Neuroevolution with CMA-ES for real-time gain tuning of a car-like robot controller

Ashley Hill[1], Eric Lucet[1], & Roland Lenain[2]

CEA, LIST, Interactive Robotics Laboratory[1]
Université Clermont Auvergne, Irstea, UR TSCF[2]

30th July 2019

Contents

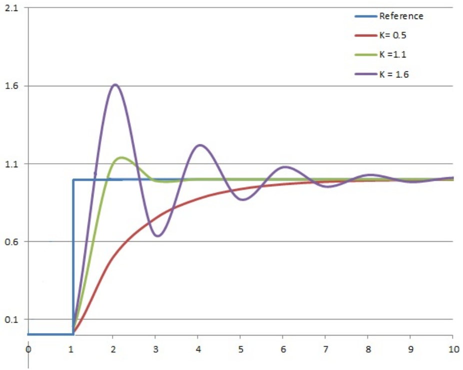
Controller gains and tuning - Controller gains

Controller gains :

$\Longrightarrow$ Control effort relative to the errors

$\Longrightarrow$ Time/distance to convergence

$\Longrightarrow$ Usually set so the controller is critically damped



Controller gains and tuning - Gains tuning

Desired behaviour from the optimal gain:

- A fast convergence to the setpoint
- A non-oscillatory control
- To minimise the errors

A lot of fixed gain tuning methods exist:

- Empirical tuning by hand
- Algorithmic methods (eg: Ziegler–Nichols)
- A black box optimiser in a simulation
- ...

Controller gains and tuning - Short comings of fixed gains

The optimal gain will depend on:

- Changes in the environment
- Changes in the perception quality
- Highly dynamics system

Dynamic gains - Dynamic gain family

Gain tuning

Continuous

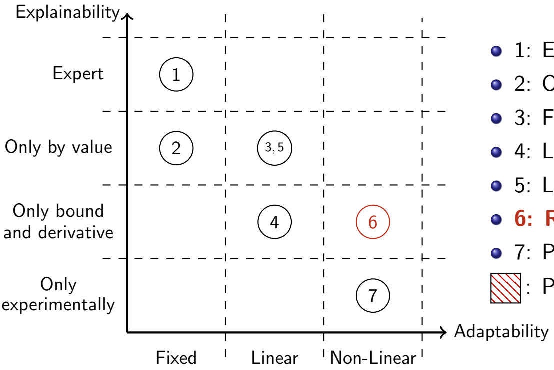
- Linear
- Neural network based
  - Reinforcement Learning
  - **Neuroevolution**

Discrete / Scheduling

- Fixed gain
  - Expert tuned
  - Optimiser tuned
- Gain scheduling
  - Fuzzy
  - Learnt state
  - Fixed

Figure 1: Dynamic gain family tree

Dynamic gains - Explainability vs adaptability



- 1: Expert tuned gain
- 2: Optimiser tuned gain
- 3: Fuzzy gain scheduling
- 4: Learnt state gain scheduling
- 5: Linear equation of the state
- **6: RL gain adaptation**
- 7: Pure RL controller
- : Pareto line

Figure 2: Explainability and adaptability compromise in controllers with gains.

Contents

Overview - Control Loop

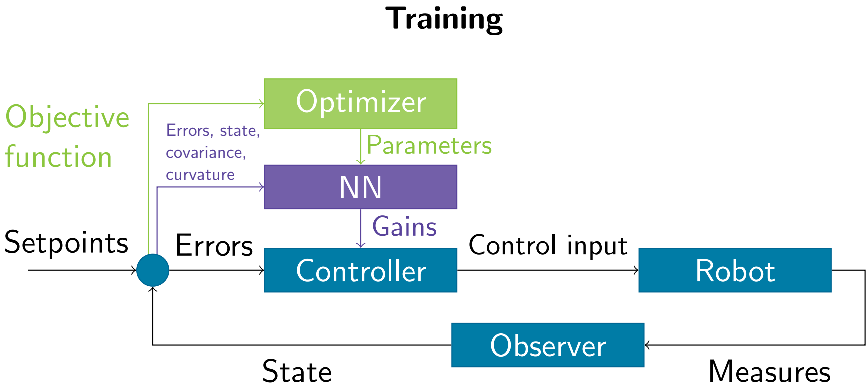


Figure 3: Overview of the proposed method.

