Implementation of GGN and SCQP methods via first-order Taylor approximations

Alejandro Astudillo

Ph.D. researcher

MECO Research Team

August 4, 2021

Problem formulation

Optimization problem with convex-over-nonlinear constraints

$$
\begin{aligned}
&min_w \; \phi_0(c_0(w)) \\
&s.t. \quad g_i(w) = 0, \qquad\qquad i = 1, ..., p \\
&\qquad\quad \phi_i(c_i(w)) \leq 0, \qquad i = 1, ..., q.
\end{aligned}
$$

KU LEUVEN

Problem formulation

Optimization problem with convex-over-nonlinear constraints

$$
\begin{aligned}
min_w \ & \phi_0(c_0(w)) \\
s.t. \quad & g_i(w) = 0, \qquad i = 1, ..., p \\
& \phi_i(c_i(w)) \leq 0, \quad i = 1, ..., q.
\end{aligned}
$$

QP from SQP method

$$
\begin{aligned}
min_{s_k} \ & \frac{1}{2} s_k^T B_k s_k + \nabla\psi_0(w_k)^T s_k \\
s.t. \quad & \nabla g_i(w_k)^T s_k + g_i(w_k) = 0, \qquad i = 1, ..., p \\
& \nabla\psi_i(w_k)^T s_k + \psi_i(w_k) \leq 0, \quad i = 1, ..., q.
\end{aligned}
$$

Problem formulation

Optimization problem with convex-over-nonlinear constraints

$$
\begin{aligned}
&min_w\ \phi_0(c_0(w)) \\
&s.t.\quad g_i(w) = 0, \qquad\qquad i = 1, ..., p \\
&\qquad\quad \phi_i(c_i(w)) \leq 0, \qquad i = 1, ..., q.
\end{aligned}
$$

QP from SQP method

$$
\begin{aligned}
&min_{s_k}\ \frac{1}{2} s_k^T B_k s_k + \nabla\psi_0(w_k)^T s_k \\
&s.t.\quad \nabla g_i(w_k)^T s_k + g_i(w_k) = 0, \qquad i = 1, ..., p \\
&\qquad\quad \nabla\psi_i(w_k)^T s_k + \psi_i(w_k) \leq 0, \qquad i = 1, ..., q.
\end{aligned}
$$

Exact Hessian of the Lagrangian

$$B_k = \nabla_w^2 \mathcal{L}(w_k, \lambda_k, \mu_k).$$

Problem formulation

Optimization problem with convex-over-nonlinear constraints

$$\begin{aligned} & min_w \ \phi_0(c_0(w)) \\ & s.t. \quad g_i(w) = 0, \qquad i = 1, ..., p \\ & \qquad\ \ \phi_i(c_i(w)) \leq 0, \quad i = 1, ..., q. \end{aligned}$$

QP from SQP method

$$\begin{aligned} & min_{s_k} \ \frac{1}{2} s_k^T B_k s_k + \nabla \psi_0(w_k)^T s_k \\ & s.t. \quad \nabla g_i(w_k)^T s_k + g_i(w_k) = 0, \qquad i = 1, ..., p \\ & \qquad\ \ \nabla \psi_i(w_k)^T s_k + \psi_i(w_k) \leq 0, \quad i = 1, ..., q. \end{aligned}$$

SCQP Hessian approximation

$$\begin{aligned} B_k^{SCQP}(w, \mu) := & \frac{\partial c_0(w)}{\partial w}^T \nabla_c^2 \phi_0(c_0(w)) \frac{\partial c_0(w)}{\partial w} \\ & + \sum_{i=1}^{q} \mu_i \frac{\partial c_i(w)}{\partial w}^T \nabla_c^2 \phi_i(c_i(w)) \frac{\partial c_i(w)}{\partial w} \succeq 0. \end{aligned}$$

Problem formulation

Optimization problem with convex-over-nonlinear constraints

$$\begin{aligned} & min_w \ \phi_0(c_0(w)) \\ & s.t. \quad g_i(w) = 0, \qquad i = 1, ..., p \\ & \qquad \phi_i(c_i(w)) \leq 0, \quad i = 1, ..., q. \end{aligned}$$

QP from SQP method

$$\begin{aligned} & min_{s_k} \ \frac{1}{2} s_k^T B_k s_k + \nabla \psi_0(w_k)^T s_k \\ & s.t. \quad \nabla g_i(w_k)^T s_k + g_i(w_k) = 0, \qquad i = 1, ..., p \\ & \qquad \nabla \psi_i(w_k)^T s_k + \psi_i(w_k) \leq 0, \quad i = 1, ..., q. \end{aligned}$$

SCQP Hessian approximation

$$\begin{aligned} B_k^{SCQP}(w, \mu) := & \frac{\partial c_0(w)}{\partial w}^T \nabla_c^2 \phi_0(c_0(w)) \frac{\partial c_0(w)}{\partial w} \\ & + \sum_{i=1}^{q} \mu_i \frac{\partial c_i(w)}{\partial w}^T \nabla_c^2 \phi_i(c_i(w)) \frac{\partial c_i(w)}{\partial w} \succeq 0. \end{aligned}$$

First-order Taylor approximation

$$f_{lin}(w, \overline{w}) = f(\overline{w}) + \nabla f(\overline{w})^T (w - \overline{w})$$

Problem formulation

Optimization problem with convex-over-nonlinear constraints

$$
\begin{aligned}
& min_w \ \phi_0(c_0(w)) \\
s.t. \quad & g_i(w) = 0, \qquad i = 1, \ldots, p \\
& \phi_i(c_i(w)) \leq 0, \qquad i = 1, \ldots, q.
\end{aligned}
$$

QP from SQP method

$$
\begin{aligned}
& min_{s_k} \ \frac{1}{2} s_k^T B_k s_k + \nabla\psi_0(w_k)^T s_k \\
s.t. \quad & \nabla g_i(w_k)^T s_k + g_i(w_k) = 0, \qquad i = 1, \ldots, p \\
& \nabla\psi_i(w_k)^T s_k + \psi_i(w_k) \leq 0, \qquad i = 1, \ldots, q.
\end{aligned}
$$

SCQP Hessian approximation

$$
\begin{aligned}
B_k^{SCQP}(w, \mu) := & \frac{\partial c_0(w)}{\partial w}^T \nabla_c^2 \phi_0(c_0(w)) \frac{\partial c_0(w)}{\partial w} \\
& + \sum_{i=1}^{q} \mu_i \frac{\partial c_i(w)}{\partial w}^T \nabla_c^2 \phi_i(c_i(w)) \frac{\partial c_i(w)}{\partial w} \succeq 0.
\end{aligned}
$$

First-order Taylor approximation

$$
f_{lin}(w, \overline{w}) = f(\overline{w}) + \nabla f(\overline{w})^T (w - \overline{w})
$$

Trick in evaluation with the same symbolic variable

$$
\left.
\begin{aligned}
\tilde{f}(w) &:= f_{lin}(w, w) = f(w), \\
\nabla \tilde{f}(w) &:= \nabla f(w), \\
\nabla^2 \tilde{f}(w) &:= 0
\end{aligned}
\right\} \ lin(f(w))
$$

Problem formulation

Optimization problem with convex-over-nonlinear constraints

$$
\begin{aligned}
& min_w \ \phi_0(c_0(w)) \\
& s.t. \quad g_i(w) = 0, \qquad i = 1, ..., p \\
& \qquad \ \ \phi_i(c_i(w)) \leq 0, \qquad i = 1, ..., q.
\end{aligned}
$$

QP from SQP method

$$
\begin{aligned}
& min_{s_k} \ \frac{1}{2} s_k^T B_k s_k + \nabla \psi_0(w_k)^T s_k \\
& s.t. \quad \nabla g_i(w_k)^T s_k + g_i(w_k) = 0, \qquad i = 1, ..., p \\
& \qquad \ \ \nabla \psi_i(w_k)^T s_k + \psi_i(w_k) \leq 0, \qquad i = 1, ..., q.
\end{aligned}
$$

SCQP Hessian approximation

$$
\begin{aligned}
B_k^{SCQP}(w, \mu) := & \frac{\partial c_0(w)}{\partial w}^T \nabla_c^2 \phi_0(c_0(w)) \frac{\partial c_0(w)}{\partial w} \\
& + \sum_{i=1}^{q} \mu_i \frac{\partial c_i(w)}{\partial w}^T \nabla_c^2 \phi_i(c_i(w)) \frac{\partial c_i(w)}{\partial w} \succeq 0.
\end{aligned}
$$

First-order Taylor approximation

$$
f_{lin}(w, \overline{w}) = f(\overline{w}) + \nabla f(\overline{w})^T (w - \overline{w})
$$

Trick in evaluation with the same symbolic variable

$$
\left.
\begin{aligned}
\tilde{f}(w) &:= f_{lin}(w, w) = f(w), \\
\nabla \tilde{f}(w) &:= \nabla f(w), \\
\nabla^2 \tilde{f}(w) &:= 0
\end{aligned}
\right\} \ lin(f(w))
$$

Ready to be solved with SCQP

$$
\begin{aligned}
& min_w \ \phi_0(lin(c_0(w))) \\
& s.t. \quad lin(g_i(w)) = 0, \qquad i = 1, ..., p \\
& \qquad \ \ \phi_i(lin(c_i(w))) \leq 0, \qquad i = 1, ..., q.
\end{aligned}
$$

Problem formulation

Optimization problem with convex-over-nonlinear constraints

$$
\begin{aligned}
&min_w \ \phi_0(c_0(w)) \\
&s.t. \quad g_i(w) = 0, \qquad i = 1, \ldots, p \\
&\qquad\quad \phi_i(c_i(w)) \leq 0, \quad i = 1, \ldots, q.
\end{aligned}
$$

QP from SQP method

$$
\begin{aligned}
&min_{s_k} \ \frac{1}{2} s_k^T B_k s_k + \nabla \psi_0(w_k)^T s_k \\
&s.t. \quad \nabla g_i(w_k)^T s_k + g_i(w_k) = 0, \qquad i = 1, \ldots, p \\
&\qquad\quad \nabla \psi_i(w_k)^T s_k + \psi_i(w_k) \leq 0, \quad i = 1, \ldots, q.
\end{aligned}
$$

SCQP Hessian approximation

$$
\begin{aligned}
B_k^{SCQP}(w, \mu) := &\frac{\partial c_0(w)}{\partial w}^T \nabla_c^2 \phi_0(c_0(w)) \frac{\partial c_0(w)}{\partial w} \\
&+ \sum_{i=1}^{q} \mu_i \frac{\partial c_i(w)}{\partial w}^T \nabla_c^2 \phi_i(c_i(w)) \frac{\partial c_i(w)}{\partial w} \succeq 0.
\end{aligned}
$$

First-order Taylor approximation

$$
f_{lin}(w, \overline{w}) = f(\overline{w}) + \nabla f(\overline{w})^T (w - \overline{w})
$$

Trick in evaluation with the same symbolic variable

$$
\left.
\begin{aligned}
\tilde{f}(w) &:= f_{lin}(w, w) = f(w), \\
\nabla \tilde{f}(w) &:= \nabla f(w), \\
\nabla^2 \tilde{f}(w) &:= 0
\end{aligned}
\right\} \ lin(f(w))
$$

Ready to be solved with GGN

$$
\begin{aligned}
&min_w \ \phi_0(lin(c_0(w))) \\
&s.t. \quad lin(g_i(w)) = 0, \qquad i = 1, \ldots, p \\
&\qquad\quad lin(\phi_i(c_i(w))) \leq 0, \quad i = 1, \ldots, q.
\end{aligned}
$$

Results

Tunnel-following NMPC for a Robot Manipulator

$$
\min \int_0^{t_f} \left\| \begin{bmatrix} \dot{\theta}(\tau) - \dot{\theta}_{ref}(\theta(\tau)) \\ e_T(q(\tau), \theta(\tau)) \\ x(\tau) \\ u(\tau) \\ \nu(\tau) \end{bmatrix} \right\|_{\mathbf{W}}^2 + \alpha_l l(\tau) + \alpha_l l_{\mathcal{O}}(\tau)\, d\tau
$$

$$
\begin{aligned}
s.t. \quad & 0 = f_x(\dot{x}(\tau), x(\tau), u(\tau)), \\
& \dot{\zeta}(\tau) = f_\zeta(\zeta(\tau), \nu(\tau)), \\
& \underline{r}(\tau) \leq r(x(\tau), u(\tau)) \leq \overline{r}(\tau), \\
& \underline{q}(\tau) \leq q(\tau) \leq \overline{q}(\tau), \\
& ||e_p(q(\tau), \theta(\tau))||^2 \leq \rho^2 + l(\tau), \\
& ||e_{\mathcal{O}}(q(\tau), \theta(\tau))||^2 \leq \rho_{\mathcal{O}}^2 + l_{\mathcal{O}}(\tau), \\
& l(\tau) \geq 0, \\
& l_{\mathcal{O}}(\tau) \geq 0, \\
& 0 \leq \theta(\tau) \leq \overline{\theta}, \\
& x(0) = \hat{x}_0, \\
& (x(t_f), \zeta(t_f)) \in \varepsilon.
\end{aligned}
$$

Results

Tunnel-following NMPC for a Robot Manipulator

$$\min \int_0^{t_f} \boxed{\left\| \begin{bmatrix} \dot{\theta}(\tau) - \dot{\theta}_{ref}(\theta(\tau)) \\ e_T(q(\tau), \theta(\tau)) \\ x(\tau) \\ u(\tau) \\ \nu(\tau) \end{bmatrix} \right\|_{\mathbf{W}}^2} + \alpha_l l(\tau) + \alpha_l l_{\mathcal{O}}(\tau)\, d\tau$$

