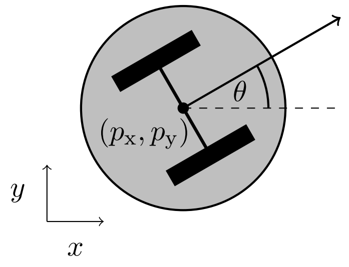
Exercise 4: Zero-order robust NMPC with `acados`

Florian Messerer, Jonathan Frey, Moritz Diehl

In this exercise, we learn how to use the algorithm ZORO in `acados`. The aim is to drive a robot from its starting position p_{start} to a goal position p_{target} while robustly avoiding an obstacle.



The state of the robot is given by $x = (p_{\text{x}}, p_{\text{y}}, \theta, v, \omega)$, where $p_{\text{x}}, p_{\text{y}}$ parametrize the 2D-position of the center of the robot, θ is the heading angle, v the forward velocity and ω the angular velocity. The controls $u = (a, \alpha)$ are the forward acceleration a and angular acceleration α. We assume that both the forward acceleration as well as the angular acceleration are subject to additive disturbances $w = (w_a, w_\alpha)$ yielding the ODE

$$\dot{p}_{\text{x}} = v \cos\theta, \tag{1a}$$
$$\dot{p}_{\text{y}} = v \sin\theta, \tag{1b}$$
$$\dot{\theta} = \omega, \tag{1c}$$
$$\dot{v} = a + \sigma_a w_a, \tag{1d}$$
$$\dot{\omega} = \alpha + \sigma_\alpha w_\alpha \tag{1e}$$

with σ_a, σ_α determining the scaling of the disturbances. This allows us to consider the unit sphere for the values of w_k, $w_k \in \mathcal{E}(0, I)$. We discretize the continuous-time system using one step of an explicit Runge-Kutta integrator of order 4 on an integration interval of length $h = 0.35$, with piecewise constant controls and disturbances to obtain the discrete-time dynamics

$$x_{k+1} = f(x_k, u_k, w_k). \tag{2}$$

As stage and terminal cost we use

$$l(x, u) = \gamma l_{\text{Huber}}(p - p_{\text{target}}) + u^\top R u, \tag{3a}$$
$$l_N(x) = \gamma_N l_{\text{Huber}}(p - p_{\text{target}}) + \tau v^2, \tag{3b}$$

with $p = (p_{\text{x}}, p_{\text{y}})$, weights R, γ, γ_N, τ, and where the pseudohuber loss $l_{\text{Huber}}(p) = \sqrt{p^\top p + 1}$ behaves like a quadratic function near the origin and like the unsquared 2-norm for larger values. This yields a smooth tracking behavior near the target position while proportionally penalizing larger distances.
The constraints include upper and lower bounds on the controls, bounds on the position, as well as an obstacle avoidance constraints,

$$u_{\min} \leq u \leq u_{\max}, \tag{4a}$$
$$p_{\min} \leq p, \tag{4b}$$
$$r_{\text{obs}} \leq \|p - p_{\text{obs}}\|_2, \tag{4c}$$

where r_{obs} denotes the radius of the obstacle and p_{obs} its center.

The nominal OCP, which we obtain by setting $w_k = 0$ for $k = 0, \ldots, N-1$, takes the form

$$\begin{aligned}
\min_{\bar{x},\,\bar{u}} \quad & \sum_{k=0}^{N-1} l_k(\bar{x}_k, \bar{u}_k) + l_N(\bar{x}_N) && \text{(5a)}\\
\text{s.t.} \quad & \bar{x}_0 = \bar{\bar{x}}_0, && \text{(5b)}\\
& \bar{x}_{k+1} = f_k(\bar{x}_k, \bar{u}_k, 0), \quad k = 0, \ldots, N-1, && \text{(5c)}\\
& 0 \geq h_k(\bar{x}_k, \bar{u}_k), \qquad k = 0, \ldots, N-1, && \text{(5d)}\\
& 0 \geq h_N(\bar{x}_N), && \text{(5e)}
\end{aligned}$$

with $\bar{u} = (\bar{u}_0, \ldots, \bar{u}_{N-1})$, $\bar{x} = (\bar{x}_0, \ldots, \bar{x}_N)$.
Recall that the nonlinear ellipsoidal tube OCP is given as