Overview - Control Loop

Training - Online testing

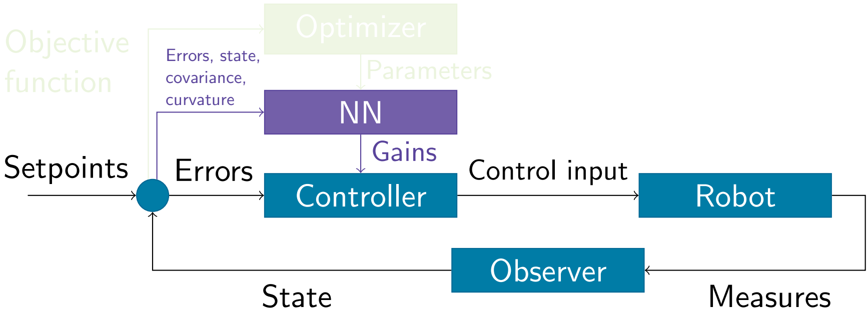


Figure 3: Overview of the proposed method.

Overview - Control Loop

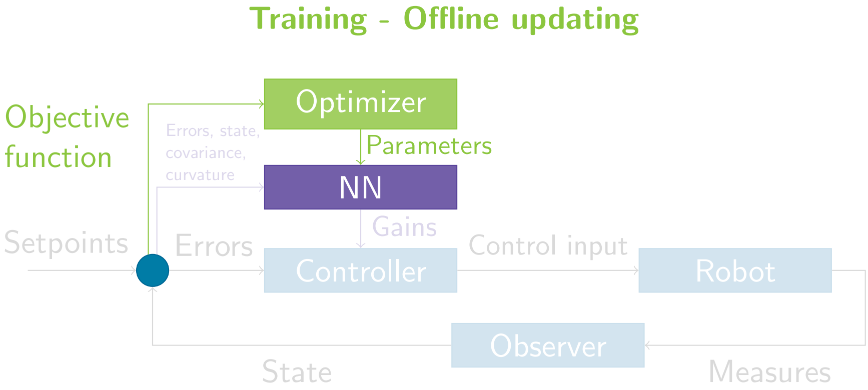


Figure 3: Overview of the proposed method.

Overview - Control Loop

Training - Online testing

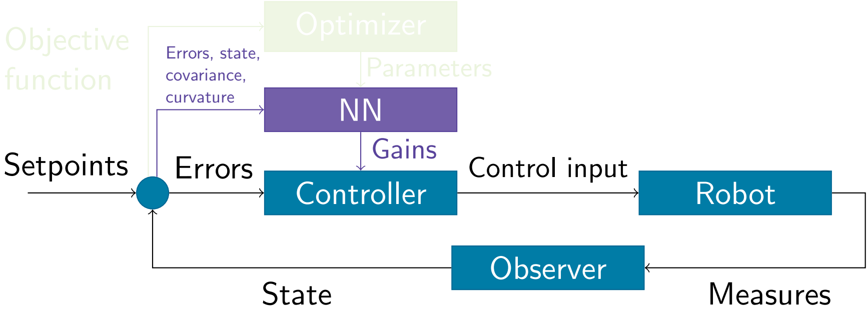


Figure 3: Overview of the proposed method.

Overview - Control Loop

Training - Offline updating

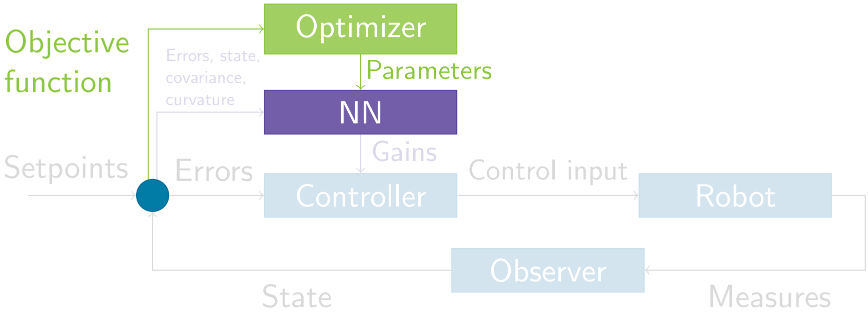


Figure 3: Overview of the proposed method.