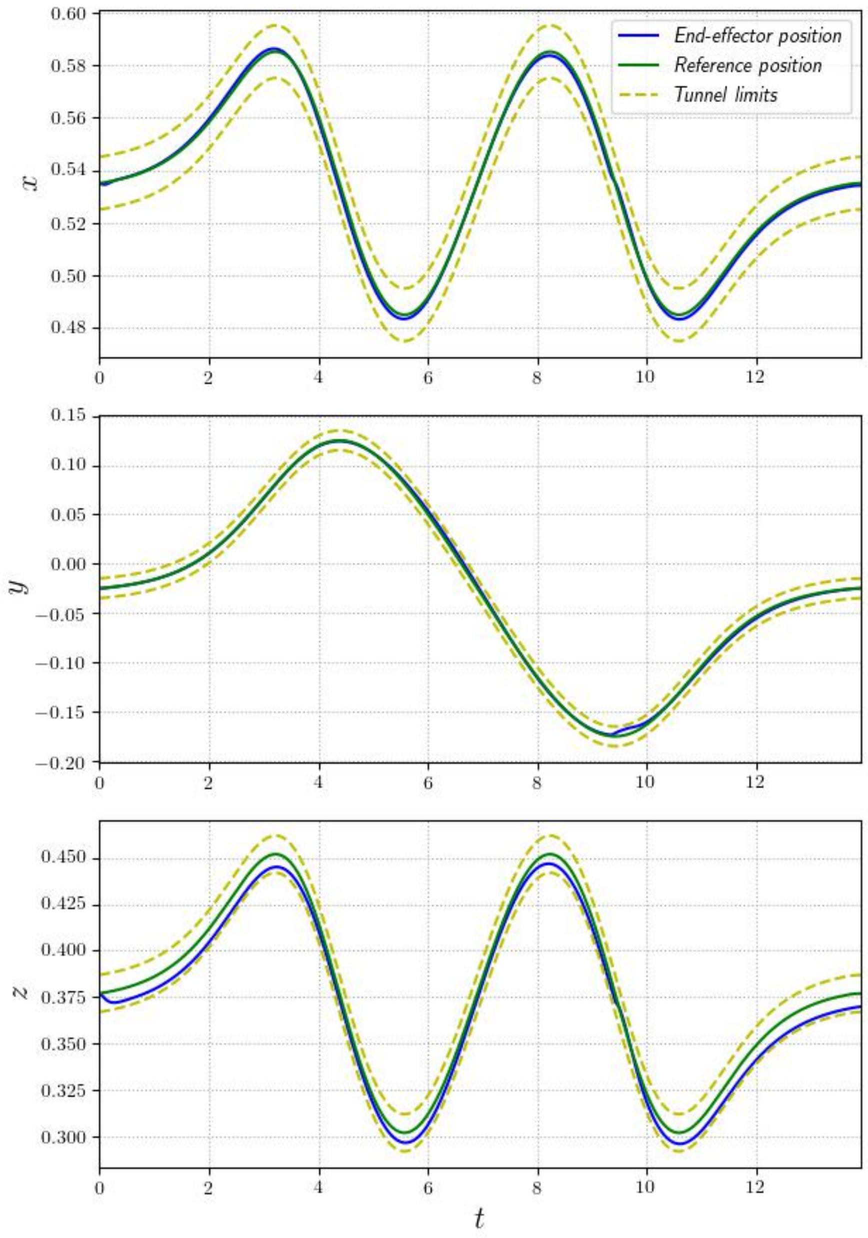
$$\begin{aligned}
s.t. \quad & 0 = f_x(\dot{x}(\tau), x(\tau), u(\tau)), \\
& \dot{\zeta}(\tau) = f_\zeta(\zeta(\tau), \nu(\tau)), \\
& \underline{r}(\tau) \leq r(x(\tau), u(\tau)) \leq \overline{r}(\tau), \\
& \underline{q}(\tau) \leq q(\tau) \leq \overline{q}(\tau), \\
& \boxed{||e_p(q(\tau), \theta(\tau))||^2} \leq \rho^2 + l(\tau), \\
& \boxed{||e_{\mathcal{O}}(q(\tau), \theta(\tau))||^2} \leq \rho_{\mathcal{O}}^2 + l_{\mathcal{O}}(\tau), \\
& l(\tau) \geq 0, \\
& l_{\mathcal{O}}(\tau) \geq 0, \\
& 0 \leq \theta(\tau) \leq \overline{\theta}, \\
& x(0) = \hat{x}_0, \\
& (x(t_f), \zeta(t_f)) \in \varepsilon.
\end{aligned}$$

Results

Tunnel-following NMPC for a Robot Manipulator

$$
min \int_0^{t_f} \boxed{\left\| \begin{bmatrix} \dot\theta(\tau) - \dot\theta_{ref}(\theta(\tau)) \\ e_T(q(\tau), \theta(\tau)) \\ x(\tau) \\ u(\tau) \\ \nu(\tau) \end{bmatrix} \right\|_{\mathbf{W}}^2} + \alpha_l l(\tau) + \alpha_l l_{\mathcal{O}}(\tau)\, d\tau
$$

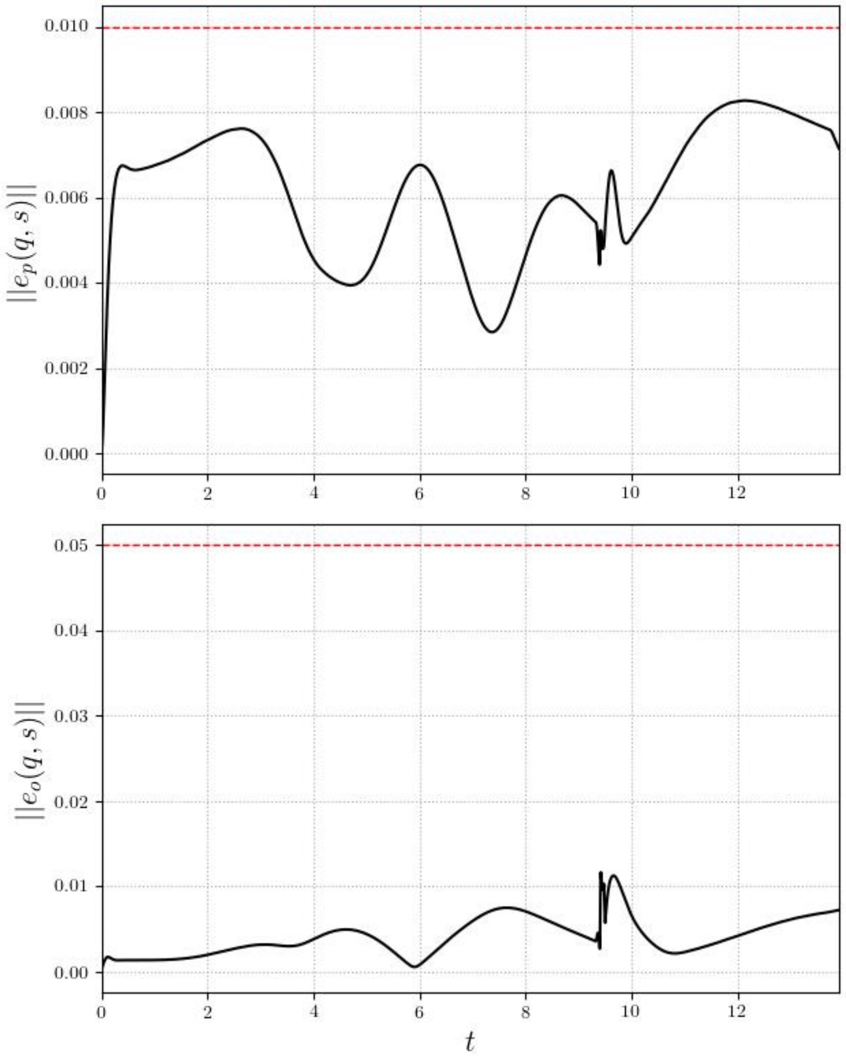
$$
\begin{aligned}
s.t. \quad & 0 = f_x(\dot x(\tau), x(\tau), u(\tau)), \\
& \dot\zeta(\tau) = f_\zeta(\zeta(\tau), \nu(\tau)), \\
& \underline{r}(\tau) \le r(x(\tau), u(\tau)) \le \overline{r}(\tau), \\
& \underline{q}(\tau) \le q(\tau) \le \overline{q}(\tau), \\
& \boxed{||e_p(q(\tau), \theta(\tau))||^2} \le \rho^2 + l(\tau), \\
& \boxed{||e_{\mathcal{O}}(q(\tau), \theta(\tau))||^2} \le \rho_{\mathcal{O}}^2 + l_{\mathcal{O}}(\tau), \\
& l(\tau) \ge 0, \\
& l_{\mathcal{O}}(\tau) \ge 0, \\
& 0 \le \theta(\tau) \le \overline{\theta}, \\
& x(0) = \hat{x}_0, \\
& (x(t_f), \zeta(t_f)) \in \varepsilon.
\end{aligned}
$$

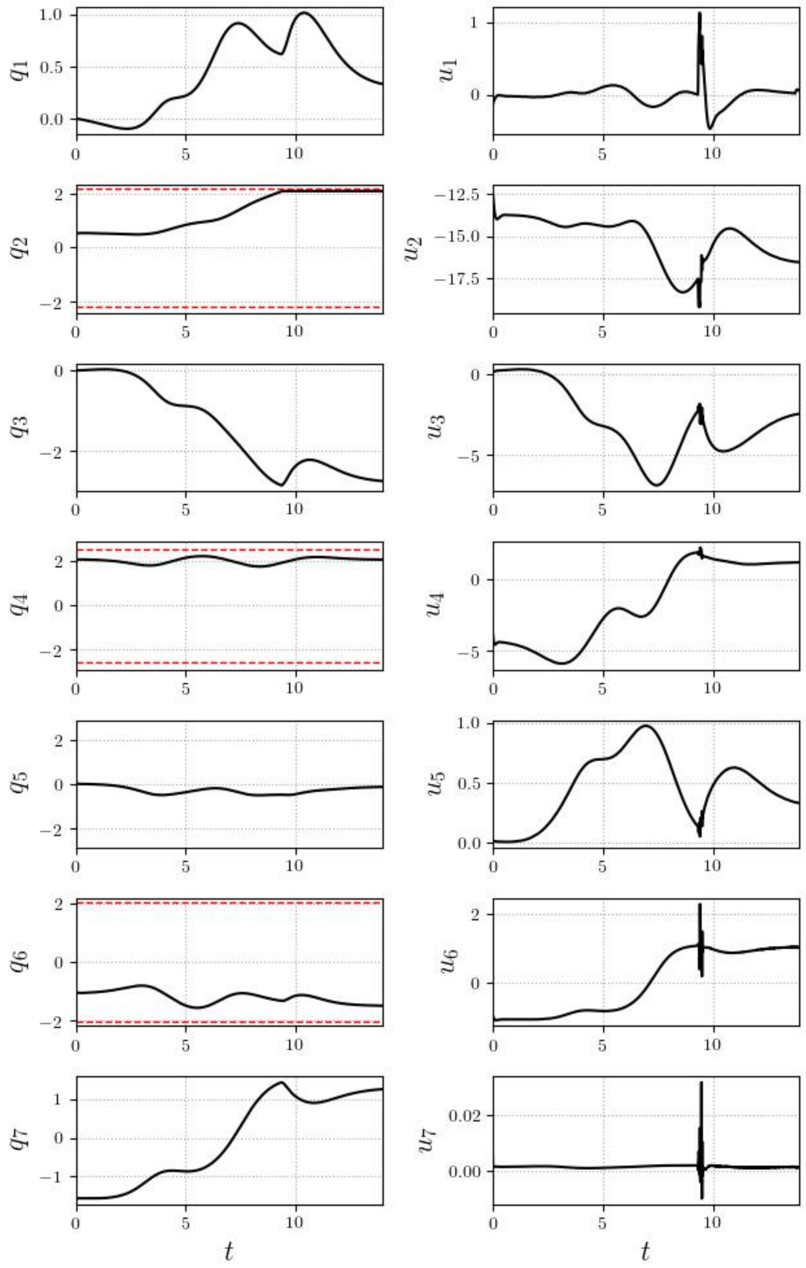


Thank you!

e-mail: alejandro.astudillovigoya@kuleuven.be

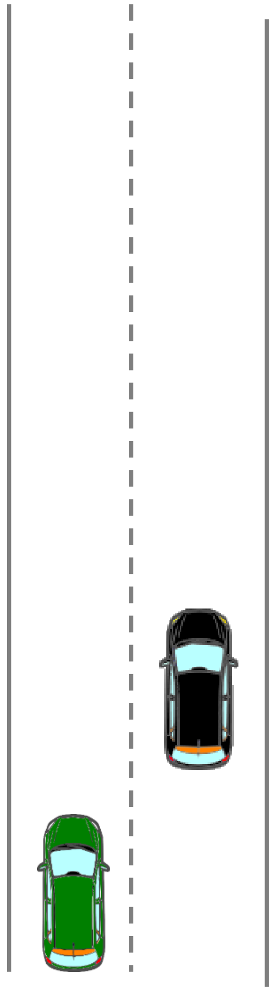
Extra





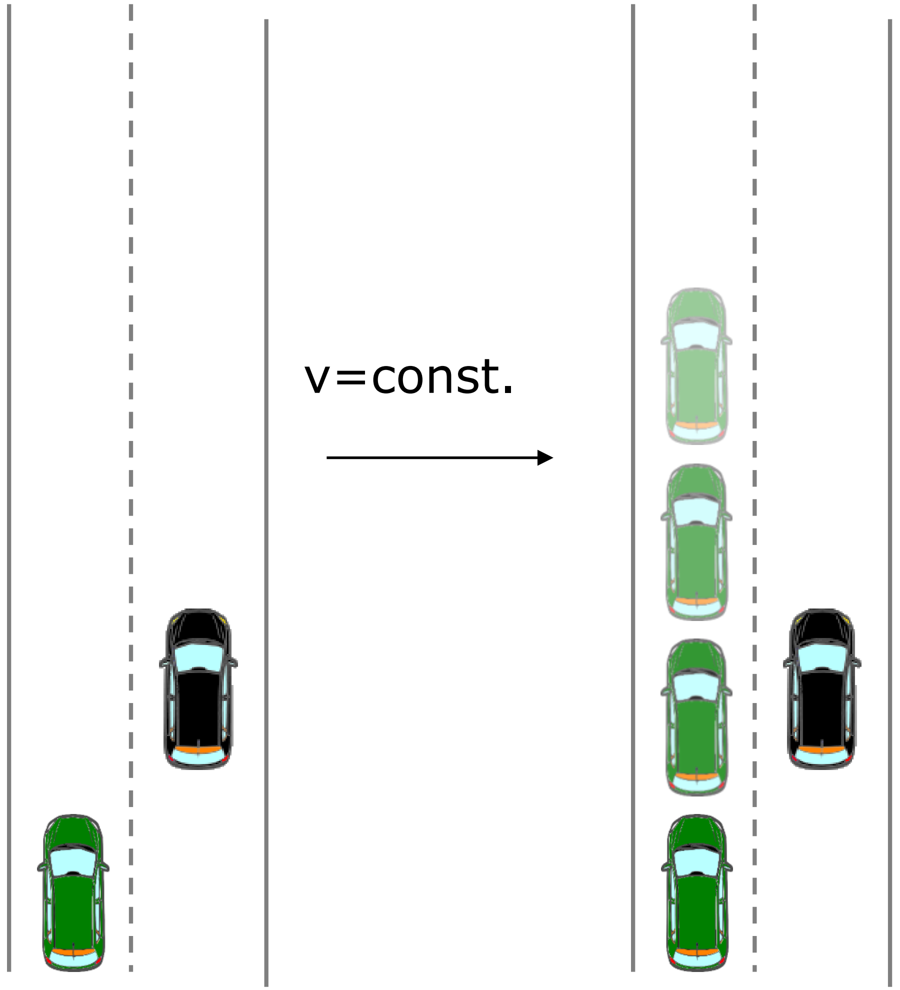
Interaction aware trajectory planning for automated vehicles

The Problem



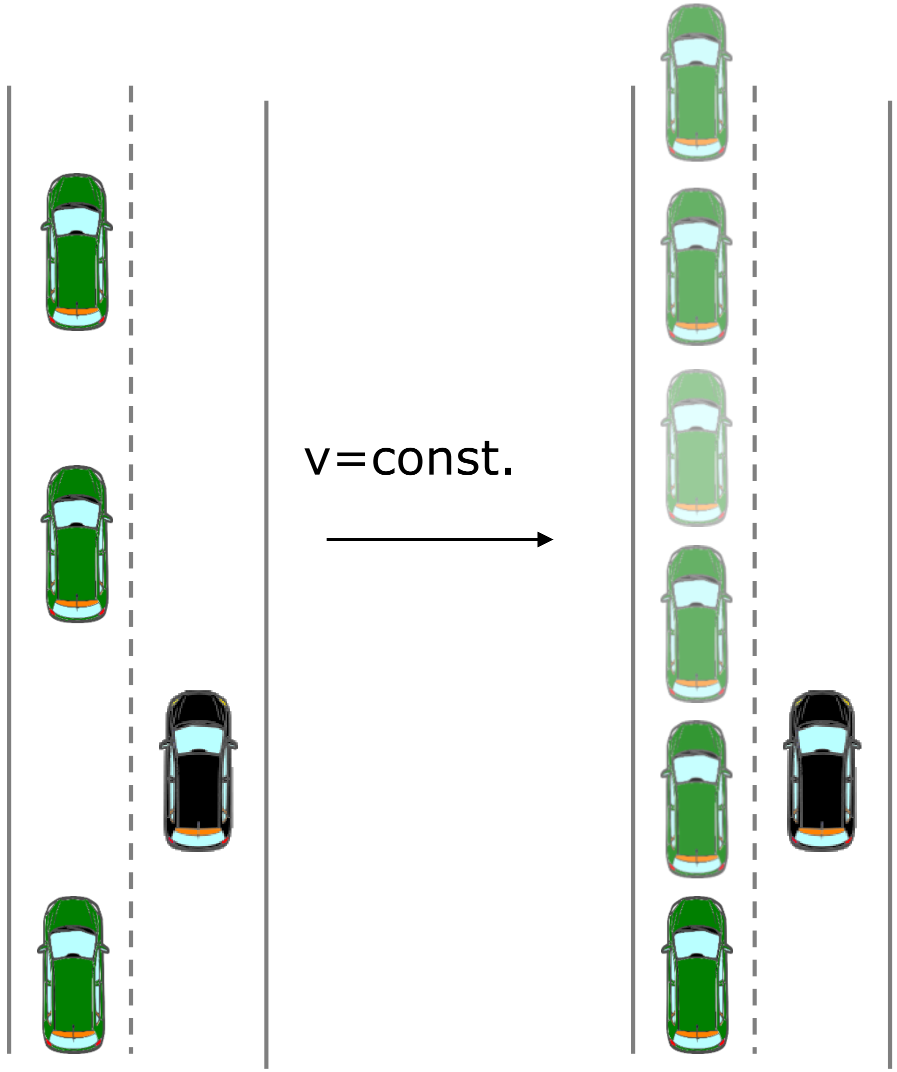
Interaction aware trajectory planning for automated vehicles

