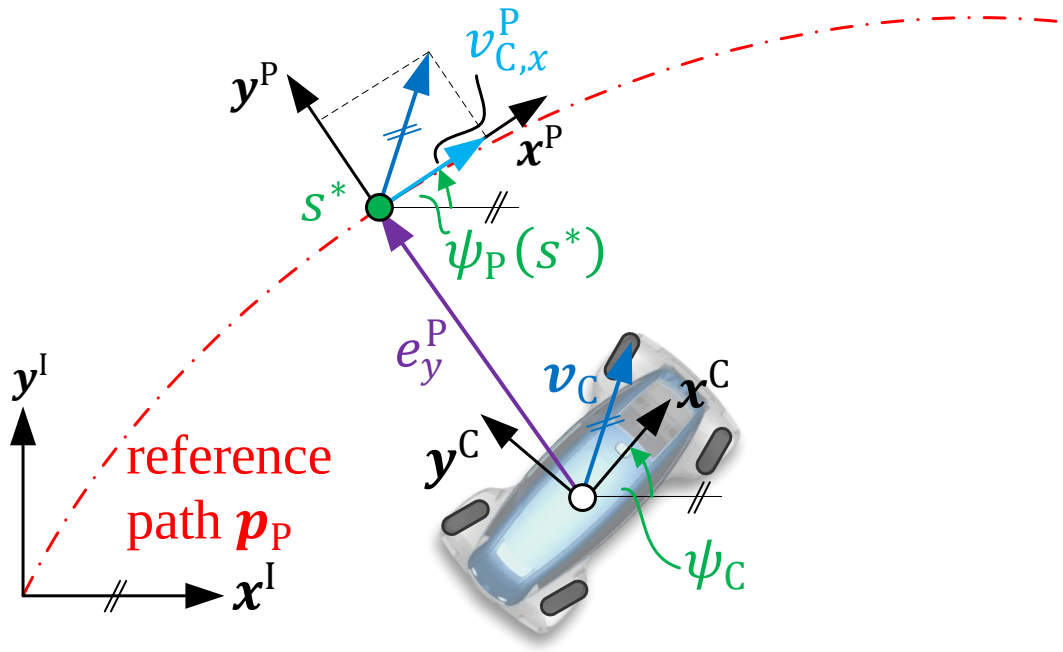
- The reference point s^* on the path is defined as:

$$s^* = \arg\min_s \| \underbrace{\boldsymbol{p}_\mathrm{P}(s) - \boldsymbol{p}_\mathrm{C}}_{\boldsymbol{e}(s)} \|_2$$

- We use the time independent path interpolation from [1] to efficiently calculate the reference point on the path
- <u>Control goal:</u>

1. $\min e_y^\mathrm{P}$ with the position error: $e_y^\mathrm{P} = \underbrace{y_\mathrm{P}^\mathrm{P}}_{=0} - y_\mathrm{C}^\mathrm{P}$

2. a) $\min e_{v_x}^\mathrm{P}$ with the velocity error: $e_{v_x}^\mathrm{P} = v_{\mathrm{P},x}^\mathrm{P} - v_{\mathrm{C},x}^\mathrm{P}$
   b) $\min e_\psi$ with the orientation error: $e_\psi = \psi_\mathrm{P} - \psi_\mathrm{C}$