Overview - Control Loop

Training - Online testing

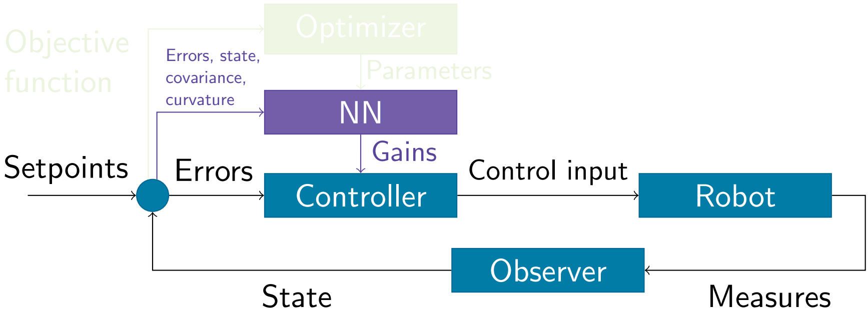


Figure 3: Overview of the proposed method.

Overview - Control Loop

Training - Offline updating

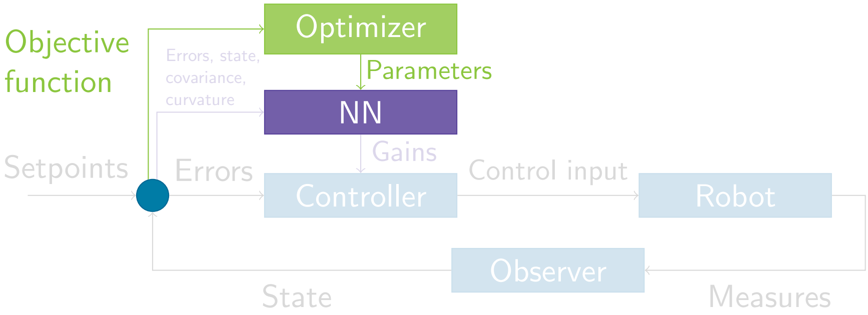


Figure 3: Overview of the proposed method.

Overview - Control Loop

End of training

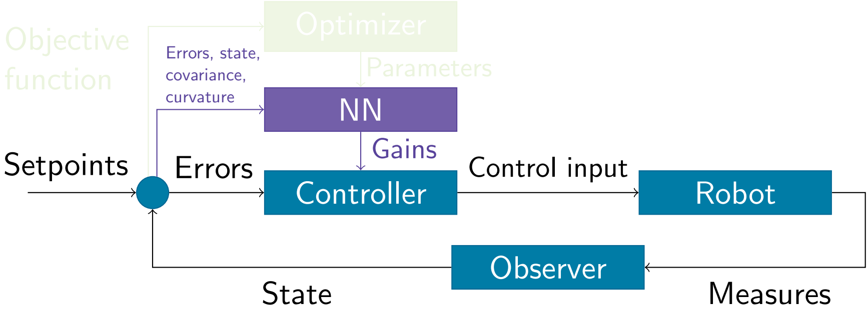


Figure 3: Overview of the proposed method.

Gain prediction Model - Neural network

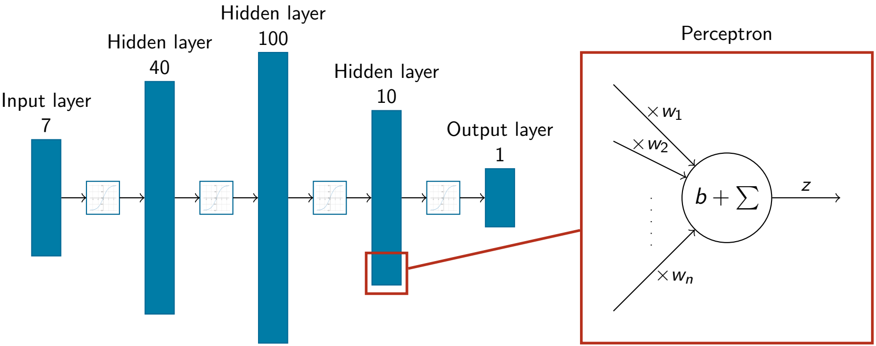


Figure 4: The Prediction model.