The Problem



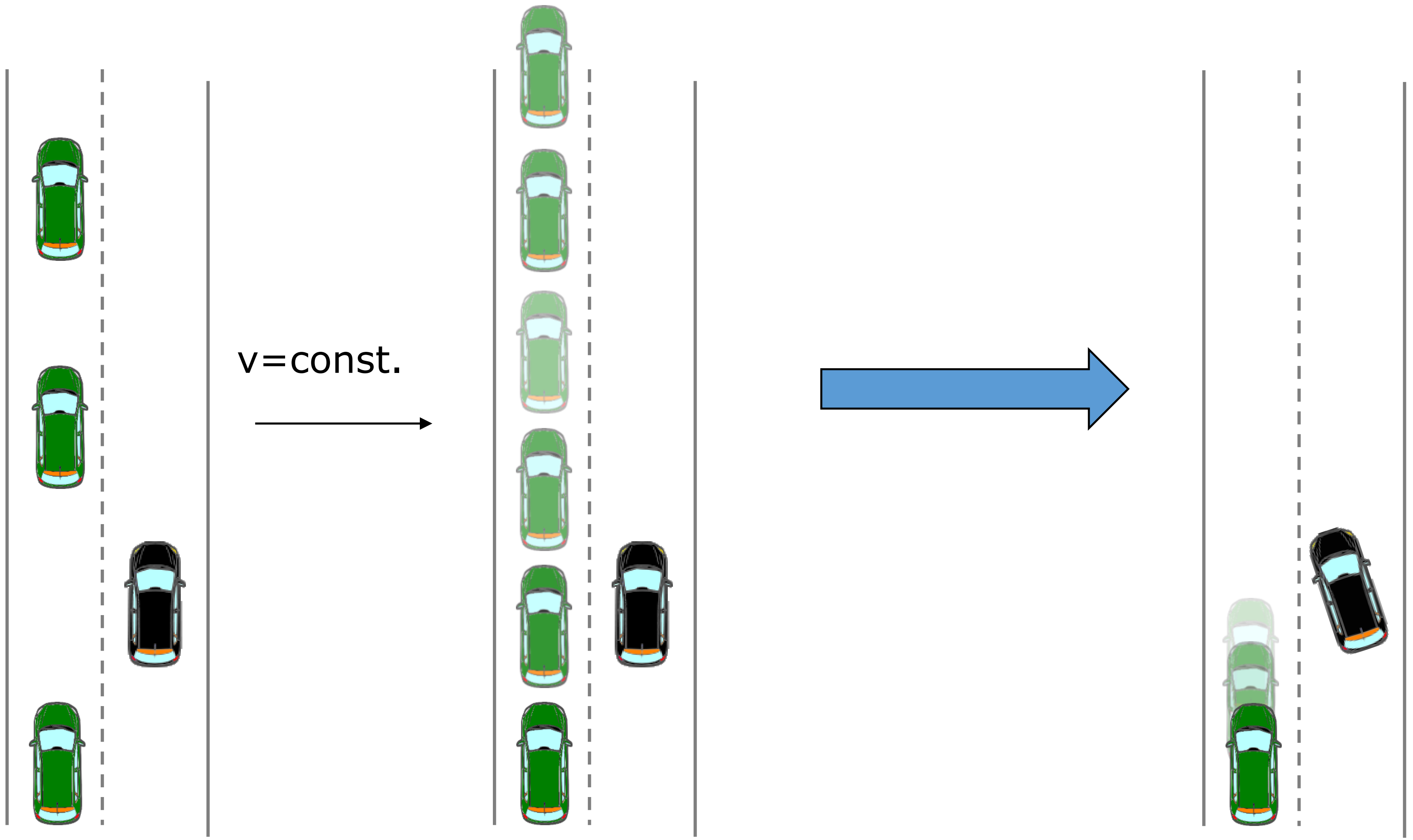
Interaction aware trajectory planning for automated vehicles

The Problem

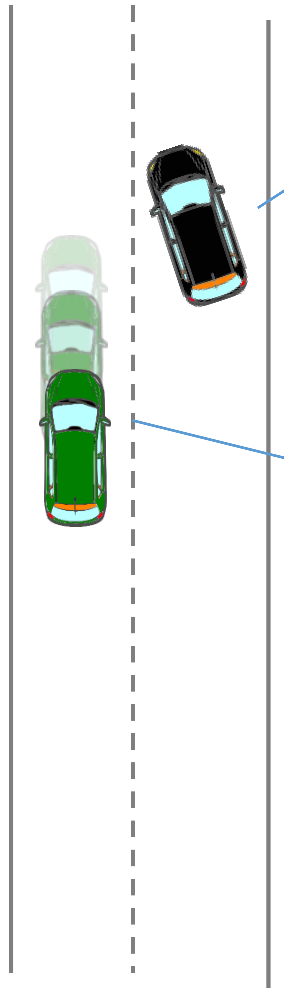


Interaction aware trajectory planning for automated vehicles

The Problem



Interaction aware trajectory planning for automated vehicles



$F(x,y)$

$G(x,y)$

$f(x,y)$

$g(x,y)$

Dynamic model

$$\vec{x} = \begin{bmatrix} x \\ y \\ \psi \\ v \end{bmatrix} \qquad f = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} v\cos(\psi+\beta) \\ v\sin(\psi+\beta) \\ \frac{v}{\ell}\tan(\delta)\cos(\beta) \\ a_x \end{bmatrix}$$

$$\boxed{\begin{aligned} \min\ & F(x,y) \\ \text{s.t.}\ & G(x,y) \leq 0 \\ & y \in \{\operatorname{argmin} f(x,y);\ g(x,y) \leq 0\} \end{aligned}}$$

MPCC

$$\min_{x,y,z,w} \|F^1(x,y)\|_2^2 + F^2(x,y) + \rho\, z^T \cdot w$$

$$\text{s.t.} \quad M \cdot s\bar{o} + G(x,y) \leq 0$$

$$0 = \underbrace{\nabla_y f(x,y) + \nabla_y g(x,y) \cdot z}_{:= H(x,y,z)}$$

$$0 \leq z \perp w \geq 0$$

$$0 = w + g(x,y)$$

Interaction aware trajectory planning for automated vehicles

Video

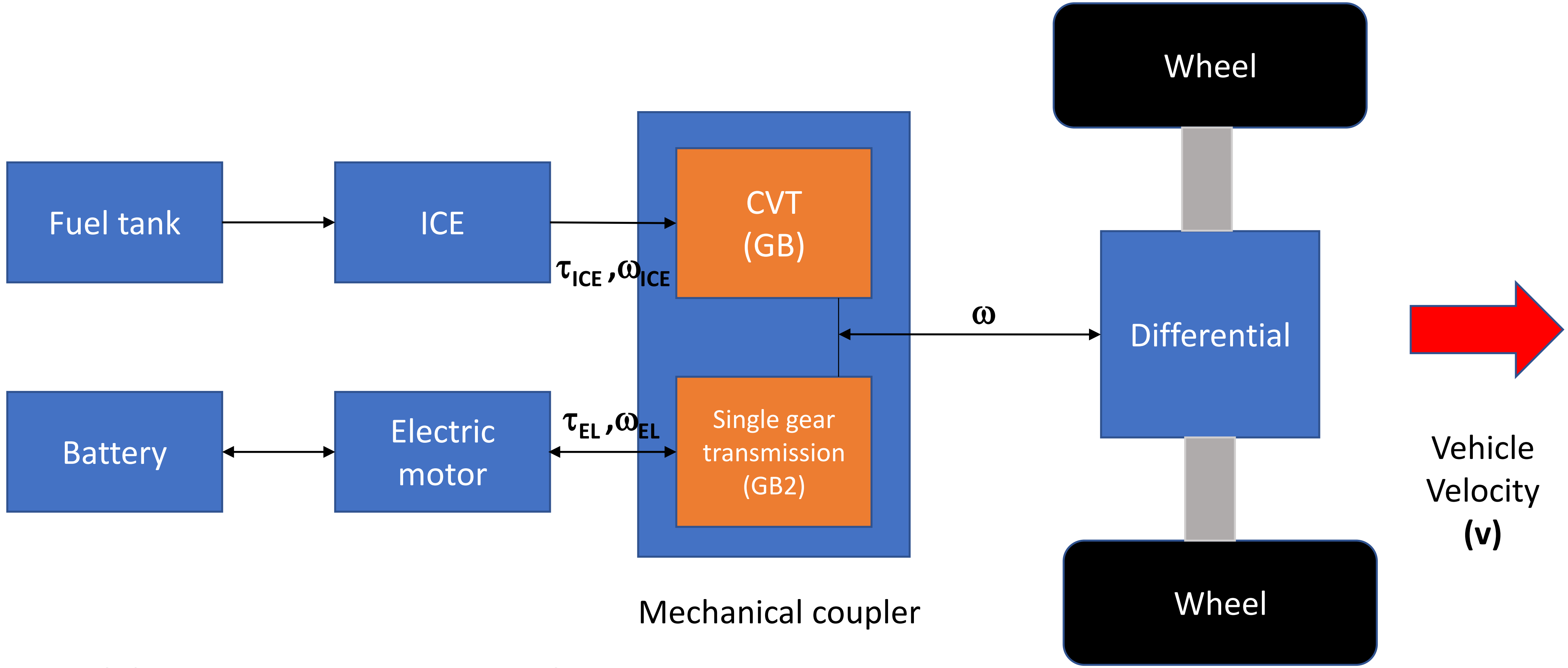
Hybrid powertrain optimization

Manuel Antonio Perez Estevez

mperezestevez@unibz.it

Freiburg, 04/08/2021

Parallel hybrid powertrain simplified model



How to control the system to minimize greenhouse emissions?

Application problem → **MPC**

Model

$$\frac{dv(t)}{dt} = \frac{\frac{\tau_{ICE(t)}}{r}GB(t) + \frac{\tau_{EL(t)}}{r}GB_2(t) - \frac{1}{2}\rho_{air}A_f C_d v(t) - (\mu cos(\alpha) + \sin(\alpha))(g\left(m_v + N_p N_s m_{cell} + \frac{P_p}{\rho_{ICE}}\right))}{m_v + N_p N_s m_{cell} + \frac{P_p}{\rho_{ICE}}}$$

$$\frac{dSOC(t)}{dt} = -100\frac{GB_2(t)v(t)\tau_{EL}(t)}{Cap_{cell}\, N_p\, N_s\, r\, \eta_{EL}(t)\, u(t)\, 3600}$$

$$\eta_{ICE}(t) = \eta_0(a_1 + b_1\left(\frac{v(t)}{n_p r}GB(t)\right) + c_1\left(\frac{v(t)}{n_p r}GB(t)\right)^2 + d_1\left(\frac{v(t)}{n_p r}GB(t)\right)^3)(a_2 + b_2(A) + c_2(A)^2 + d_2(A)^3)$$

Where:

$$A = \frac{v(t)GB(t)\tau_{ICE}(t)}{P_e(t)r}$$

$$Pe(t) = P_p\left(a_3\left(\frac{v(t)}{n_p r}GB(t)\right) + b_3\left(\frac{v(t)}{n_p r}GB(t)\right)^2 + c_3\left(\frac{v(t)}{n_p r}GB(t)\right)^3\right) + \varepsilon$$

$$\eta_{EL}(t) = \mathrm{LT}\,(\tau_{EL}, v(t))$$

$$u(t) = u_{nom}(a_4 + b_4(SOC(t)) + c_4(SOC(t))^2 + d_4(SOC(t))^3 + e_4(SOC(t))^4 + f_4(SOC(t))^5) \ldots - \mathrm{R}\frac{v(t)GB_2(t)\tau_{EL}(t)}{N_p N_s r\, u(t)\eta_{EL}(t)}$$

Differential states

$$x(t) = \begin{bmatrix} v(t) \\ SOC(t) \end{bmatrix}$$

Algebraic states

$$z(t) = \begin{bmatrix} \eta_{ICE}(t) \\ \eta_{EL}(t) \\ u(t) \end{bmatrix}$$

Control actions

$$a(t) = \begin{bmatrix} \tau_{ICE}(t) \\ \tau_{EL}(t) \\ GB(t) \end{bmatrix}$$

Optimization problem

$$\min_{\boldsymbol{x,z,a}} \frac{Coef_1}{r\ HV}\int_0^{t_{N-1}} \frac{v(t)GB(t)\tau_{\mathrm{ICE}}(t)}{(\eta_{\mathrm{ICE}}(t) + \varepsilon)}dt + (SOC_{\mathrm{target}} - SOC_N)Cap_{\mathrm{cell}}N_{\mathrm{p}}N_{\mathrm{s}}\, u_{\mathrm{nom}}Coef_2$$

subject to

$$f_{\mathrm{impl}}(x(t), z(t), a(t)) = 0$$

Constraints:

$v(t) = v_{ref}(t)$ $\quad$ $\tau_{\mathrm{ICE}}(t) \geq 0$ $\quad$ $0 \leq \eta_{\mathrm{EL}}(t) \leq 1$

$2.5 \leq u(t) \leq 4.3$ $\quad$ $-200 \leq \tau_{\mathrm{EL}}(t) \leq 180$ $\quad$ $0 \leq \eta_{\mathrm{ICE}}(t) \leq 1$

$0 \leq SOC(t) \leq 100$ $\quad$ $\frac{\tau_{\mathrm{ICE}}(t)v(t)GB(t)}{r} \leq P_{\mathrm{p}} * 1.15$ $\quad$ $0 \leq GB(t) \leq 7$