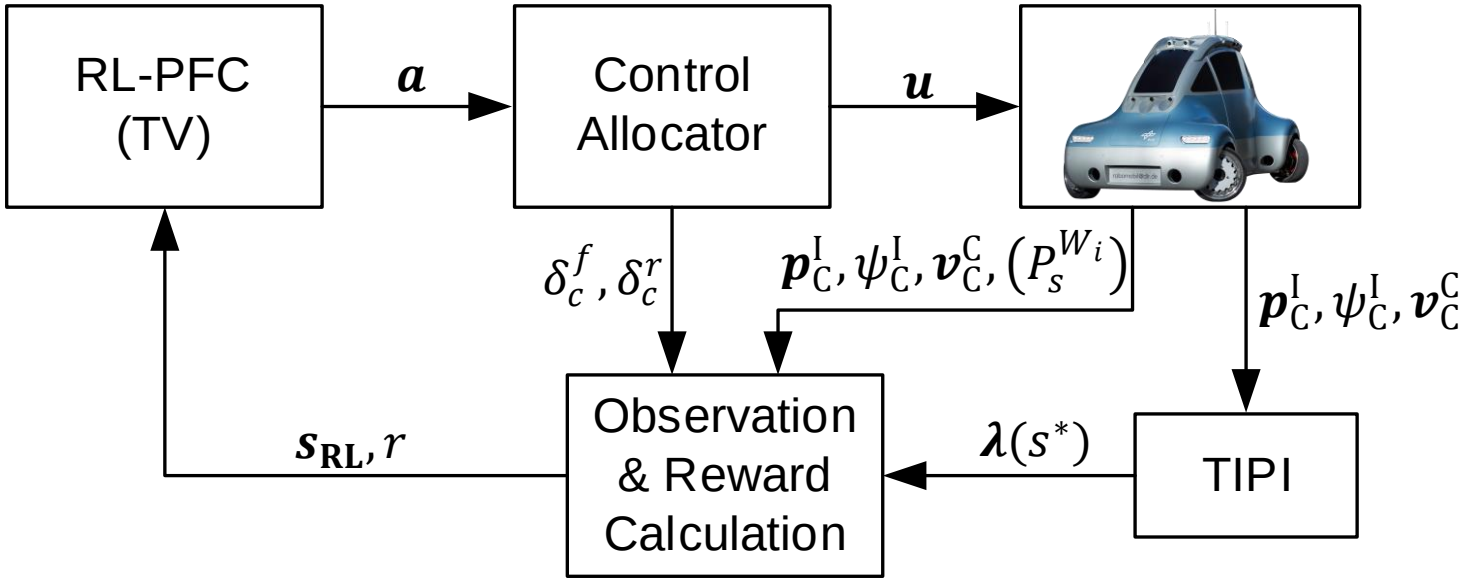
[1] P. Ritzer, C. Winter and J. Brembeck, „Advanced path following control of an overactuated robotic vehicle,“ in Proceedings of the IEEE Intelligent Vehicles Symposium, 2015.

Application to the ROboMObil Architecture [2]

- RL-PFC:
  - $\boldsymbol{a} = [\delta^{\mathrm{f}}, \delta^{\mathrm{r}}, \tau]$
  - with $\tau = \tau^{\mathrm{W}_1} = \tau^{\mathrm{W}_2} = \tau^{\mathrm{W}_3} = \tau^{\mathrm{W}_4}$
- The control allocator distributes $(\delta^{\mathrm{f}}, \delta^{\mathrm{r}})$ according to Ackermann to all 4 wheels

  RL-PFC TV:
  - Torque vectoring (TV)
  - $\boldsymbol{a} = [\delta^{\mathrm{f}}, \delta^{\mathrm{r}}, \tau^{\mathrm{W}_1}, \tau^{\mathrm{W}_2}, \tau^{\mathrm{W}_3}, \tau^{\mathrm{W}_4}]$

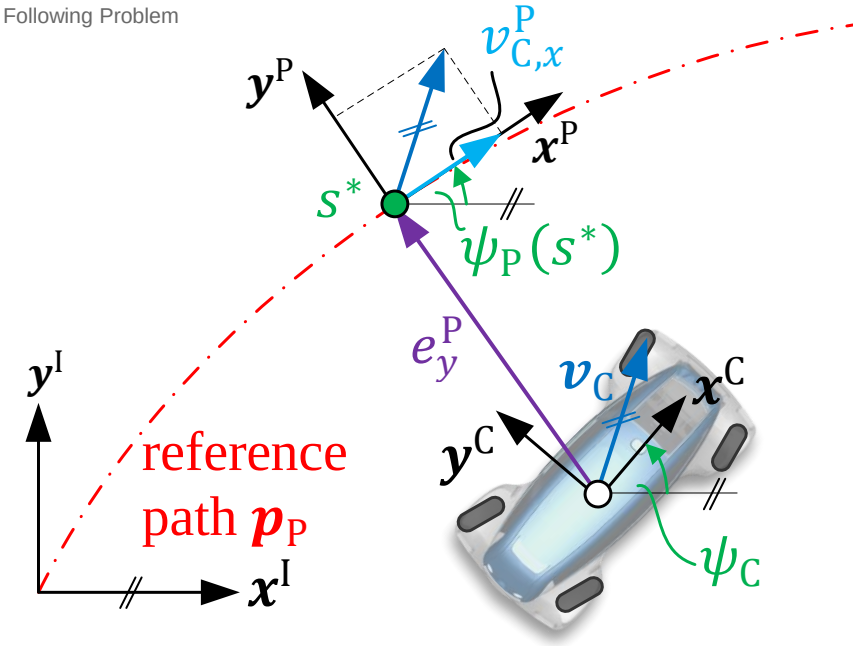
$$\boldsymbol{u} = [\delta^{\mathrm{W}_1}, \delta^{\mathrm{W}_2}, \delta^{\mathrm{W}_3}, \delta^{\mathrm{W}_4}, \tau^{\mathrm{W}_1}, \tau^{\mathrm{W}_2}, \tau^{\mathrm{W}_3}, \tau^{\mathrm{W}_4}]$$