Training - CMA-ES

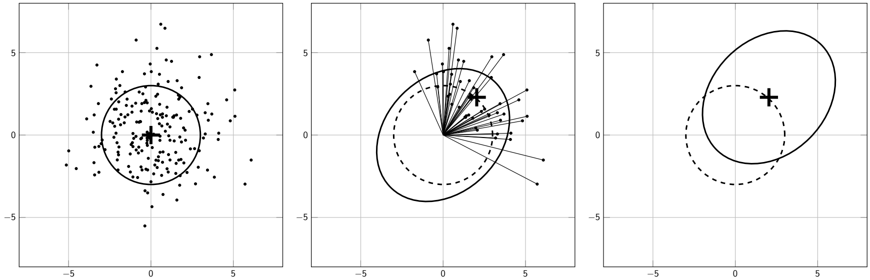


Figure 5: The CMA Evolution strategy.[1]

[1] Hansen, "The CMA Evolution Strategy: A Tutorial".

Training - Objective functions

The tested Objective functions:

- $$ob_1 = \sum_{\tau=0}^{T} |\epsilon_L(\tau)| + |\epsilon_\theta(\tau)\, L|$$

- $$ob_2 = \sum_{\tau=0}^{T} |\epsilon_L(\tau)| + |\epsilon_\theta(\tau)\, L| + |u_{\text{steer}}(\tau)\, L|$$

- $$ob_3 = \sum_{\tau=0}^{T} |\epsilon_L(\tau)| + |\epsilon_\theta(\tau)\, L| + k_{\text{steer}}\, |u_{\text{steer}}(\tau)\, L|$$

ob_3 is a generalisation of ob_1 and ob_2, with k_{steer} defining the penalty for the energy of the steering.

Contents

Modelling - Robot model

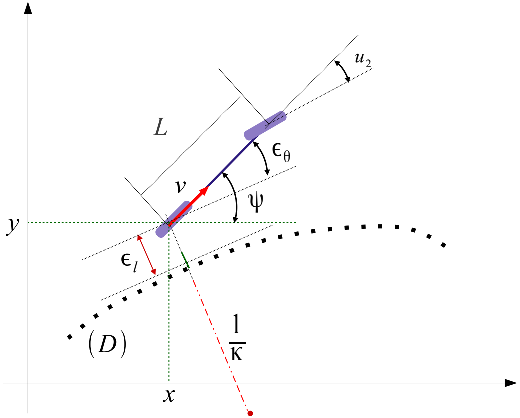


Figure 6: The mobile robot studied.

Modelling - Controller

$$u_2 = \arctan\left(\frac{L\cos^3\epsilon_\theta}{1-\kappa\epsilon_L}\left(2\sqrt{k_p}\tan\epsilon_\theta - \frac{k_p\epsilon_L}{1-\kappa\epsilon_L} + \frac{\kappa}{\cos^2(\epsilon_\theta)}\right)\right)$$ [2]

with :

- κ the curvature
- L the wheel base length
- ϵ_L the lateral error
- ϵ_θ the angular error
- k_p the gain defining the theoretical distance of convergence of the robot to the trajectory

[2] Lenain et al., "Robust sideslip angles observer for accurate off-road path tracking control".

Localisation - Extended Kalman Filter

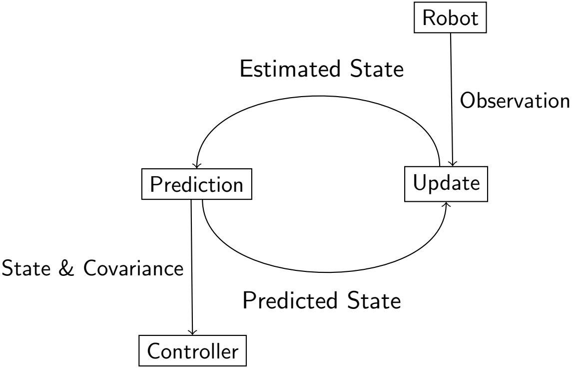


Figure 7: The Prediction model.[3]

[3]Welch and Bishop, *An Introduction to the Kalman Filter*.