Differential states

$$x(t) = \begin{bmatrix} v(t) \\ SOC(t) \end{bmatrix}$$

Algebraic states

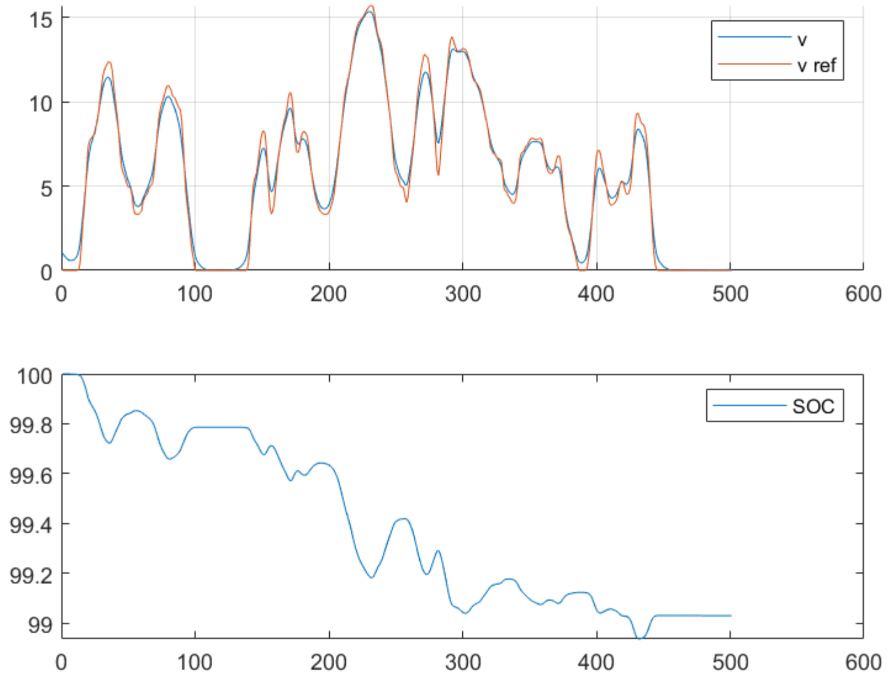
$$z(t) = \begin{bmatrix} \eta_{\mathrm{ICE}}(t) \\ \eta_{\mathrm{EL}}(t) \\ u(t) \end{bmatrix}$$

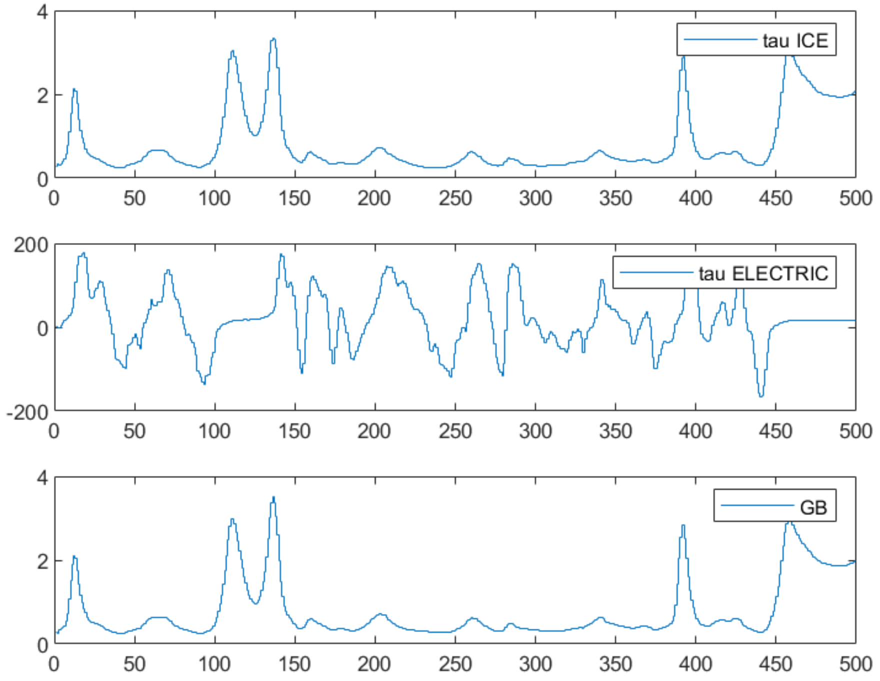
Control actions

$$a(t) = \begin{bmatrix} \tau_{\mathrm{ICE}}(t) \\ \tau_{\mathrm{EL}}(t) \\ GB(t) \end{bmatrix}$$

Preliminary results

$$\min_{x,z,a} \frac{Coef_1}{r\ HV} \int_0^{t_{N-1}} \left(\frac{v(t)GB(t)\tau_{ICE}(t)}{(\eta_{ICE}(t) + \ \varepsilon\ )} + \beta\ (v(t) - vref)^2 \right) dt$$





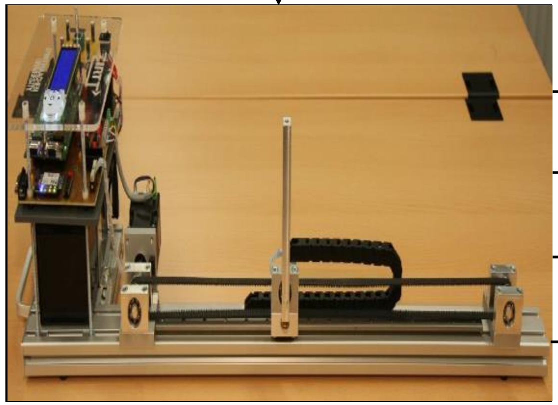
Control demonstrater cartpole with RL (TD3)

Manipulated Variable	
$\dot{x}_{target}$	Target velocity of speed controlled step motor

Challenges
Continuous state space
Continuous action space

$d?$

$\dot{x}_{target}(k)$



$x(k)$

$\dot{x}(k)$

$\varphi(k)$

$\dot{\varphi}(k)$

dT= 0.01 sec

Controlled Variable	
x	Cart position
$\dot{x}$	Cart velocity
φ	Pendulum angle
$\dot{\varphi}$	Angular velocitry of pendulum

Control Goal	Translates to
Swing up	-1
Maintain upright position	-1
Stay within available space	Barrier?

Ahmed Elsaid Elkhashap, Eugen Nuss, Rudolf Popp, Thuc Anh Nguyen

irt Institut für Regelungstechnik | RWTH AACHEN UNIVERSITY

Control Problem (Maintain upright position)

If stays in boundaries : $Reward\ = 1\ - x^2 + \cos(\theta)^2\ - \dot{x}^2 - \dot{\theta}^2$, **else** $Reward\ = -100$

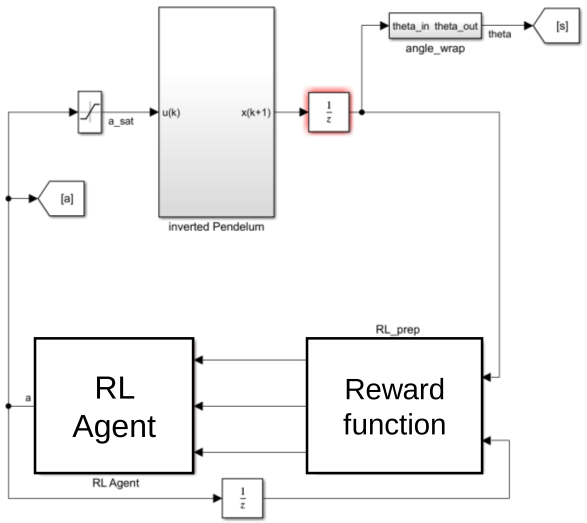
s.t. $$\frac{d}{dt}\begin{pmatrix} x \\ \dot{x} \\ \varphi \\ \dot{\varphi} \end{pmatrix} = \begin{pmatrix} \dot{x} \\ \dfrac{K\ \cdot \dot{x}_{target} - \dot{x}}{T} \\ \dot{\varphi} \\ \dfrac{m \cdot g \cdot \mathrm{L} \cdot \sin(\varphi)}{2 \cdot J} - \dot{x} \cdot \cos(\varphi) - \mu \cdot \dot{\varphi} \end{pmatrix}$$

$-0.25\ m < x_{min} < 0.25\ \mathrm{m}$

$-12\,° < \varphi < 12\,°$

<u>Procedure</u>

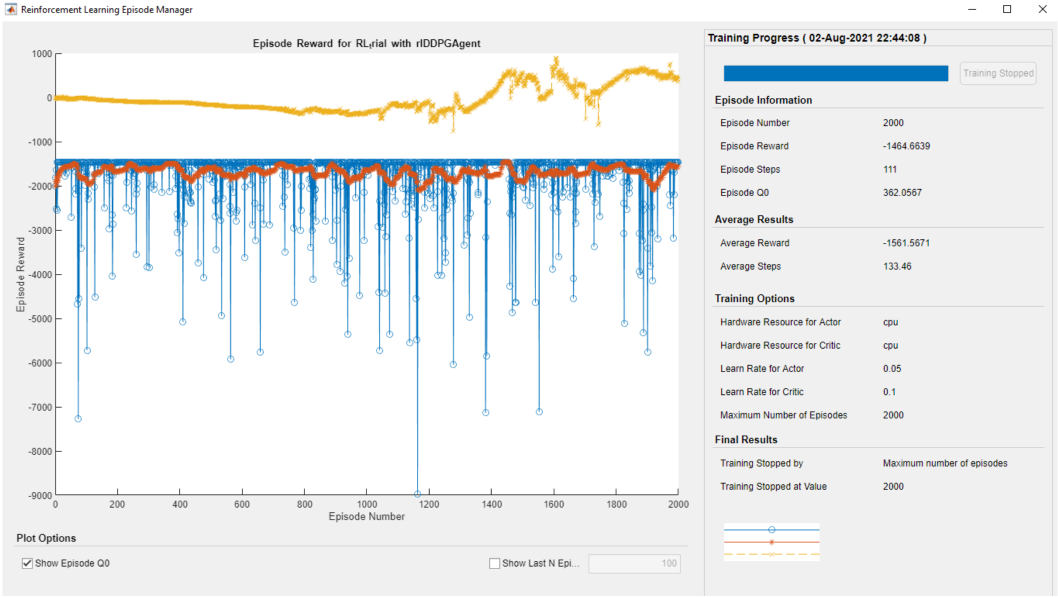
- Matlab RL Toolbox
  - *Training time > 3 h, not converged*



- *Baseline3: Pythorch implementation RL*
  - *training time 10 Min*
  - *train cardpole gym environmet (exercise 8)*
  - *train IRT gym env -> not successful yet*
  - *export agent Neuronal Network to Matlab to control demonstrater -> to do*

Training in Simulink

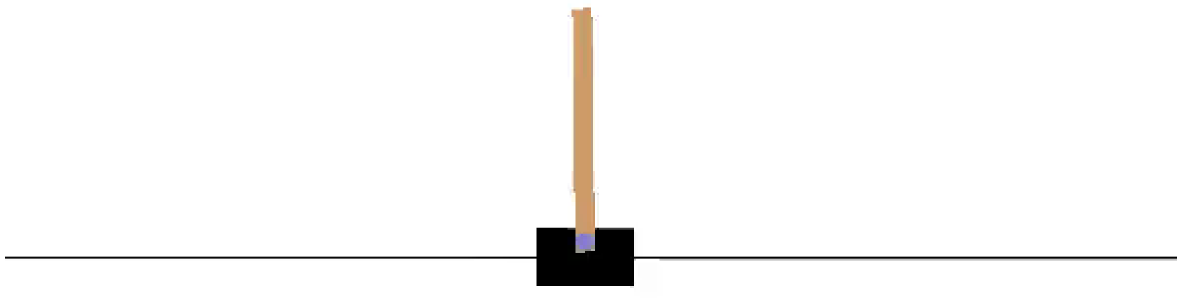
DDPG



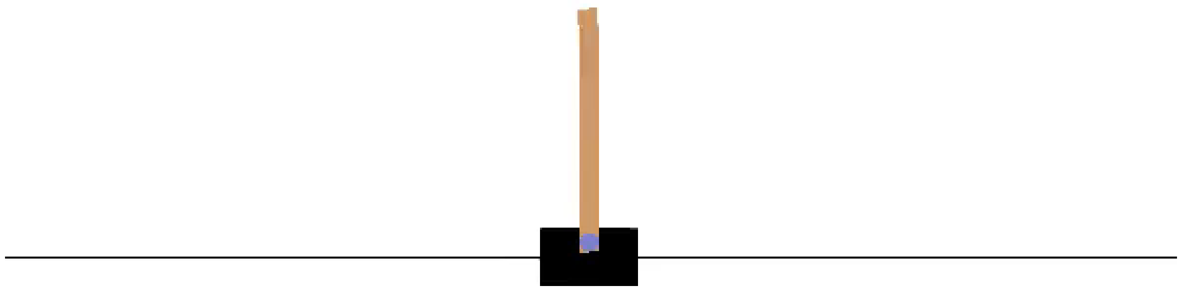
TD3

Some results (1)

$$Reward = 1$$

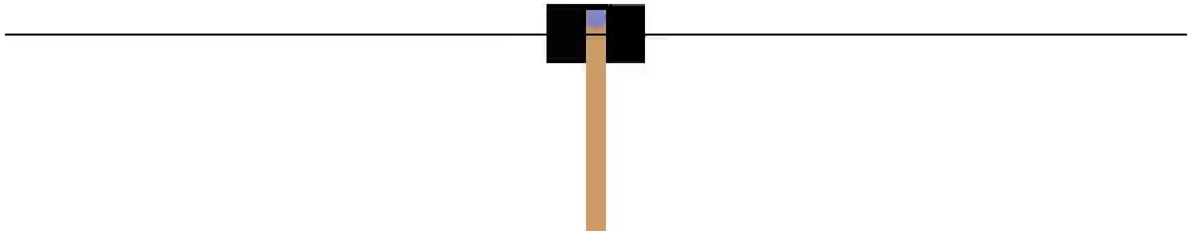


$$Reward = 1 - x^2 + \cos(\theta)^2 - \dot{x}^2 - \dot{\theta}^2$$



Some results (2)

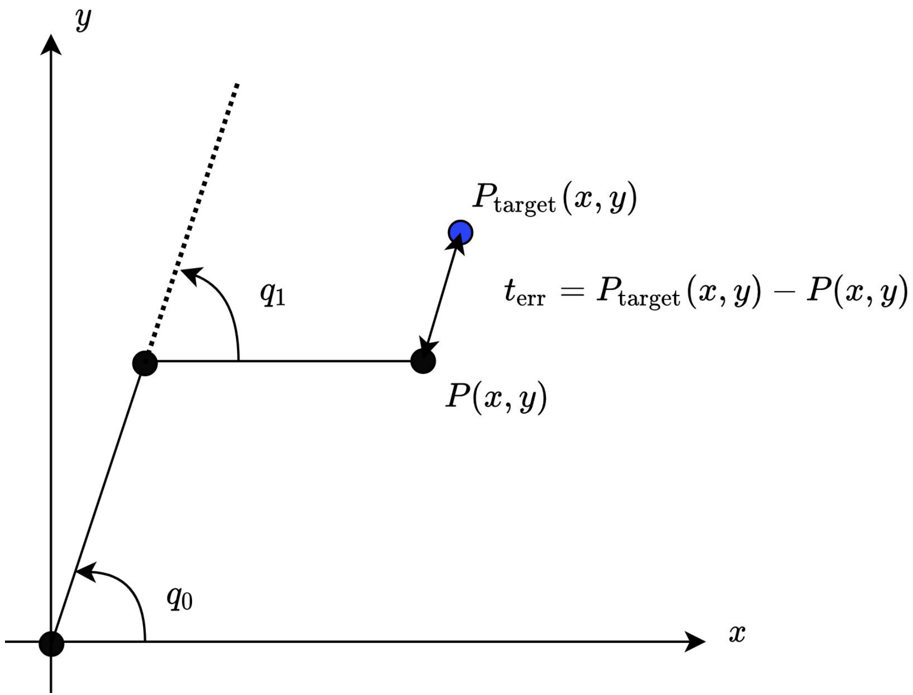
$$Reward = 1 - x^2 + \cos(\theta)^2 - \dot{x}^2 - \dot{\theta}^2$$
$$s_0 = (0,0,\pi,0)^T$$