- $\boldsymbol{s}_{\mathbf{RL},k} = \Big[\underbrace{e_{y,k}^{\mathrm{P}}, e_{v_x,k}^{\mathrm{P}}, e_{v_y,k}^{\mathrm{P}}, e_{\psi,k}, e_{y,k-1}^{\mathrm{P}}, e_{v_x,k-1}^{\mathrm{P}}, e_{v_y,k-1}^{\mathrm{P}}, e_{\psi,k-1}}_{\text{position , velocity and orientation error}}, \underbrace{\kappa_{\mathrm{P},k}, \kappa_{\mathrm{P},k-1}}_{\text{curvature}}, \underbrace{\delta_{c,k}^{\mathrm{f}}, \delta_{c,k}^{\mathrm{r}}, \delta_{c,k-1}^{\mathrm{f}}, \delta_{c,k-1}^{\mathrm{r}}}_{\text{steering angles}}\Big]$ at timestep k and $k-1$

[2] J. Ultsch, J. Brembeck and R. de Castro, „Learning-Based Path Following Control for an Over-Actuated Robotic Vehicle,“ in VDI-AUTOREG, Düsseldorf, 2019.

Reward Function Design (1)

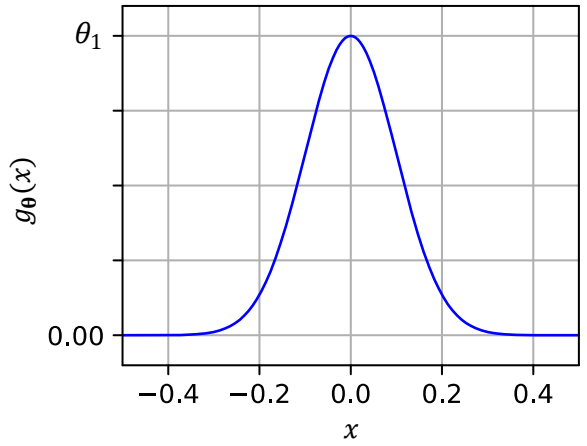


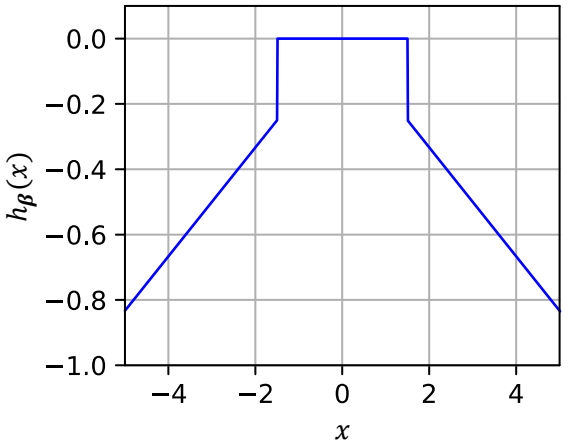
Hierarchical control goals:

1. a) Minimize the position error: $\min e^{\mathrm{P}}_y$
   b) Minimize the velocity error: $\min e^{\mathrm{P}}_{v_x}$
   c) Minimize the orientation error: $\min e_\psi$
2. Generate smooth steering reference: $\min \Delta\delta^i$ with $\Delta\delta^i(k) = \delta^i(k) - \delta^i_{\mathrm{c}}(k-1); \forall\, i \in \{\mathrm{f}, \mathrm{r}\}$