Contents

Trajectories - The tested trajectories

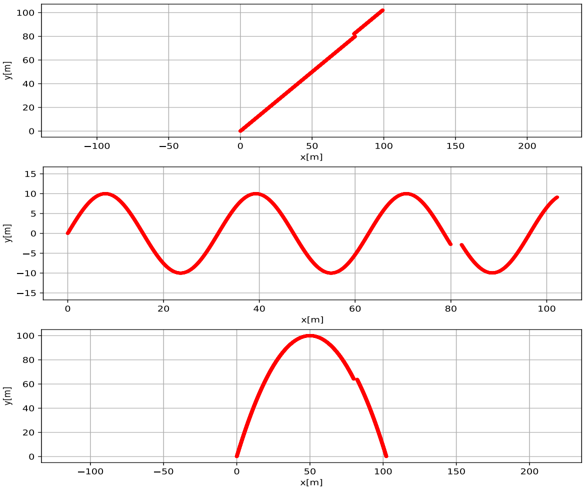


Figure 8: line, sine, and parabola trajectories, with a change lane.

Trajectories - The tested trajectories

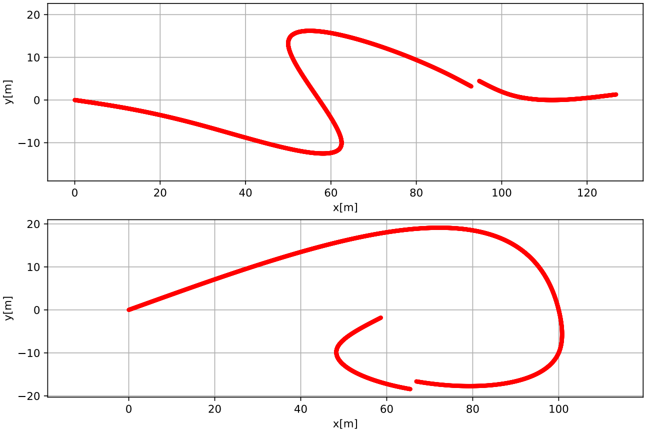


Figure 9: spline1 and spline2 trajectories, with a change lane.