$$\begin{aligned}
\min_{\bar{x},\,\bar{u},\,P} \quad & \sum_{k=0}^{N-1} l_k\left(\bar{x}_k, \bar{u}_k\right) + l_N\left(\bar{x}_N\right) && \text{(6a)}\\
\text{s.t.} \quad & \bar{x}_0 = \bar{\bar{x}}_0, && \text{(6b)}\\
& P_0 = 0, && \text{(6c)}\\
& \bar{x}_{k+1} = f_k(\bar{x}_k, \bar{u}_k, 0), \qquad k = 0, \ldots, N-1, && \text{(6d)}\\
& P_{k+1} = \psi_k(\bar{x}_k, \bar{u}_k, P_k), \qquad k = 0, \ldots, N-1, && \text{(6e)}\\
& 0 \geq h_k(\bar{x}_k, \bar{u}_k) + b_k(\bar{x}_k, \bar{u}_k, P_k), \quad k = 0, \ldots, N-1, && \text{(6f)}\\
& 0 \geq h_N(\bar{x}_N) + b_N(\bar{x}_N, P_N). && \text{(6g)}
\end{aligned}$$

The main idea of the zero-order robust NMPC algorithm[1] (zoRO) is to fix the disturbance ellipsoids within one optimization iteration and only update the trajectory of disturbance ellipsoids outside the OCP.
In this exercise we only consider a very simple MPC problem, with a fixed reference point. For a more elaborate implementation, with time-varying reference tracking, including the generation of a reference trajectory alongside a given path, see `https://github.com/acados/acados/blob/main/examples/acados_python/zoRO_example/diff_drive`, which corresponds to the folder `acados/examples/acados_python/zoRO_example/diff_drive` in your local acados clone. The details of this implementation are described in Gao2023[2].
A side remark on the Generalized Gauss Newton (GGN) Hessian approximation (not necessary to read to solve the exercise, but helps to understand how the stage cost is set in acados): Recall that the stage and terminal cost functions are given as

$$\begin{aligned}
l(x, u) &= \gamma l_{\text{Huber}}(p - p_{\text{target}}) + u^\top R u, && \text{(7a)}\\
l_N(x) &= \gamma_N l_{\text{Huber}}(p - p_{\text{target}}) + \tau v^2. && \text{(7b)}
\end{aligned}$$

Both have a "convex-over-nonlinear" structure, $\psi(F(z))$, where we call $\psi(\cdot)$ the "outer convexity" and $F(\cdot)$ the "inner nonlinearity". The stage cost can be written in this form by considering $z = (x, u)$, $F(z) = Sz$, with $S \in \mathbb{R}^{4 \times (n_x + n_u)}$ a selector matrix which selects the entries p_{x}, p_{y}, u_1, u_2 from z (i.e., the entries which actually enter the stage cost). Note that in this case, F is not actually nonlinear, but it could be in general. The outer convexity is then given as $\psi(y) = \gamma l_{\text{Huber}}((y_1, y_2) - p_{\text{target}}) + (y_3, y_4)^\top R\,(y_3, y_4)$, with $y = F(z)$. For problems with this "convex-over-nonlinear" structure, we can use, e.g., the Generalized Gauss Newton (GGN) Hessian approximation. For more details, see[3].

[1] Andrea Zanelli, Jonathan Frey, Florian Messerer, and Moritz Diehl. Zero-order robust nonlinear model predictive control with ellipsoidal uncertainty sets. *Proceedings of the IFAC Conference on Nonlinear Model Predictive Control (NMPC)*, 2021

[2] Yunfan Gao, Florian Messerer, Jonathan Frey, Niels van Duijkeren, and Moritz Diehl. Collision-free motion planning for mobile robots by zero-order robust optimization-based mpc. In *Proceedings of the European Control Conference (ECC)*, 2023

[3] Florian Messerer, Katrin Baumgärtner, and Moritz Diehl. Survey of sequential convex programming and generalized Gauss-Newton methods. *ESAIM: Proceedings and Surveys*, 71:64–88, 2021

Tasks

1. First consider the file `main_ocp.py`. Here, solve the initial OCP with both IPOPT and zoRO, in order to compare the two solutions. Complete the zoRO solver found in `solver_zoro_acados.py` such that you are able to run `main_ocp.py`. Note that we split the inequality constraints into box constraints on the states and controls as well as nonlinear constraints. Take a moment to compare the two resulting trajectories.

2. Now consider `main_mpc.py`. Take a moment to familiarize yourself with the code, and run it. How does the resulting trajectory compare to the initial OCP prediction?

3. In the previous simulation, all disturbance values were set to 0. Look for the `TODO` in `main_mpc.py` and play around with the disturbance scaling.

4. The above MPC setting was fairly simple. If you are interested, take a look at the more elaborate path following zoRO mentioned above.