Control of Mobile Robots
Project work
Prof. Luca Bascetta

Instructions

Each student should upload the project to the WeBeep folder "Assignments/Project work" as a PDF file, at least 3 working days before the date of the exam call, as a compressed zip file named "studentID_surname_name.zip", including:

- a PDF file, written in English or in Italian, answering to the questions (see below);
- a ROS workspace including all the code used to answer the questions.

If running the launch file of the scripts reported in question 4 no results are generated, or results that are different from the ones reported in the PDF file are generate, the project work gets zero points.
The project work can be submitted only once for each academic year. The mark obtained with the first submission is valid for all the academic year and cannot be changed submitting another solution (even in the case of zero points).

Problem

Consider a car-like robot simulated using its single-track dynamic model, with a linear tyre model. The parameters that characterise the robot are:

- mass m, given in the parameter table;
- distance of the center of gravity from the front axle a, 0.14 m;
- distance of the center of gravity from the rear axle b, 0.12 m;
- ground friction coefficient μ, 0.385;
- front wheel cornering stiffness C_{α_f}, 50 N/rad;
- rear wheel cornering stiffness C_{α_r}, 120 N/rad;
- yaw inertia I_z, given in the parameter table.

Set up a ROS package "car_simulator", based on Boost Odeint library, to simulate the robot.
The package should include:

- a ROS node that simulates the robot;
- a ROS node that allows to feed the simulator with simple (e.g., step) commands to test the model behaviour;
- a yaml file including model parameters;
- a launch file to run the simulator and the test node;
- a python or MATLAB script to plot the test results starting from a bag file.

Set up a ROS package "car_traj_ctrl", implementing a trajectory tracking controller.
The controller is composed of a feedback linearisation law, based on the bicycle kinematic model, and a PI trajectory tracking controller with velocity feedforward (PI controller can be implemented using Forward Euler discretisation).
The trajectory tracking controller should be tested using an 8-shaped trajectory, generated according to the following equations:

$$x = a \sin\left(\frac{2\pi}{T}t\right)$$
$$y = a \sin\left(\frac{2\pi}{T}t\right)\cos\left(\frac{2\pi}{T}t\right)$$

where a =2 m, and the value of T is given in the parameter table.
The package should include:

- a ROS node that implements the trajectory tracking controller, including reference trajectory generation;
- a yaml file including model parameters;
- a launch file to run the simulator and the trajectory tracking controller;
- a python or MATLAB script to plot the test results starting from a bag file.

Questions

1. Tune the trajectory tracking controller, selecting in an appropriate way the PI parameters and the sampling time, in order to achieve the best performance in terms of tracking error, and report the values of:
   (a) the proportional gains K_{P_x}, K_{P_y};
   (b) the integral time constants T_{I_x}, T_{I_y};
   (c) the sampling time T_s.
2. Report the results of a test of the trajectory tracking controller with the bicycle kinematic model. In particular, the following pictures should be reported:
   (a) a figure showing the reference and actual robot trajectory;
   (b) a figure showing the x and y components of the tracking error;
   (c) a figure showing the two control signals (velocity and steering);

   together with the value of the maximum tracking error in the x and y directions.
3. Report the results of a test of the trajectory tracking controller with the single-track dynamic model. In particular, the following pictures should be reported:
   (a) a figure showing the reference and actual robot trajectory;
   (b) a figure showing the x and y components of the tracking error;
   (c) a figure showing the two control signals (velocity and steering);

   together with the value of the maximum tracking error in the x and y directions.
4. Report the names of the launch files and python/MATLAB scripts one have to run in order to generate the results required in questions 2 and 3.