Qualitative results - Gains vs path

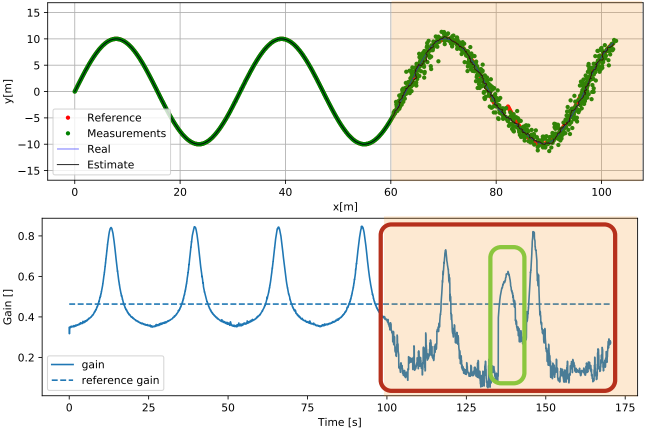


Figure 10: Above: the path, below: the gain

Qualitative results - Gains vs curvature

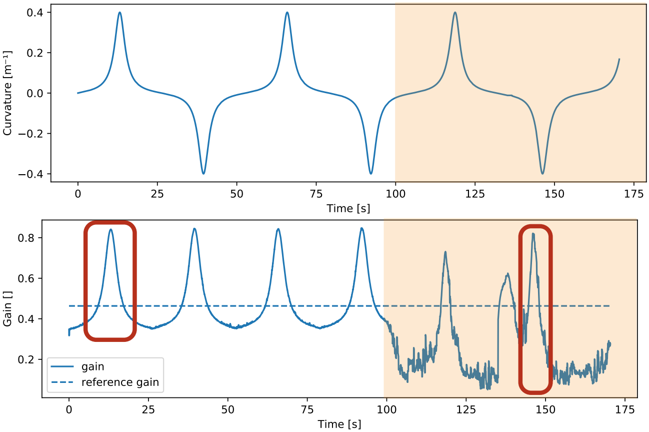


Figure 11: Above: the curvature, below: the gain

Qualitative results - Steering output

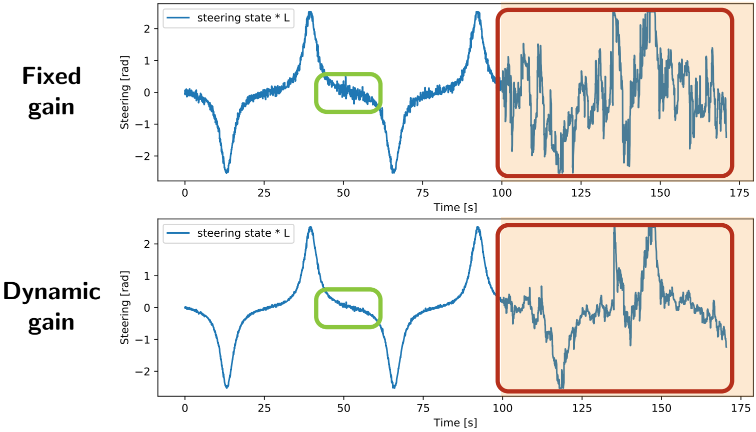


Figure 12: Above: the steering for the fixed gain, below: the steering for the proposed method

Quantitative results - Welsh t-test

trajectory	**ob_1**	**ob_2**	ob_3
line	**$3.22\mathrm{e}{-2}$**	$1.80\mathrm{e}{-8}$	$7.01\mathrm{e}{-5}$
sine	$1.61\mathrm{e}{-4}$	$4.07\mathrm{e}{-9}$	$6.06\mathrm{e}{-7}$
parabola	$6.53\mathrm{e}{-12}$	$4.20\mathrm{e}{-21}$	$1.49\mathrm{e}{-17}$
spline1	$2.09\mathrm{e}{-28}$	$3.63\mathrm{e}{-21}$	$5.08\mathrm{e}{-19}$
spline2	**$3.48\mathrm{e}{-1}$**	$8.42\mathrm{e}{-16}$	$1.49\mathrm{e}{-9}$

Table 1: Welch test[4] p-values between fixed gain and the suggested method, for every trajectory over the objective functions. With $k_{\text{steer}} = 0.5$ [5].

[4]Welch, "The generalization of 'student's' problem when several different population variances are involved".

[5]The value of k_{steer} was chosen so ob_3 would be the compromise between ob_1 and ob_2

Quantitative results - Objective function performance

trajectory	**$\mathbf{ob_1}$**	**$\mathbf{ob_2}$**	ob_3
line	**3.145%**	13.69%	9.466%
sine	4.165%	**4.575%**	**4.612%**
parabola	6.787%	**21.20%**	**17.95%**
spline1	**12.46%**	17.16%	17.20%
spline2	4.780%	10.68%	7.569%

Table 2: The percentage improvement of the objective functions for every trajectory relative to the fixed gain. With $k_{\text{steer}} = 0.5$.

Contents

Advantages - Comparison to gain tuning methods

Advantages of this method:

- Robot independent
- Controller independent
- Minimal prior knowledge
- Optimisable for a given task
- **Real time adaptation to external condition**
- A 10% to 20% improvement when compared to a constant gain in the example task

Limitations

Objective function selection

- Task specific
- Avoid local optima

Systematic error due to simulation

- Real world perturbation and noise
- Model error

Training time

- **CMA-ES Sample inefficient (1 year simulated time)**
- 5 hours wall time with 8 CPU cores of a i7-6820HQ

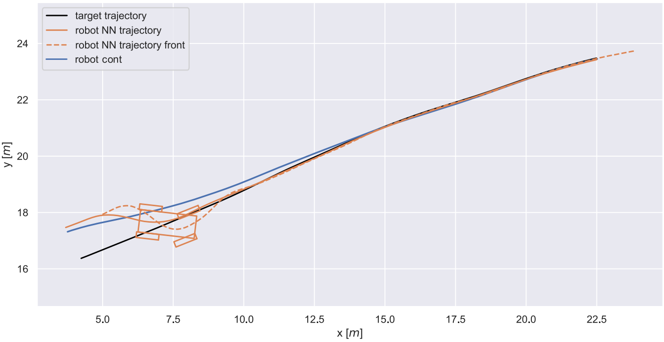
No stability proof:

- NN black boxes
- Only proof of stability through gain bounds

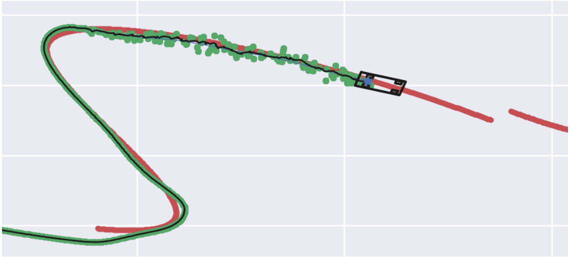
Future works

- Multiple gains prediction
- Dynamic simulation with action delay
- Tests on real robotics

Teaser:



Thank You.



Contents

Bibliography

Hansen, Nikolaus. "The CMA Evolution Strategy: A Tutorial". In: *CoRR* abs/1604.00772 (2016). arXiv: 1604.00772.

Hill, Ashley et al. *Stable Baselines*. https://github.com/hill-a/stable-baselines. 2018.

Lenain, R. et al. "Robust sideslip angles observer for accurate off-road path tracking control". In: *Advanced Robotics* 31.9 (2017), pp. 453–467.

Welch, B. L. "The generalization of 'student's' problem when several different population varlances are involved". In: *Biometrika* 34.1-2 (Jan. 1947), pp. 28–35. ISSN: 0006-3444. DOI: 10.1093/biomet/34.1-2.28.

Welch, Greg and Gary Bishop. *An Introduction to the Kalman Filter*. Tech. rep. Chapel Hill, NC, USA, 1995.

RL vs CMA-ES NN

trajectory	CMA-ES NN	SAC	PPO	DDPG	A2C	fixed gain
line	**41.60** (**±2.34**)	60.99 (±22.21)	115.11 (**±57.10**)	**158.14** (±4.09)	57.52 (±15.12)	48.20 (±2.45)
sine	**144.76** (±2.86)	1140.52 (**±3004.65**)	**2403.93** (±2824.32)	309.18 (±5.79)	280.51 (±175.61)	151.70 (**±2.83**)
parabola	**98.73** (±3.39)	191.48 (**±2.43**)	389.65 (±6.34)	**417.99** (±6.23)	208.80 (**±26.57**)	125.29 (±4.02)
spline1	**117.66** (**±3.04**)	508.24 (**±152.24**)	**2864.41** (±14.31)	272.20 (±5.27)	542.85 (±4.47)	142.04 (±3.45)
spline2	**147.32** (**±3.18**)	**6563.63** (**±3310.81**)	3169.80 (±23.61)	258.85 (±26.39)	437.22 (±736.97)	164.94 (±4.45)

Table 3: The values of the objective function ob_2 for every trajectory with RL methods. RL algorithms from the Stable-baselines library[6]

[6] Hill et al., *Stable Baselines*.

Robot equation

$$\dot{X} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} v \ \cos(\theta) + \alpha_x \\ v \ \sin(\theta) + \alpha_y \\ v \ \frac{\tan(u_2 + \alpha_{u_2})}{L} + \alpha_\theta \\ u_1 + \alpha_{u_1} \end{pmatrix} \tag{1}$$

with:

- x, y The position in the global reference
- θ The direction of the robot
- v The speed of the robot
- u_1 The acceleration command
- u_2 The steering command
- α White noise