Learning point to point motions for robot manipulators using RL

Johan & Ruan

Problem formulation



State space:

$$s = [q_0, q_1, \dot{q}_0, \dot{q}_1]$$
$$q_0, q_1 \in (-\pi, \pi]$$
$$\dot{q}_0, \dot{q}_1 \in [-8, 8]$$

Action space:

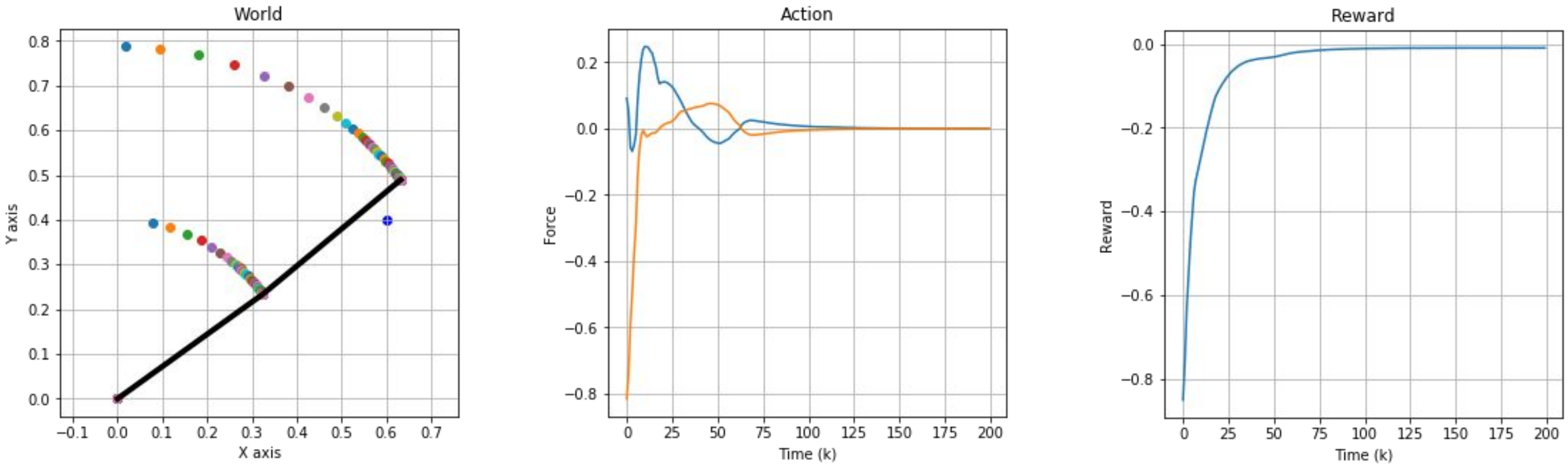
$$\text{Continuous: } \ddot{q}_0, \ddot{q}_1 \in (-1.0, 1.0]$$

Reward:

$$r = -(||t_{\text{err}}||_2^2 + 0.5||a||_2^2)$$

Results

Continuous: Soft Actor Critic



References:

- Soft Actor Critic algorithm: https://stable-baselines.readthedocs.io/en/master/modules/sac.html

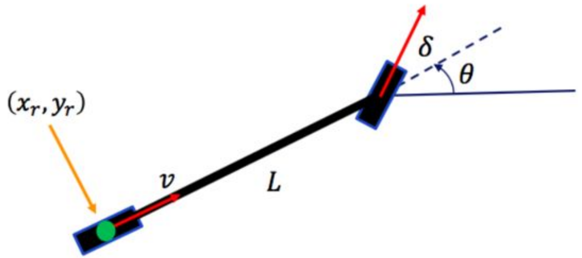
Mobile robot parking with obstacles using RL

Federico Ulloa Rios

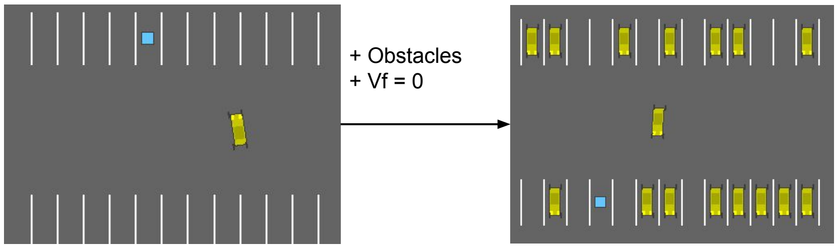
Santiago Iregui Rincón

Problem Formulation

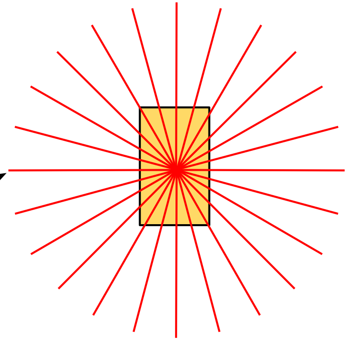
Deterministic dynamics (Bicycle model)



Modified GYM Environment



$$\boldsymbol{s} = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \\ \theta \\ \boxed{\boldsymbol{L}_d} \end{bmatrix}$$



$$\boldsymbol{L}_d = \text{Lidar data}$$

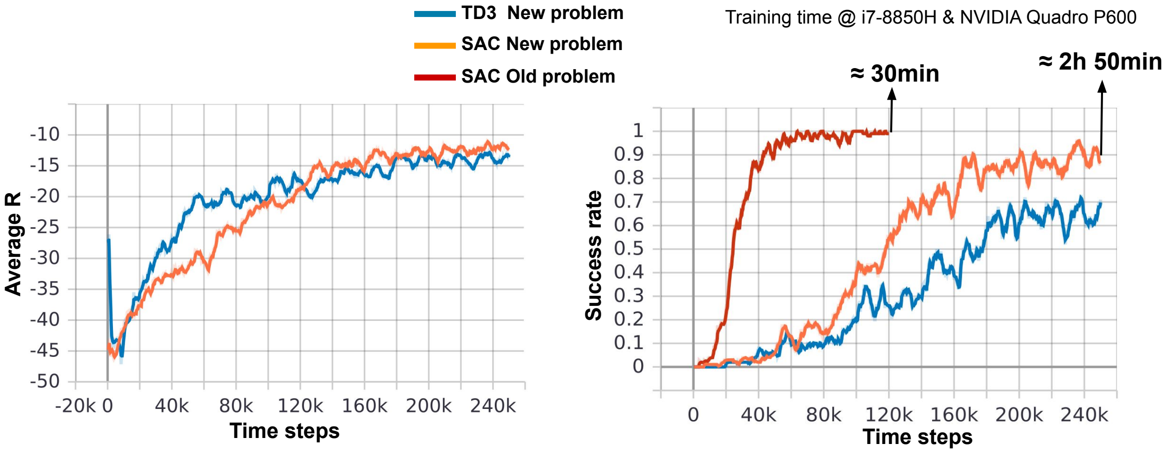
$$\boldsymbol{a} = \begin{bmatrix} \alpha \\ \delta \end{bmatrix}$$

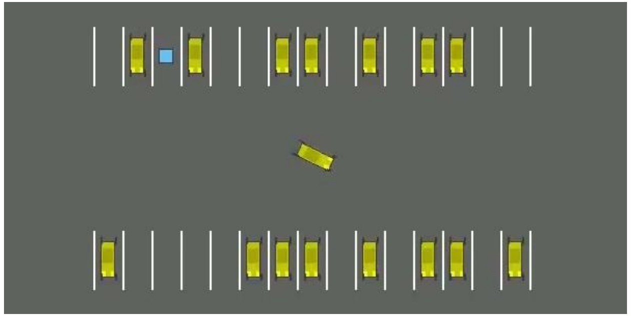
$$R = -(|s_r - s_t| \cdot w)^p - 5 * \psi$$

- Continuous action and state space
- Off-policy learning actor critic methods (SAC & TD3)

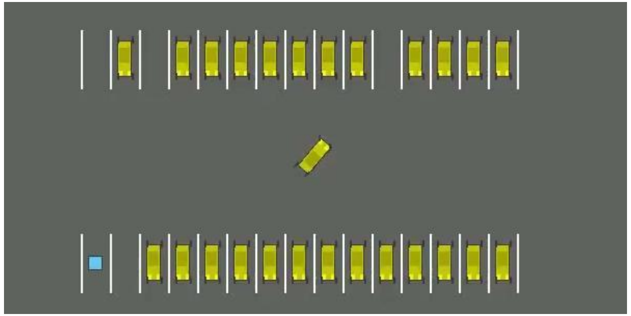
Results

SAC Old problem





SAC New problem



Model Predictive Control and Reinforcement Learning

Quadrotor Control using MPC

Date: **04.08.2021**

Sourish Pramanick
Niket Ahuja
Nayana Koneru

Problem Formulation

$$s = [x, y, z, \phi, \theta, \psi, u, v, w, p, q, r]^{\mathrm{T}}$$

$$u = [f_t, \tau_x, \tau_y, \tau_z]^{\mathrm{T}}$$

$$C(s, a) = \sum_{k=0}^{N-1} (s_k - s_{ref,k})^T Q (s_k - s_{ref,k}) + a^{\mathrm{T}} R a$$

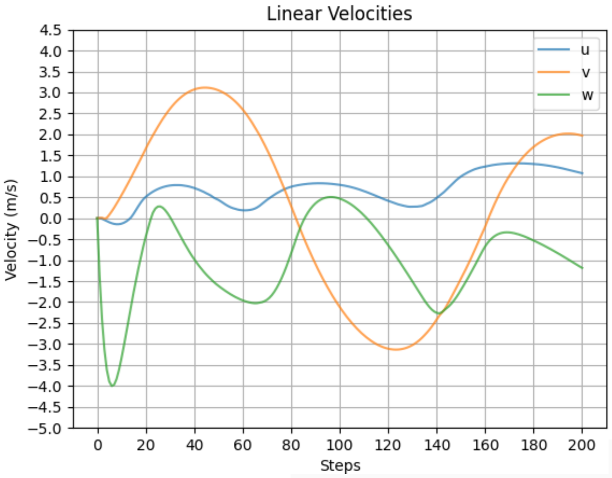
$$\dot{\phi} = p + r[c(\phi)t(\theta)] + q[s(\phi)t(\theta)]$$
$$\dot{\theta} = q[c(\phi)] - r[s(\phi)]$$
$$\dot{\psi} = r\frac{c(\phi)}{c(\theta)} + q\frac{s(\phi)}{c(\theta)}$$
$$\dot{p} = \frac{I_y - I_z}{I_x} rq + \frac{\tau_x + \tau_{wx}}{I_x}$$
$$\dot{q} = \frac{I_z - I_x}{I_y} pr + \frac{\tau_y + \tau_{wy}}{I_y}$$
$$\dot{r} = \frac{I_x - I_y}{I_z} pq + \frac{\tau_z + \tau_{wz}}{I_z}$$
$$\dot{u} = rv - qw - g[s(\theta)] + \frac{f_{wx}}{m}$$
$$\dot{v} = pw - ru + g[s(\phi)c(\theta)] + \frac{f_{wy}}{m}$$
$$\dot{w} = qu - pv + g[c(\theta)c(\phi)] + \frac{f_{wz} - f_t}{m}$$
$$\dot{x} = w[s(\phi)s(\psi) + c(\phi)c(\psi)s(\theta)] - v[c(\phi)s(\psi) - c(\psi)s(\phi)s(\theta)] + u[c(\psi)c(\theta)]$$
$$\dot{y} = v[c(\phi)c(\psi) + s(\phi)s(\psi)s(\theta)] - w[c(\psi)s(\phi) - c(\phi)s(\psi)s(\theta)] + u[c(\theta)s(\psi)]$$
$$\dot{z} = w[c(\phi)c(\theta)] - u[s(\theta)] + v[c(\theta)s(\phi)]$$

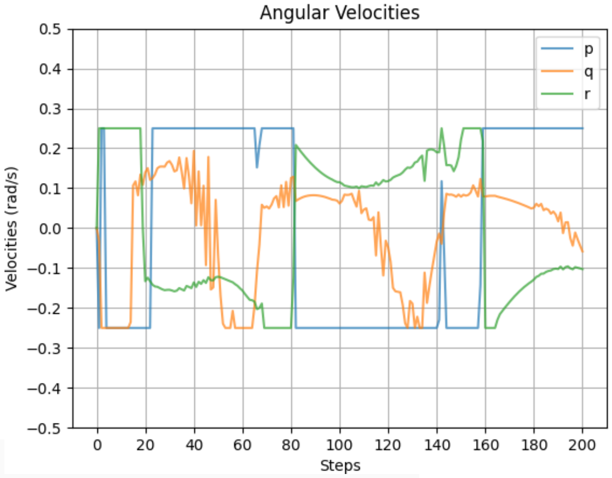
$$\min_{s,a} \sum_{k=0}^{N-1} c(s_k, ak) + E(sN)$$

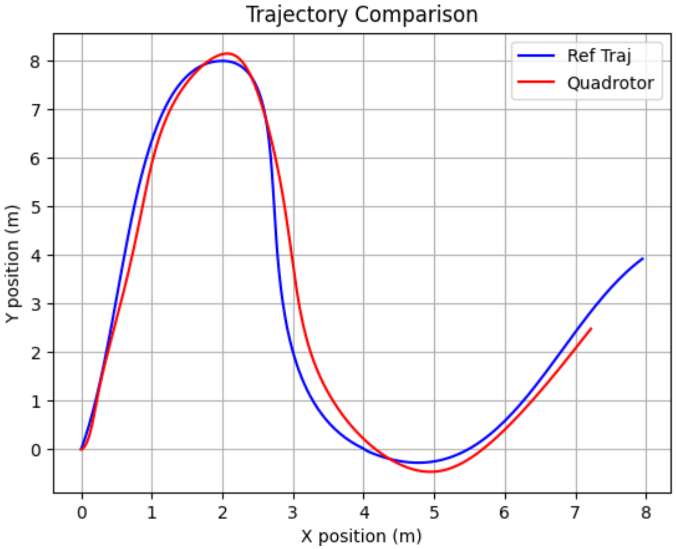
$$s_0 = \bar{s_0}$$
$$s_{k+1} = f(s_k, ak)$$

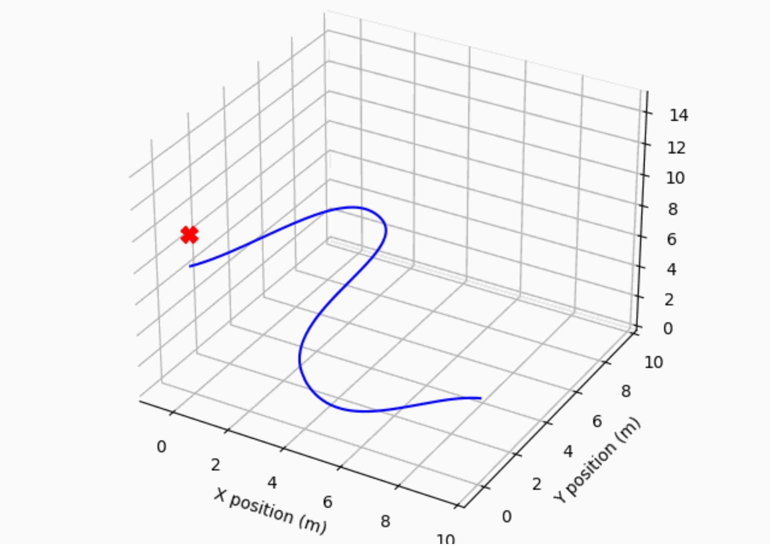
$$-\pi \leq \phi \leq \pi$$
$$-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}$$
$$-\pi \leq \psi \leq \pi$$
$$-0.25 \leq \dot{\phi} \leq 0.25$$
$$-0.25 \leq \dot{\theta} \leq 0.25$$
$$-0.25 \leq \dot{\psi} \leq 0.25$$