Used auxiliary functions:

- $g_{\boldsymbol{\theta}}(x) = \theta_1\, e^{\frac{-x^2}{2\theta_2}}$
- $h_{\boldsymbol{\beta}}(x) = \begin{cases} 0, & \text{if } |x| < \beta_1 \\ -\beta_2 \cdot |x|, & \text{else} \end{cases}$

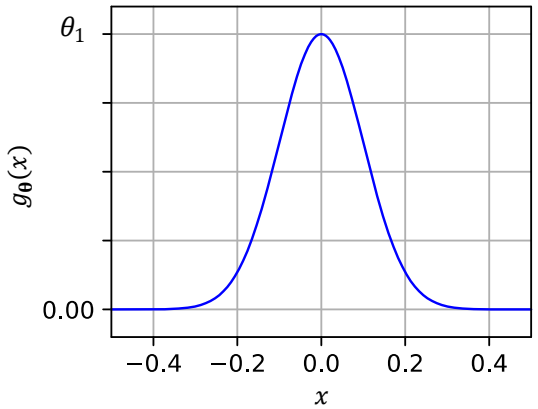


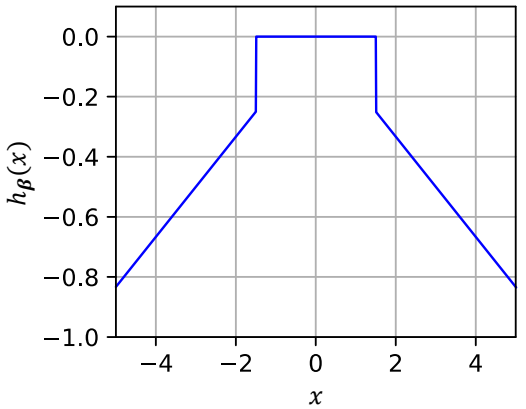


Reward Function Design (2)

$$r_{\mathrm{RL-PFC}}(\cdot) = \underbrace{g_{\boldsymbol{\theta}_e}\left(e_y^{\mathrm{P}}\right)\left(1 + g_{\boldsymbol{\theta}_\psi}\left(e_\psi^{\mathrm{P}}\right) + g_{\boldsymbol{\theta}_v}\left(e_{v_x}^{\mathrm{P}}\right)\right)}_{\text{1. goal: error minimization}} + \overbrace{g_{\boldsymbol{\theta}_{\Delta\delta}}\left(e_y^{\mathrm{P}}\right)\left(h_{\boldsymbol{\beta}_{\Delta\delta}}\left(\Delta\delta^{\mathrm{f}}\right) + h_{\boldsymbol{\beta}_{\Delta\delta}}\left(\Delta\delta^{\mathrm{r}}\right)\right)}^{\text{2. goal: smooth steering}}$$

Reward Function Design (2)





RL-PFC:

$$r_{\mathrm{RL-PFC}}(\cdot) = g_{\boldsymbol{\theta}_e}\left(e_y^{\mathrm{P}}\right)\left(1 + g_{\boldsymbol{\theta}_\psi}\left(e_\psi^{\mathrm{P}}\right) + g_{\boldsymbol{\theta}_v}\left(e_{v_x}^{\mathrm{P}}\right)\right) + g_{\boldsymbol{\theta}_{\Delta\delta}}\left(e_y^{\mathrm{P}}\right)\left(h_{\boldsymbol{\beta}_{\Delta\delta}}\left(\Delta\delta^{\mathrm{f}}\right) + h_{\boldsymbol{\beta}_{\Delta\delta}}\left(\Delta\delta^{\mathrm{r}}\right)\right)$$