Qualitative results - Ablation

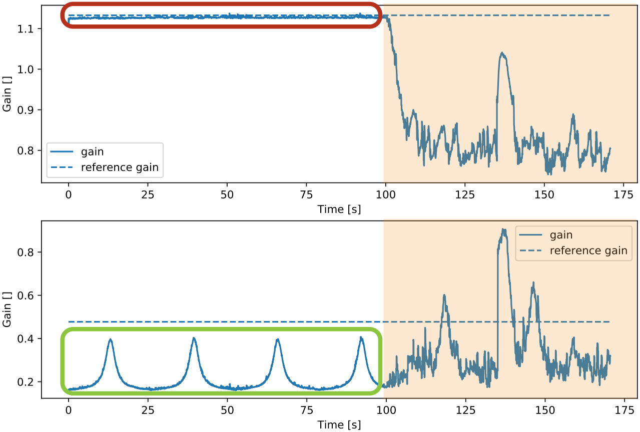


Figure 13: Above: the gain without the curvature, below: the gain without the Kalman covariance

Qualitative results - Training over time

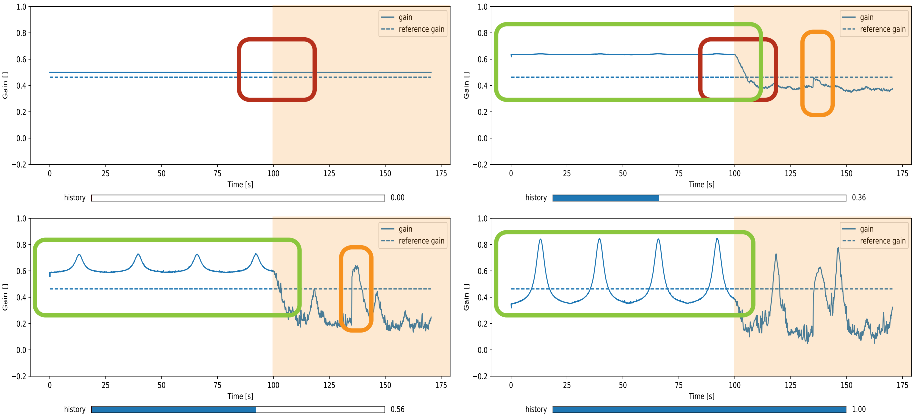


Figure 14: The gain over the training history

Objective function values

trajectory	ob_1 CMA-ES NN	ob_1 fixed gain	ob_2 CMA-ES NN	ob_2 fixed gain	ob_3 CMA-ES NN	ob_3 fixed gain
line	**27.41** (±**1.98**)	28.30 (±2.08)	**41.60** (±**2.34**)	48.20 (±2.45)	**36.63** (±**2.63**)	40.46 (±4.65)
sine	**39.81** (±2.18)	41.54 (±2.18)	**144.76** (±2.86)	151.70 (±**2.83**)	**94.11** (±**2.32**)	98.66 (±2.37)
parabola	**64.41** (±**2.90**)	69.10 (±3.04)	**98.73** (±**3.39**)	125.29 (±4.02)	**83.38** (±**3.27**)	101.62 (±4.00)
spline1	**52.34** (±**2.14**)	59.79 (±2.52)	**117.66** (±**3.04**)	142.04 (±3.45)	**87.30** (±**3.07**)	105.43 (±3.54)
spline2	**68.33** (±**2.59**)	71.76 (±25.26)	**147.32** (±**3.18**)	164.94 (±4.45)	**110.03** (±**3.12**)	119.04 (±3.79)

Table 4: The values of the objective functions for every trajectory. With $k_{\text{steer}} = 0.5$.

Bi-gain

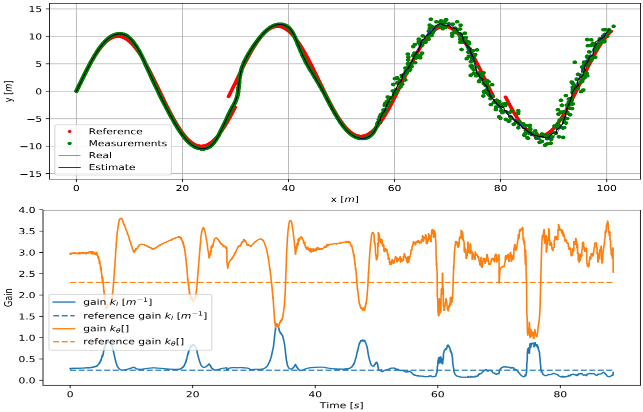


Figure 15: Above: the path, below: the gain