Results


Linear Velocities
u
v
w
Velocity (m/s)
Steps
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−3.0
−3.5
−4.0
−4.5
−5.0
0
20
40
60
80
100
120
140
160
180
200


Angular Velocities
p
q
r
Velocities (rad/s)
Steps
0.5
0.4
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.4
−0.5
0
20
40
60
80
100
120
140
160
180
200


Trajectory Comparison
Ref Traj
Quadrotor
Y position (m)
X position (m)
8
7
6
5
4
3
2
1
0
0
1
2
3
4
5
6
7
8


X position (m)
Y position (m)
0
2
4
6
8
10
0
2
4
6
8
10
0
2
4
6
8
10
12
14

Reinforcement Learning for Airborne Wind Energy Power Optimization

Jochem De Schutter and Jasper Hoffmann

Systems Control and Optimization Laboratory, ALU Freiburg

August 4, 2021

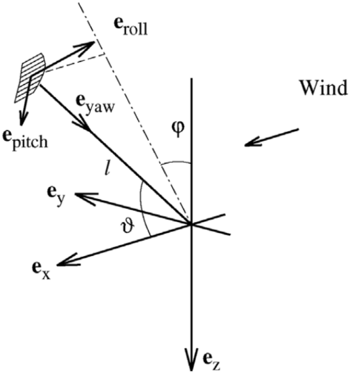
Simulation model

- system dynamics ($n_\mathrm{s} = 3$, $n_\mathrm{a} = 1$):

$$\dot{\psi} = g_\mathrm{k} v_\mathrm{a} \delta + \dot{\phi} \cos\theta$$
$$\dot{\phi} = -\frac{v_\mathrm{a}}{l \sin\theta} \sin\psi$$
$$\dot{\theta} = -\frac{v_\mathrm{w}}{l} \sin\theta + \frac{v_\mathrm{a}}{l} \cos\theta$$

- economic cost (“max. pulling force")

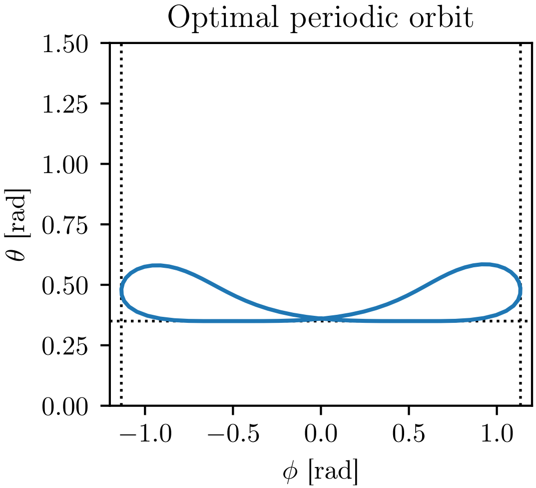
$$c(s,a) := -C_\mathrm{R} v_\mathrm{a}^2 + \alpha \dot{\psi}^2$$
$$v_\mathrm{a} := v_\mathrm{w} E \cos\theta$$



M. Erhard, G. Horn, M. Diehl, A quaternion-based model for optimal control of an airborne wind energy system, Z. Angew. Math. Mech. 97, No. 1, 7–24 (2017)

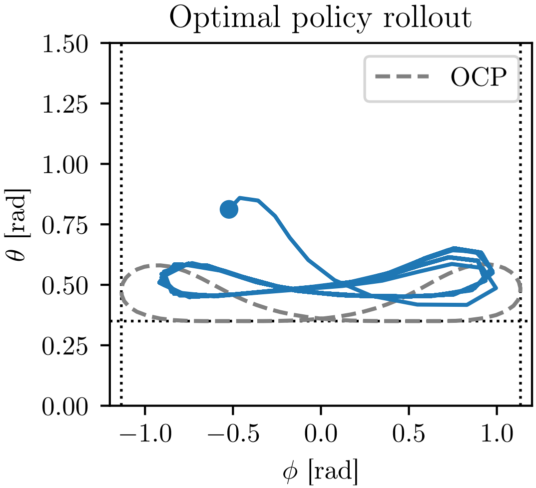
Periodic optimal control

$$
\begin{aligned}
\underset{s(\cdot),a(\cdot),T}{\text{minimize}} \quad & \frac{1}{T}\int_0^T c(s(t),a(t))\mathrm{d}t \\
\text{subject to} \quad & \dot{s} = f(s(t),a(t)), \quad t \in [0,T], \\
& s(0) = s(T), \\
& h(s(t),a(t)) \geq 0, \quad t \in [0,T]
\end{aligned}
$$



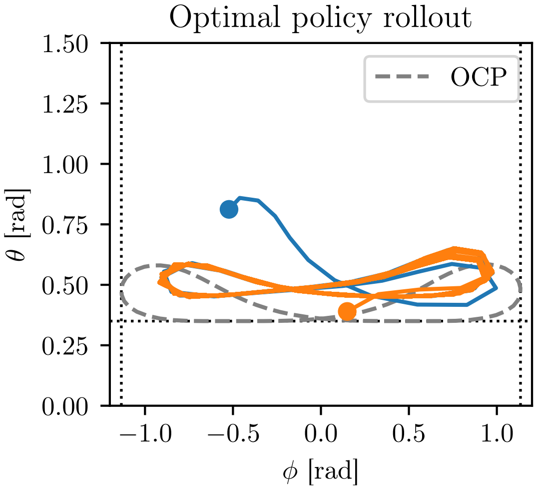
Reinforcement learning

- Penalty-based constraint formulation (gym)
- Proximal policy optimization (`stable-baselines3`)
- 10M time-steps, $t_{\text{learn}} \approx 5\text{h}$.
- $|\bar{c}_{\text{RL}}| = 21.5\text{kN} \approx 23.8\text{kN} = |\bar{c}_{\text{OCP}}|$

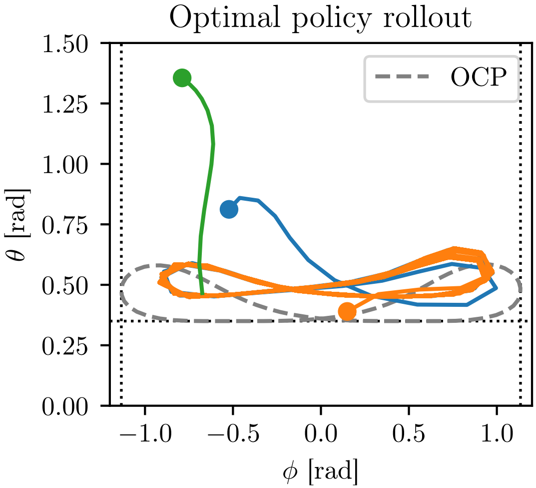


Reinforcement learning

- Penalty-based constraint formulation (gym)
- Proximal policy optimization (`stable-baselines3`)
- 10M time-steps, $t_{\text{learn}} \approx 5\text{h}$.

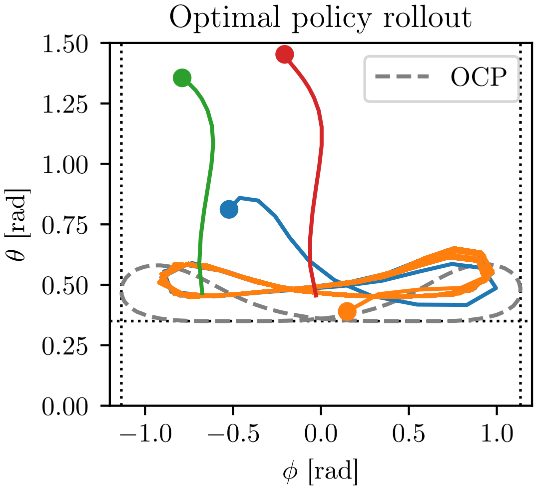


- Penalty-based constraint formulation (gym)
- Proximal policy optimization (`stable-baselines3`)
- 10M time-steps, $t_{\text{learn}} \approx 5$h.



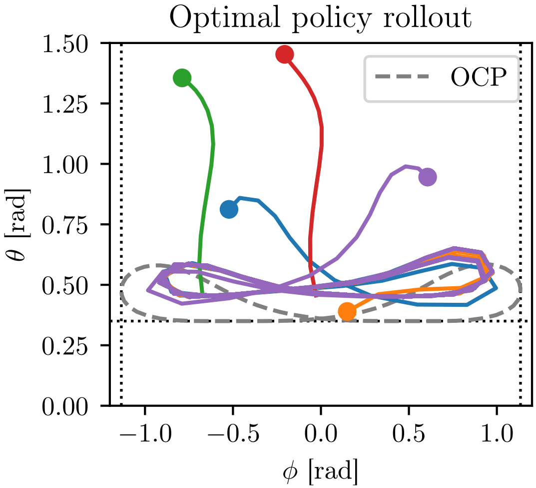
Reinforcement learning

- Penalty-based constraint formulation (`gym`)
- Proximal policy optimization (`stable-baselines3`)
- 10M time-steps, $t_{\text{learn}} \approx 5$h.



Reinforcement learning

- Penalty-based constraint formulation (`gym`)
- Proximal policy optimization (`stable-baselines3`)
- 10M time-steps, $t_{\text{learn}} \approx 5$h.

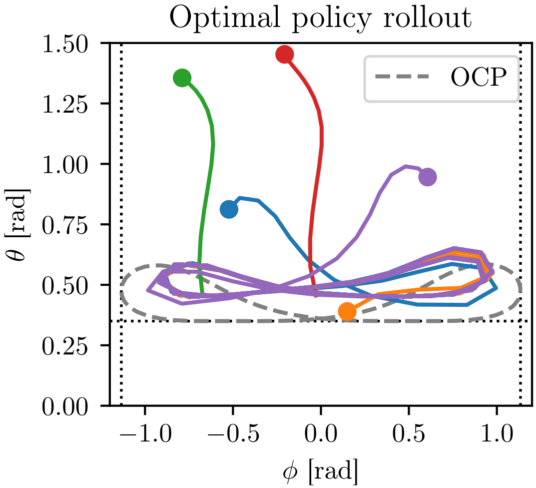


Reinforcement learning

- Penalty-based constraint formulation (`gym`)
- Proximal policy optimization (`stable-baselines3`)
- 10M time-steps, $t_{\text{learn}} \approx 5$h.

Future ideas:

- compare to economic MPC $\pi(s)$
- learn stochastic wind model



Point-Mass Racing Game

MPC and RL Course 2021

Christian Leininger, Rudolf Reiter

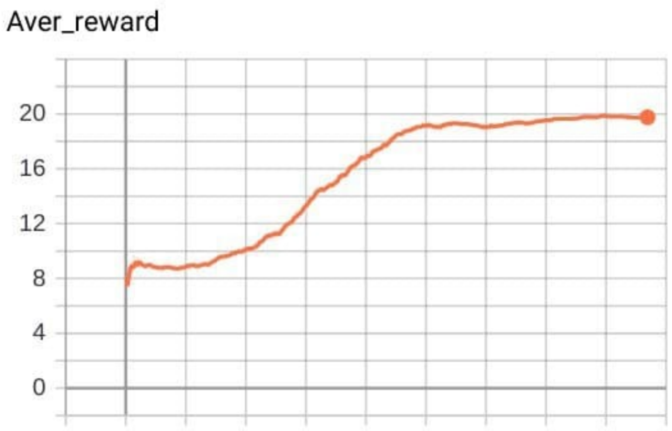
Background Idea

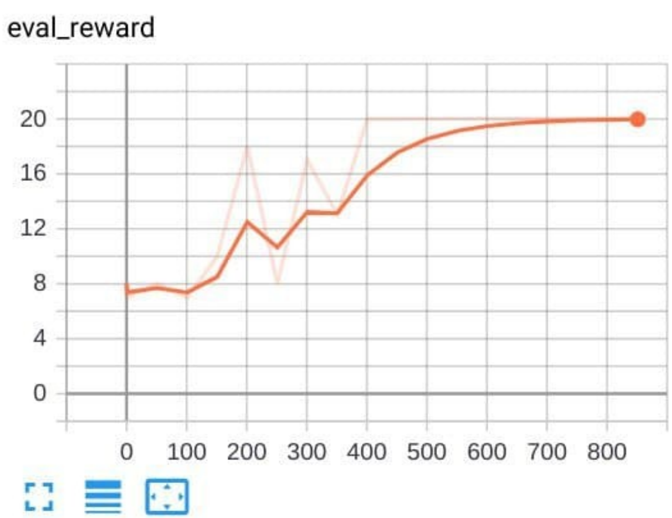
- **Game: Two cars: The ego car has better starting position, the opponent car has a higher velocity limit.**
- **Goal: Preventing the opponent car from overtaking, while still racing and avoiding obstacles.**
- **Design idea: Restrict the action space of an RL – policy to be safe by choosing the actions to influence only the cost of an MPC, that satisfies constraints.**
- **Basically the MPC “hides” inside the environment**
- **The MPC cost consists of a parameterizable part and a fixed part**
  - Parameterizable part: Linear plane in the position states of the vehicle
  - Fixed part: Quadratic cost to keep close to the middle part
- **The opponent agent is simulated by just the MPC policy with a fixed cost function**

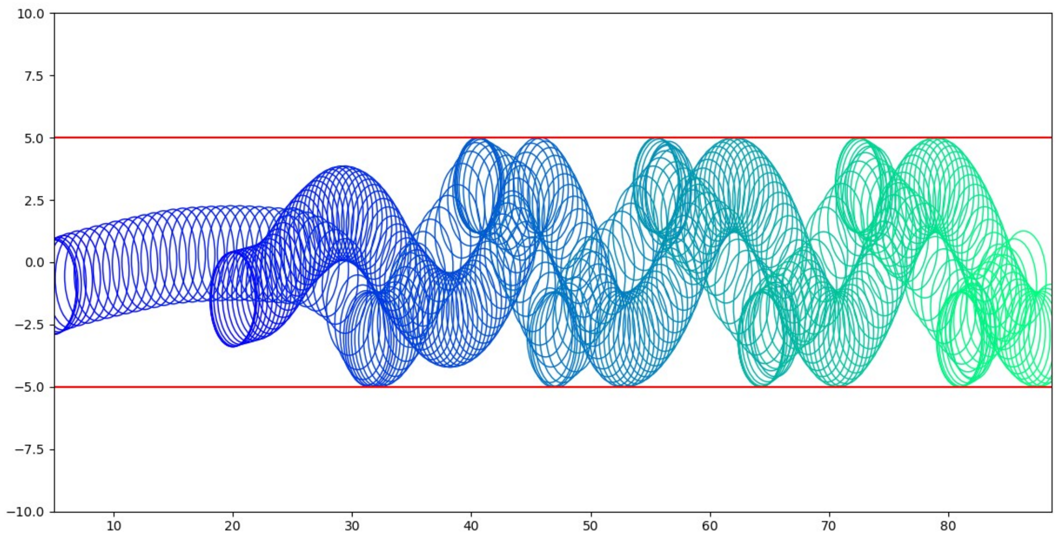
Implementation

- **MPC:**
  - “Hidden” in the RL environment
  - CasADi, to optimize for a trajectory
  - Perfect tracking is pretended
  - Trajectory followed for some steps until the RL policy changes the cost function
  - Circular shaped cars (point-masses) to keep it simple
  - Slack variables to stay feasible
  - Includes simple state estimation of the other agents trajectory (Not known to each other!)
- **RL:**
  - TDQ Truncated Mixture of Continuous Distributional Quantile Critics
  - Computes quantiles of Q-value Function
  - Control over and underestimation

Results







Works as expected!