$= 0$ for $e_y^{\mathrm{P}} \to \infty$
$= 1$ for $e_y^{\mathrm{P}} = 0$

$= 0$ for $e_\psi^{\mathrm{P}} \to \infty$
$= 1$ for $e_\psi^{\mathrm{P}} = 0$

$= 0$ for $e_{v_x}^{\mathrm{P}} \to \infty$
$= 1$ for $e_{v_x}^{\mathrm{P}} = 0$

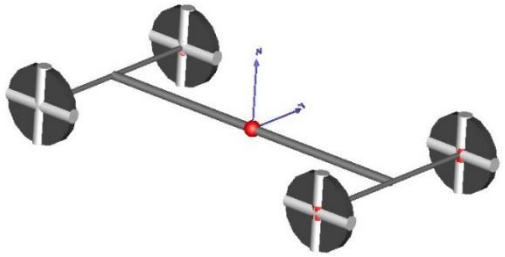
$= 0$ for $\left|\Delta\delta^i\right| < \beta_{\Delta\delta,1}$
< 0 else

RL-PFC-TV:

$$r_{\mathrm{RL-PFC\,TV}}(\cdot) = r_{\mathrm{RL-PFC}}(\cdot) - d_1 \sum_{i=1}^{4}\left(\frac{P_s^{\mathrm{W}i}}{d_2}\right)^2$$

with slip-based losses: $P_s^{\mathrm{W}i} = \boldsymbol{F}^{\mathrm{W}i} \cdot \boldsymbol{v}_s^{\mathrm{W}i}$

Implementation Details

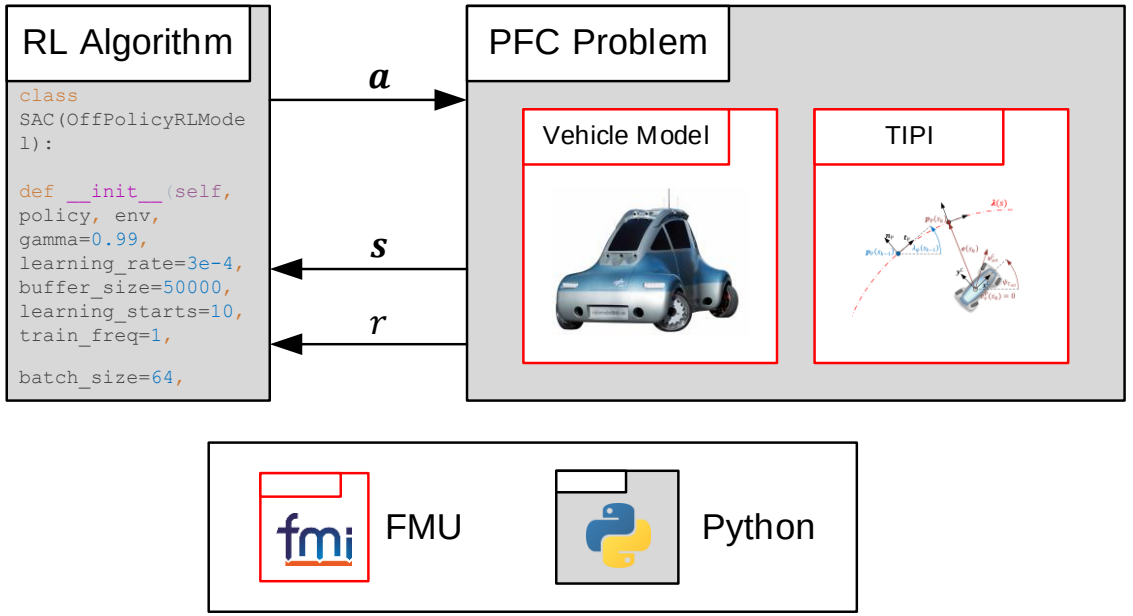


- **Aims of the training model:**
  - Fast simulation
  - Good approximation of the dominant dynamics

- The training model covers:
  - Planar two track model
  - Approximated actuator dynamics
  - Slip-based tire model

