# 10 Mario Kart challenge

4SC000, TU/e, 2025-2026

Consider an autonomous vehicle moving in the 2D track depicted in the figure below. A simulation is provided in Matlab and in Python on canvas. The simulation uses a planar dynamic bicycle model with the following state vector and control inputs:

$$x = \begin{bmatrix} X & Y & \psi & v_x & v_y & r \end{bmatrix}^\top, \qquad u = \begin{bmatrix} \delta & a_x \end{bmatrix}^\top,$$

where $\delta$ is the front steering angle and $a_x$ is the commanded longitudinal acceleration.

The vehicle is modeled using a planar dynamic bicycle model with state

$$x = \begin{bmatrix} X & Y & \psi & v_x & v_y & r \end{bmatrix}^\top,$$

and control inputs

$$u = \begin{bmatrix} \delta & a_x \end{bmatrix}^\top,$$

where $\delta$ is the front steering angle and $a_x$ is the commanded longitudinal acceleration.

The Kinematics in the Global Frame are

$$\begin{aligned} \dot{X} &= v_x \cos\psi - v_y \sin\psi, \\ \dot{Y} &= v_x \sin\psi + v_y \cos\psi, \\ \dot{\psi} &= r. \end{aligned}$$

The front and rear slip angles are given by

$$\begin{aligned} \alpha_f &= \delta - \arctan\left(\frac{v_y + l_f r}{v_x^{\text{safe}}}\right), \\ \alpha_r &= -\arctan\left(\frac{v_y - l_r r}{v_x^{\text{safe}}}\right). \end{aligned}$$

The tire Forces are modeled as ((Linear Model)

$$\begin{aligned} F_{yf} &= C_f \alpha_f, \\ F_{yr} &= C_r \alpha_r. \end{aligned}$$

These are the Vehicle Dynamics in the Body Frame

$$\begin{aligned} \dot{v}_x &= a_x - \frac{F_{yf}\sin\delta}{m} + r v_y, \\ \dot{v}_y &= \frac{F_{yf}\cos\delta + F_{yr}}{m} - r v_x, \\ \dot{r} &= \frac{l_f F_{yf}\cos\delta - l_r F_{yr}}{I_z}. \end{aligned}$$

This is the complete State-Space model

$$\dot{x} = \begin{bmatrix} v_x \cos\psi - v_y \sin\psi \\ v_x \sin\psi + v_y \cos\psi \\ r \\ a_x - \frac{F_{yf}\sin\delta}{m} + r v_y \\ \frac{F_{yf}\cos\delta + F_{yr}}{m} - r v_x \\ \frac{l_f F_{yf}\cos\delta - l_r F_{yr}}{I_z} \end{bmatrix}.$$

The continuous-time dynamics are integrated using explicit Euler discretization:

$$x_{k+1} = x_k + \dot{x}_k\,\Delta t.$$

The numerical values of all physical and controller parameters used in the simulations are:

$$\begin{aligned} m &= 150.0 \text{ kg}, \\ I_z &= 20.0 \text{ kg}\,\text{m}^2, \\ l_f &= 0.7 \text{ m}, \\ l_r &= 0.7 \text{ m}, \\ C_f &= 800.0 \text{ N/rad}, \\ C_r &= 800.0 \text{ N/rad}. \end{aligned}$$

The steering and longitudinal actuation limits are

$$\begin{aligned} \delta_{\max} &= 25^\circ \ (\approx 0.436 \text{ rad}), \\ a_x^{\max} &= 4.0 \text{ m/s}^2, \\ a_x^{\min} &= -6.0 \text{ m/s}^2. \end{aligned}$$

For the numerical integration the step size $\Delta t = 0.02$ is considered. The initial state is given in the simulator. The finish fine is indicated in grey in the figure. The boundaries of the track are defined by a set of straight lines and 3rd order splines. The parameters of these straight lines and splines can be found in the Simulator provided for this assignment in canvas.

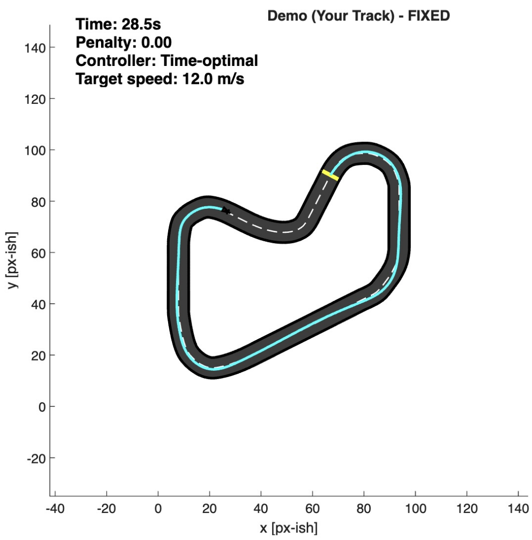
The goal of this assignment is to compute a control input which allows the car to complete one lap (anti-clockwise direction) starting from the initial position and crossing the finish line in minimum time, without stepping out of the track boundaries[1] and satisfying the input constraints. The current simulator contains a function

```
[delta, ax_cmd ] = time_optimal_controller( state, s_progress,
s_dist, waypts, vx_ref, max_accel, max_brake, look_ahead, k_delta,
max_steer, a_lat_max, nsamples)
```

that output the input commands given the state and additional arguments. You are only allowed to change this function (you do not have to use the additional arguments).

You should submit a video of at most 4 minutes long to canvas, explaining how you tackled the problem and your results, together with your function time_optimal_controller. Your results should be reproducible if the current simulation function time_optimal_controller is replaced by yours. You can use Matlab or Python.

[1] the position $(x, y)$ should stay inside the track


Demo (Your Track) - FIXED
Time: 28.5s
Penalty: 0.00
Controller: Time-optimal
Target speed: 12.0 m/s
y [px-ish]
x [px-ish]
140
120
100
80
60
40
20
0
-20
-40
-20
0
20
40
60
80
100
120
140