Generalizes even to minor parameter changes. (positions, bounds, dimension)

“Guaranteed safety” (to some degree)

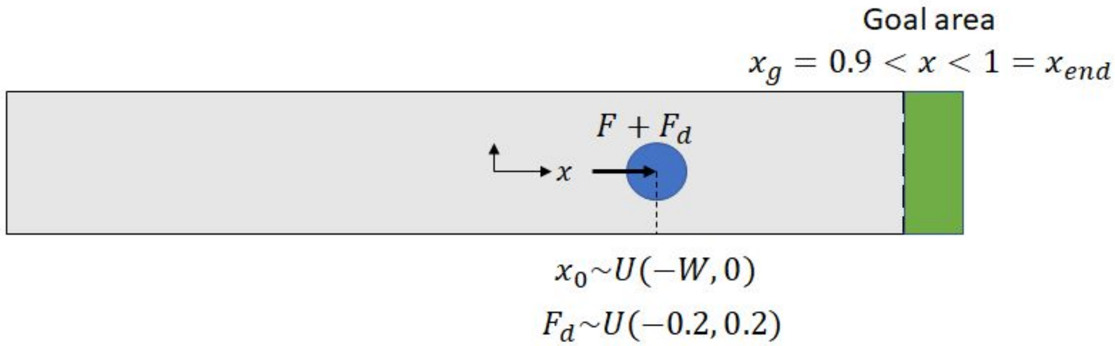
Stability might be an issue for a tracking controller

Thank you for your attention! Questions?

Non-prehensile Table Top Manipulation

Flavia Acerbo, Tommaso Sartor, Ajay Sathya

Problem Formulation



$$s = x$$

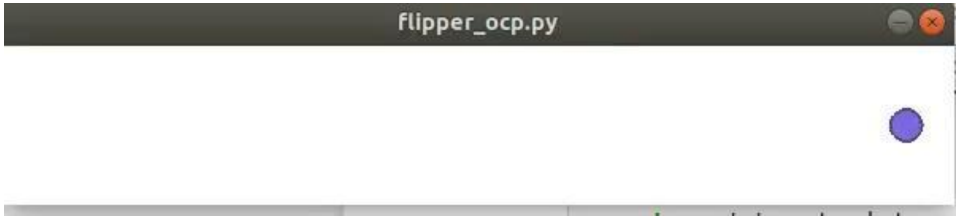
$$a = F \ \in [0, 10]$$

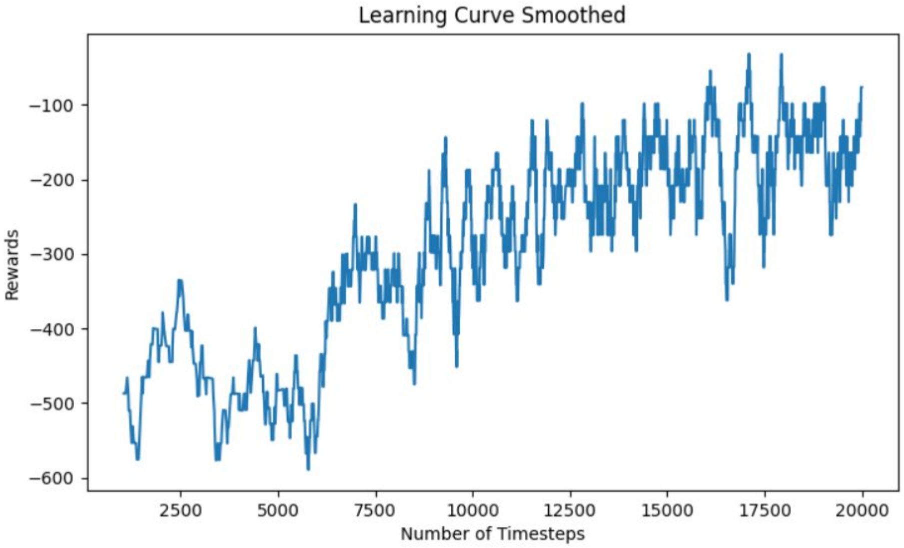
$$r_t = \begin{cases} 100, & if \quad x_g < x < x_{end} \\ -1000, & if \quad x > x_{end} \\ x - x_g & else \end{cases}$$

A disk lies flat on a surface (0.7 friction) at a random initial position.
The disk can be pushed on the x-axis with an impulse (continuous force amplitude) at each time step.
We observe the next position after the disk has come to rest.
A stochastic disturbance modifies the force that is applied to disk.
The goal is to make the disk reach the border of the surface (where it is possible to be grasped), without making it fall.

Results

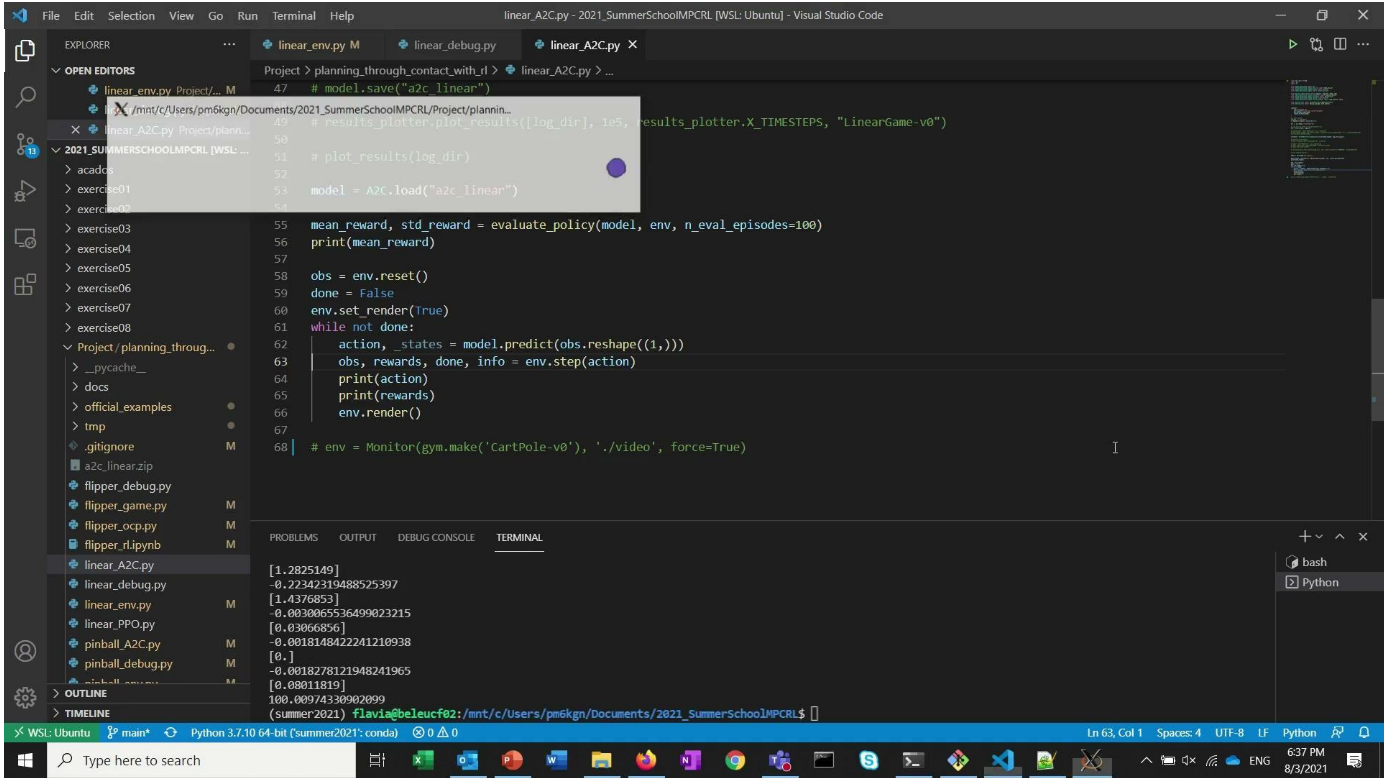
- Implemented two approaches: MPC and RL.
- A2C algorithm: it learns to take big actions when away from the border and more cautious actions when close (where disturbance can be more influential)
- Using MPC, the controller failed only in case of large model discrepancy.
- Expected reward:
    - MPC policy 98.45
    - A2C policy: 99.41
    - Random policy: -871





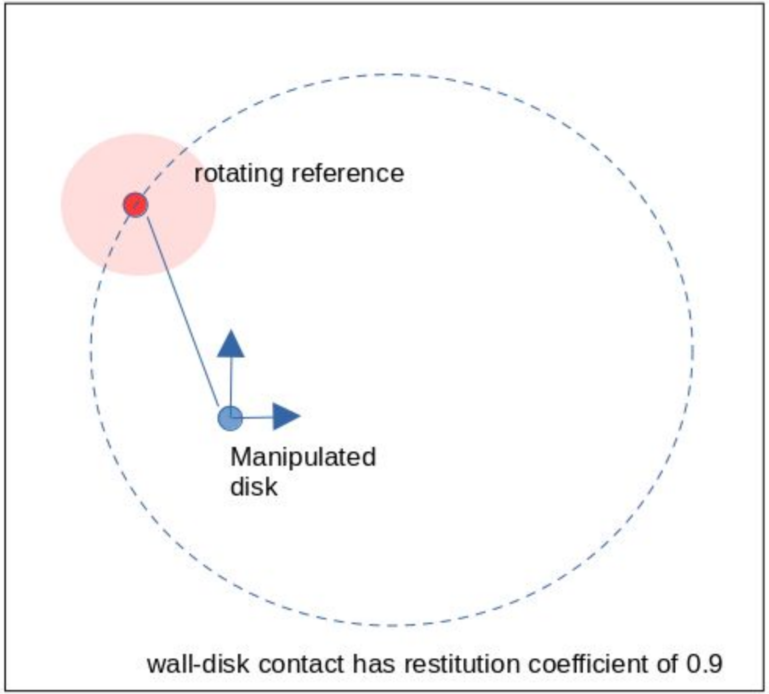
Extra slides

A2C rollout



linear_A2C.py - 2021_SummerSchoolMPCRL [WSL: Ubuntu] - Visual Studio Code
model = A2C.load("a2c_linear")
mean_reward, std_reward = evaluate_policy(model, env, n_eval_episodes=100)
print(mean_reward)
obs = env.reset()
done = False
env.set_render(True)
while not done:
action, _states = model.predict(obs.reshape((1,)))
obs, rewards, done, info = env.step(action)
print(action)
print(rewards)
env.render()
# env = Monitor(gym.make('CartPole-v0'), './video', force=True)

Tracking time-varying reference



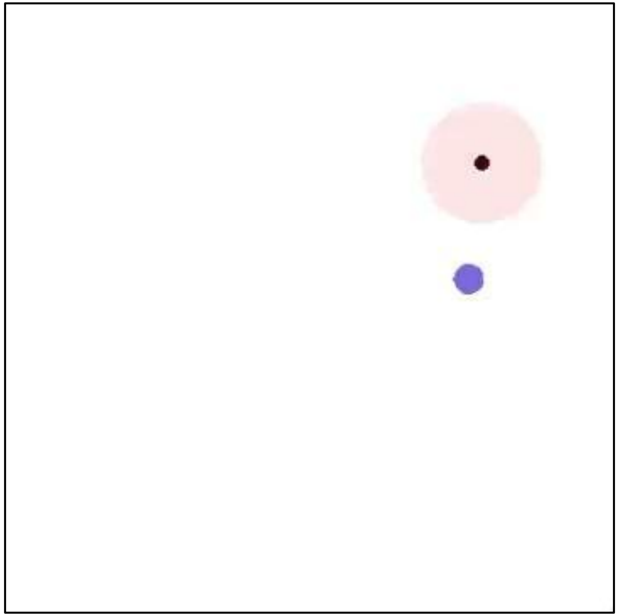
Reward is a fuction of distance
between reference and manipulated disk

Episode is **completed** if disk is inside reference
shaded area for a given number of steps

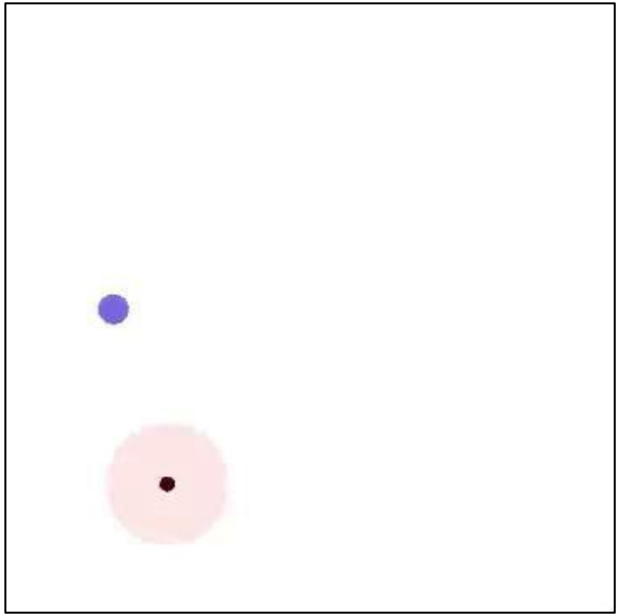
Action space is a discrete 9x9 space
4 level of force in two plus no-op for each cartesian axis,
a pair of force fx, fy is applied at each step

Chasing reference - videos

heuristic



a2c policy

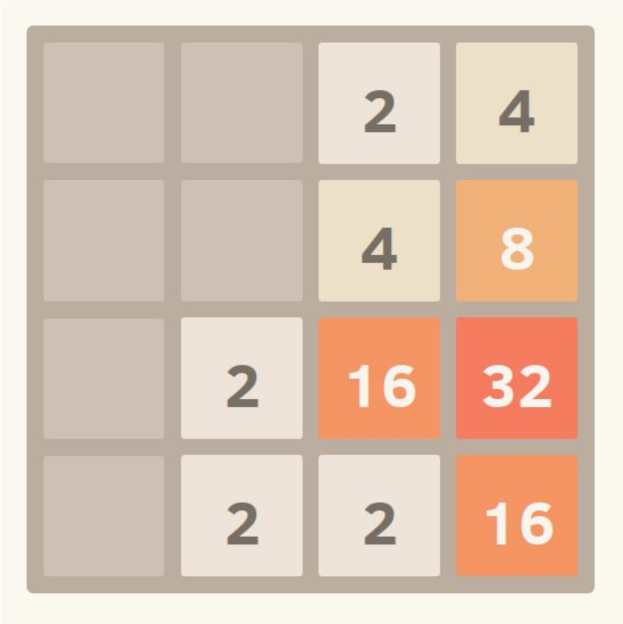


RL approaches to the game 2048

Project by Hoang Dang & Mario Kantz

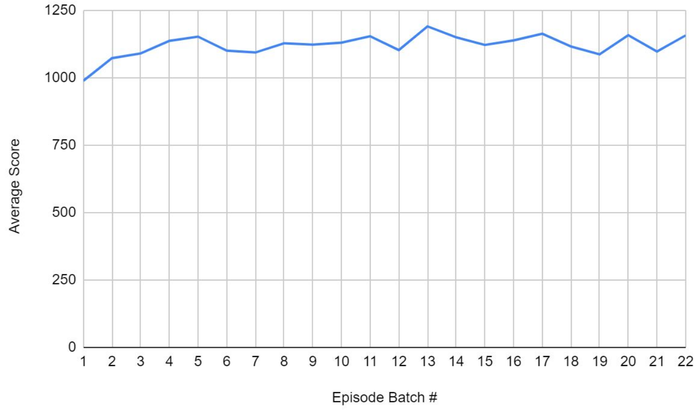
The Game 2048

- Solitaire game on 4x4 square grid
- Player can shove all tiles into a cardinal direction, merging equal tiles to their sum
- Score is the sum of values of merged tiles
- Between each player action, another tile is spawned (90% for 2, 10% for 4)
- Player may only shove into a direction if that moves or merges at least one tile
- Game ends when no more moves are possible
- Termination is thus guaranteed after at most 131,070 player moves