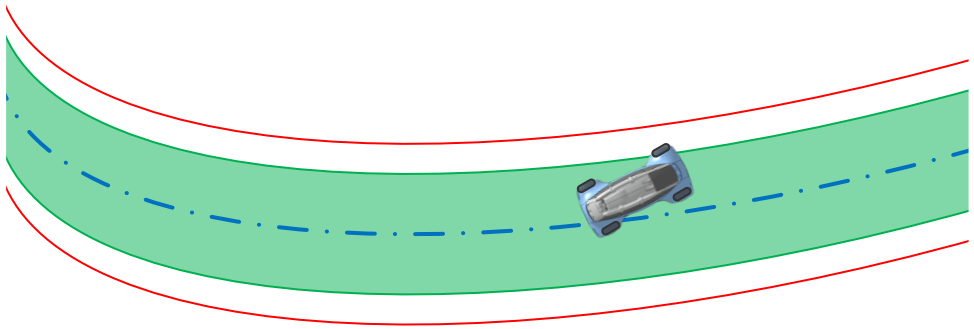
- Modular approach including 2 FMUs

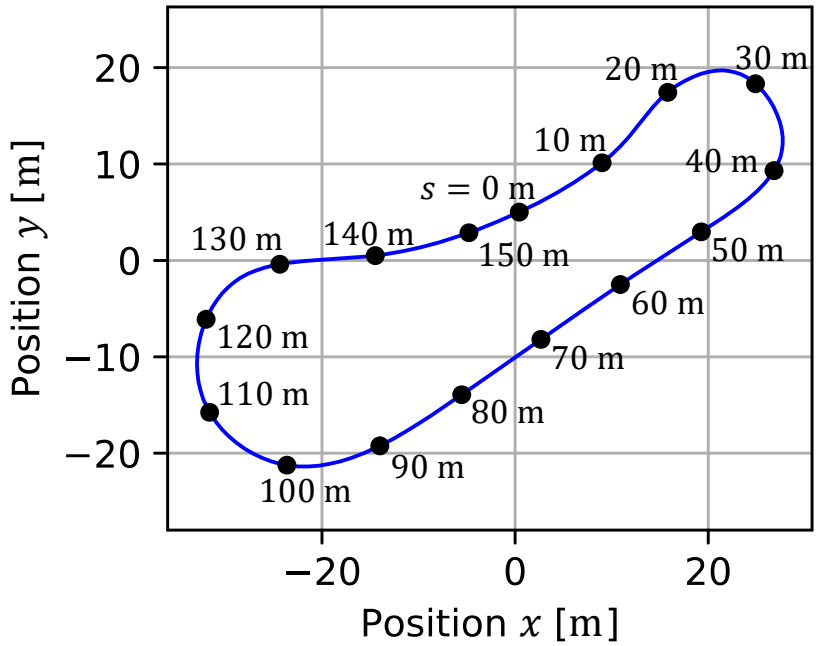
- RL algorithm Soft Actor-Critic (SAC):
  - Robust to hyperparametervariationen
  - High efficiency

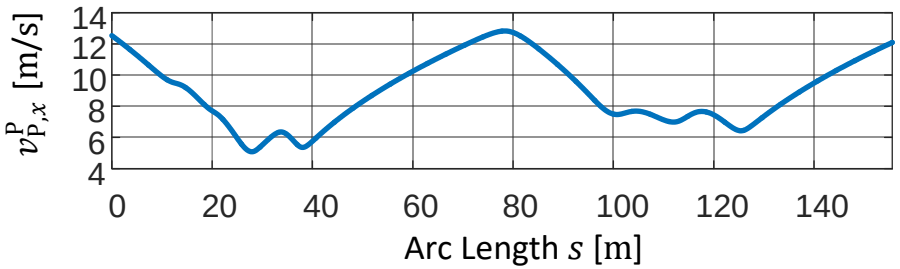


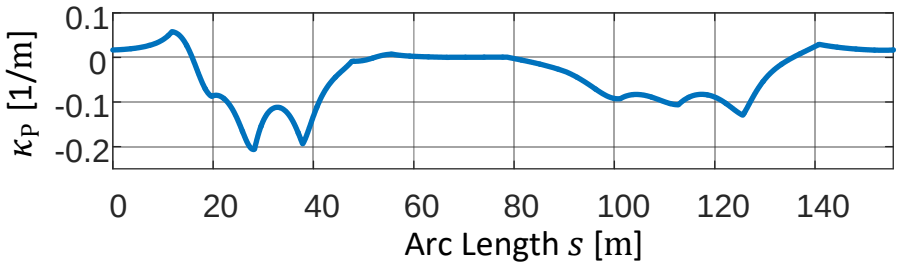
Trainingssetup

- Challenging training path (up to $5.5\frac{\mathrm{m}}{\mathrm{s}^2}$ lateral acceleration, tight curve)
- Random initialization to support exploration
- Early stopping of episodes to avoid unnecessary exploration