Preliminary Results

- We trained a DQN agent like in exercise 7 on an OpenAI gym domain of the game
- 16, 50, 50, 4 nodes
- Training occurred in blocks of 1000 episodes on a running index n counting completed blocks
- $\varepsilon_n = 0.1 / \sqrt{(1+n)}$, to be able in theory to reach deeper parts of the game tree
- Performance capped at only ~ +10% of random performance after ~5000 episodes



Q-learning on-policy from Reinforcement Learning in MPC

Alvaro Javier Florez Martinez
Alejandro Astudillo Vigoya

KU Leuven

Model Predictive Control and Reinforcement Learning summer course

Faculty of Engineering, University of Freiburg

Problem description

A parametrized MPC will be use to approximate the optimal policy and the value functions

$$
\begin{aligned}
Q_\theta(\boldsymbol{s},\boldsymbol{a}) = \underset{\boldsymbol{x},\boldsymbol{u}}{\text{minimize}} \quad & \lambda_\theta(x_0) + \gamma^N V_\theta(x_N) + \sum_{k=0}^{N-1} \gamma^k \ell_\theta(x_k,u_k) \\
\text{subject to} \quad & x_0 = s \quad u_0 = a, \\
& x_{k+1} = f_\theta(x_k,u_k), \\
& h_\theta(x_k,u_k) \leq 0
\end{aligned}
$$

$$
\pi_\theta(s) = \arg\min_a Q_\theta(s,a), \quad V_\theta(s) = \min_a Q_\theta(s,a)
$$

Sensitivity of fully converged MPC

$$
\nabla_\theta Q_\theta(s,a) = \nabla_\theta \mathcal{L}_\theta(s,y^*)
$$

Q-Learning

$$
\begin{aligned}
\delta_k =& \ell(s_k,a_k) + \gamma \min_{a_{k+1}} Q_\theta(s_{k+1},a_{k+1}) - Q_\theta(s_k,a_k) \\
\boldsymbol{\theta} \leftarrow& \boldsymbol{\theta} + \alpha \delta_k \nabla_\theta Q_\theta(s_k,a_k)
\end{aligned}
$$

Problem description

A parametrized MPC will be use to approximate the optimal policy and the value functions

$$
\begin{aligned}
\underset{\boldsymbol{x},\boldsymbol{u},\sigma}{\text{minimize}} \quad & V_0 + \frac{\gamma^N}{2} x_N^T S_N x_N + \sum_{k=0}^{N-1} f^T \begin{bmatrix} x_k \\ u_k \end{bmatrix} + \sum_{k=0}^{N-1} \frac{1}{2}\gamma^k \left(||x_k||^2 + ||u_k||^2 + \omega^T \sigma_k \right) \\
\text{subject to} \quad & x_0 = s, \\
& \boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{b}, \\
& \begin{bmatrix} 0 \\ -1 \end{bmatrix} + \underline{\boldsymbol{x}} - \boldsymbol{\sigma}_k \le \boldsymbol{x}_k \le \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \overline{\boldsymbol{x}} + \boldsymbol{\sigma}_k, \\
& \boldsymbol{\sigma}_k \ge 0, \\
& -1 \le u_k \le 1
\end{aligned}
$$

where

$$\theta = [V_0, \underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}, \boldsymbol{b}, f, \boldsymbol{A}, \boldsymbol{B}]$$

Q-Learning

$$
\begin{aligned}
\delta_k =& \ell(s_k, a_k) + \gamma V_\theta(s_{k+1}) - Q_\theta(s_k, a_k) \\
\boldsymbol{\theta} \leftarrow& \boldsymbol{\theta} + \alpha \delta_k \nabla_\theta Q_\theta(s_k, a_k)
\end{aligned}
$$

Results

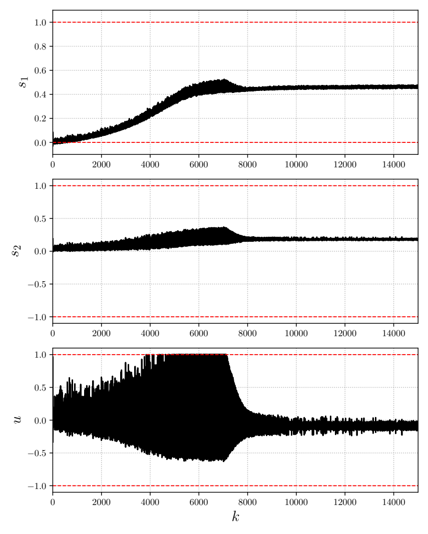


Figure 1: States and input

Results

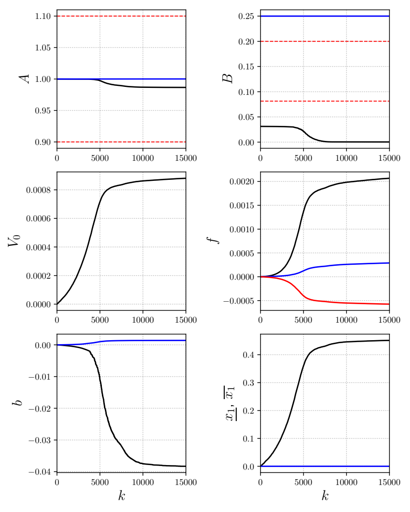


Figure 2: Parameters

How to train a computer to play **advanced** air hockey?

BASIC GAME AUTOMATIZATION USING UNITY ENVIRONMENT AND REINFORCEMENT LEARNING

PROJECT-MEMBER: NICK HARDER AND TIM FÜRMANN

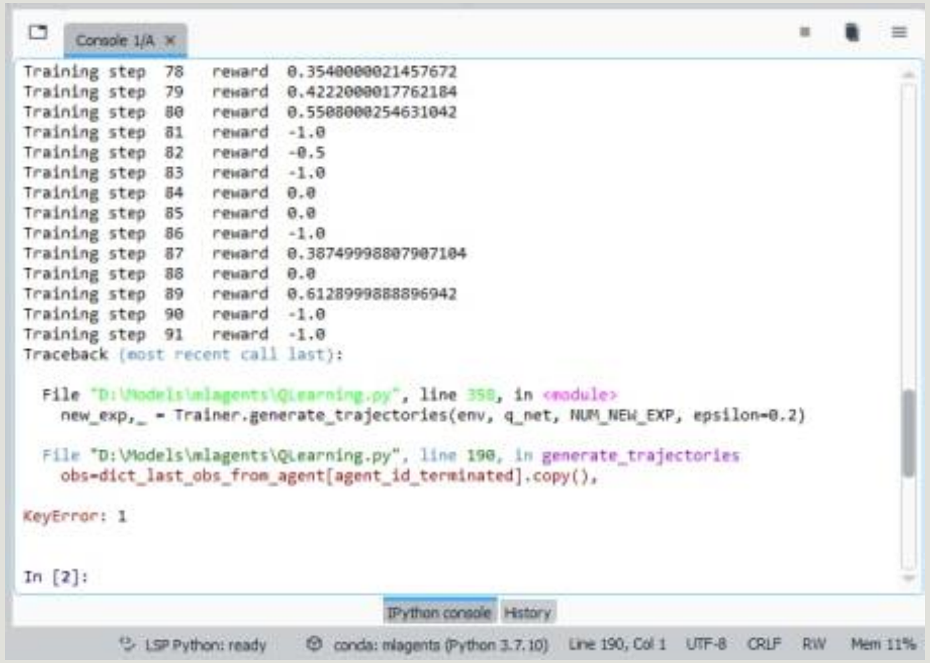
First Approach: Self-Made DQN-Agent

DQN: Lecture code from exercise 8 adapted to the unity environment and game scenario

State Space: 336 states, but
only 264 accessible

Action Space: 3x3 actions per agent

Training Idea: Train agent using DQN with the opponent agent as random noise (actions)



Results:

Training: Unsuccessful -> Error in the environment and python interface

Unity environment, not as straight forward to use in combination with python as expected

Final Results: POCA vs. PPO

MA-POCA (MultiAgent Posthumous Credit Assignement): Novel multi-agent trainier that trains a centralized ciritc, a neural network that acts as a „coach“ for a whole group of agents
=> Team rewards are included, the agents learn cooperatively

PPO (Proximal Policy Optimization)
=> Only single rewards are accounted for, the agent learn only for themselves and team efforts not additionally accounted for

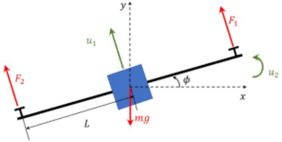
For the Presentation we hopefully have results on a second screened PC (remote work is today still not the best option)

(Optimal) Quadrotor Control

MPC vs RL

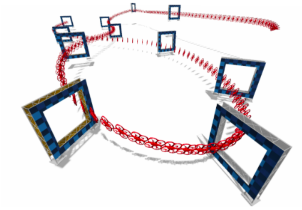
Shamil Mamedov, Mathias Bos
KU Leuven
August 2021

1 Problem Formulation

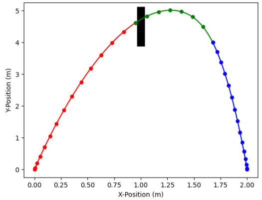


$$m\ddot{x} = -(F_1 + F_2)\sin(\phi)$$
$$m\ddot{y} = (F_1 + F_2)\cos(\phi) - mg$$
$$I\ddot{\phi} = (F_1 - F_2)L$$

states: $\left[x, y, \phi, \dot{x}, \dot{y}, \dot{\phi}\right]$
control actions: $[F_1, F_2]$



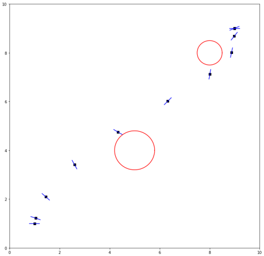
Y. Song*, M. Steinweg*, E. Kaufmann, D. Scaramuzza, "Autonomous Drone Racing with Deep Reinforcement Learning", 2021

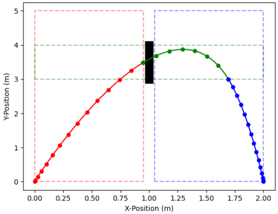


2 Results

Optimal control

minimize T





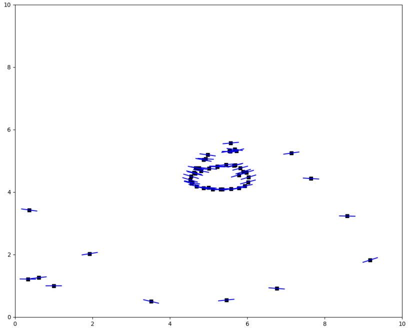
Reinforcement Learning

Stable baselines 3: SAC

Reward:

- negative distance to goal
- or 1/dist
- penalty on hitting sides

'Future work': other reward choices...



ILQR-controlled inverted pendulum

Yizhen Wang

Microsystems engineering

$$\min_{\{s_N\},\{a_{N-1}\}} \sum_{k=0}^{N-1} \frac{1}{2} \begin{bmatrix} s_k \\ a_k \end{bmatrix}^T \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix} \begin{bmatrix} s_k \\ a_k \end{bmatrix} + \frac{1}{2} s_N^T P s_N$$

subject to,

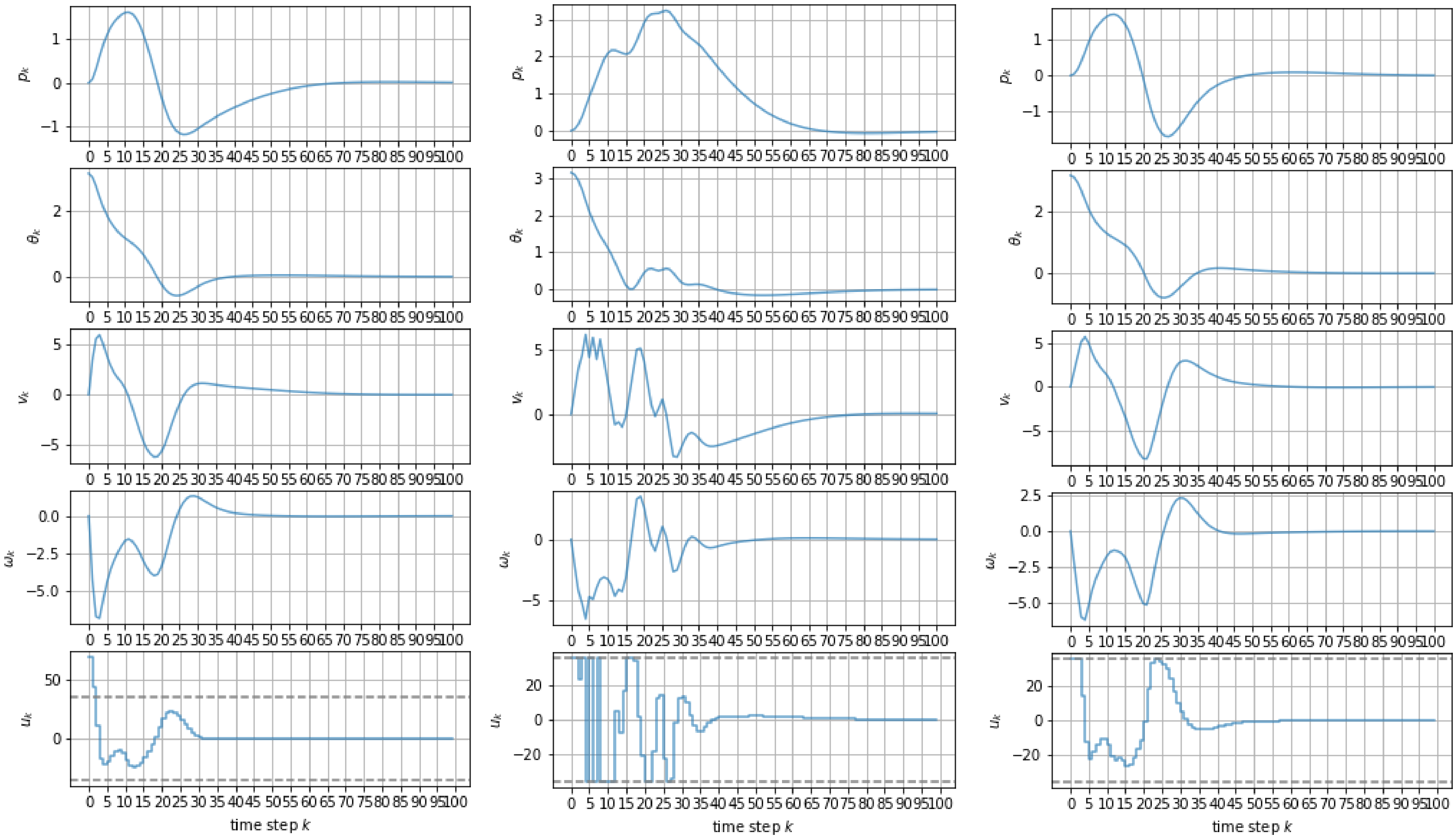
$$s_0 - \bar{\bar{s}}_0 = 0,$$
$$s_{k+1} - f_k(s_k, a_k) = 0, \; k = 0, \dots, N-1$$

where,

$$s = \begin{bmatrix} p \\ \theta \\ v \\ \omega \end{bmatrix}, \; a = [F]$$

For OCP,
N=20
Q=diag(10^3, 10^4, 10^{-2}, 10^{-2})

For LQR,
N=4
Q= diag(10^2, 10^3, 10^{-2}, 10^{-2})


p_k
θ_k
v_k
ω_k
u_k
time step k
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100