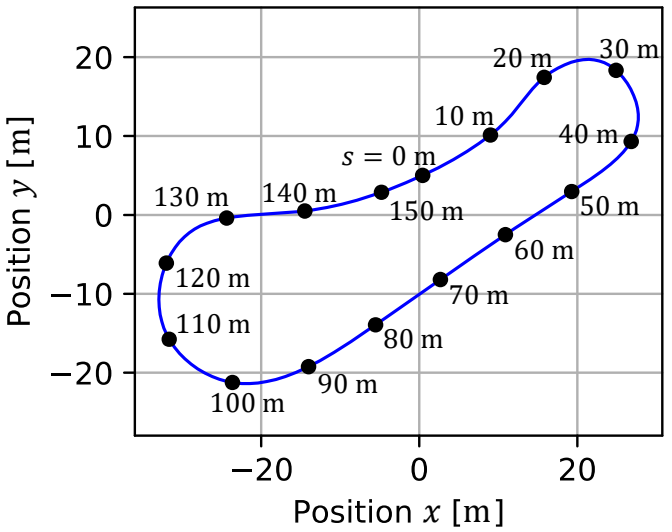






Simulative Assessment on Training Path

Error		RL-PFC TV	RL-PFC	ref PFC
Position: e_y^{P} [cm]	RMSE	**1.41**	1.60	2.39
	max	4.19	**3.55**	9.99
Velocity: $e_{v_x}^{\mathrm{P}}$ [m/s]	RMSE	0.165	0.174	**0.0965**
	max	0.413	0.526	**0.383**
Orientation: e_ψ [°]	RMSE	1.39	**0.679**	0.964
	max	3.39	**1.38**	2.66

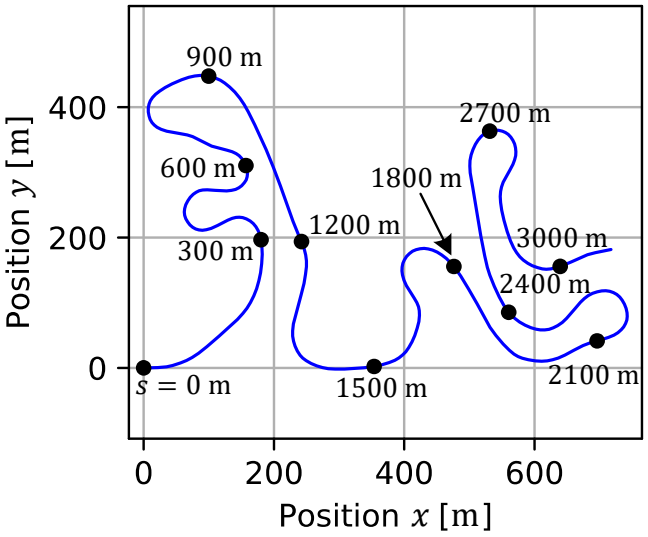


- ref PFC: 3 outer loop PD controllers combined with an optimization based control allocator (online optimization) [2]

- RL-PFC TV and RL-PFC exhibit superior position tracking
- RL-PFC exhibits superior orientation tracking
- RL-PFC controllers are designed to primary optimize position tracking
  - ➢ Velocity tracking is degraded in order to optimize position tracking
- RMSE: $\sqrt{\frac{1}{s_{\max}} \int_0^{s_{\max}} (e(s))^2 \, \mathrm{ds}}$

Simulative Assessment on Unknown Path

Error		RL-PFC TV	RL-PFC	ref PFC
Position: e_y^{P} [cm]	RMSE	1.47	1.54	**0.172**
	max	2.59	3.51	**0.579**
Velocity: $e_{v_x}^{\mathrm{P}}$ [m/s]	RMSE	0.130	0.142	**0.00652**
	max	0.311	0.314	**0.0140**
Orientation: e_ψ [°]	RMSE	0.866	0.813	**0.121**
	max	2.29	1.39	**0.354**



- ref PFC controller used for comparison performs better
- But: Both (model-free) RL-based controllers are able to follow the unknown path with an maximum position error in the cm range

➢ Training should cover a wide range of scenarios/ situations relevant in application

Thank you for your attention