Report on Car-like Robot trajectory controller

Andrea Romanzi

February 14, 2024

Contents

Chapter 1

Problem Statement

Consider a car-like robot with the following parameters:

- Mass m, 2.2 kg.
- 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 $C_{\alpha f}$, 50 N/rad.
- Rear wheel cornering stiffness $C_{\alpha r}$, 120 N/rad.
- Yaw inertia I_z, 0.019 kgm^2.

Set up a ROS architecture to simulate the robot and a trajectory tracker. Answer the following questions testing the controller on a 8-shaped trajectory according to the following equations:

$$x = a \sin\left(\frac{2\pi}{T}\right) \tag{1.1}$$

$$y = a \sin\left(\frac{2\pi}{T}\right) \cos\left(\frac{2\pi}{T}\right) \tag{1.2}$$

where:

- a, 2.0 m.
- T, 3.0 s.

Chapter 2

Questions and Answers

Question 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. Report the values of:

- (a) Proportional gains K_{Px}, K_{Py}.
- (b) Integral time constants T_{Ix}, T_{Iy}.
- (c) Sampling time T_s.

Explain the procedures followed to determine the controller requirements (crossover frequency, zero steady-state error, disturbance rejection, actuator effort limitation, etc.) and to tune the controller.

Answer:
In order to properly tune the controller parameters we need to first find out the model of the robot. It is a car-like robot, therefore we can model it using a single-track model, however this is non-linear and this makes the control quite complex. In order to simplify it we can use the feedback linearization based on the kinematic model, this reduces the model to 2 independent integrators, but makes the trajectory controller performance depend on an additional parameter P_{dist}:

$$G_x(s) = \frac{1}{s} \tag{2.1}$$

$$G_y(s) = \frac{1}{s} \tag{2.2}$$

therefore is it possible to design two independent PI regulators.

To tune the parameters let's select first the desired cut-off frequency, we want to follow the 8-shaped trajectory with frequency:

$$\omega = 2\pi/T = 2.094 rad/s \tag{2.3}$$

so we want to have $w_c \geq \omega$, additionally we would like to have a good disturbance rejection so we want to have the cut-off frequency as high as possible, therefore let's choose:

$$w_c = 20Hz \tag{2.4}$$

We need a sampling frequency at least 10 times faster than w_c, therefore the sampling frequency chosen is:

$$T_s = 0.005s \tag{2.5}$$

Then the loop transfer function will be:

$$L_x(s) = R_x(s)G_x(s) = \frac{K_{Px} \cdot (1 + T_{Ix} \cdot s)}{T_{Ix} \cdot s^2} \tag{2.6}$$

$$L_y(s) = R_y(s)G_y(s) = \frac{K_{Px} \cdot (1 + T_{Ix} \cdot s)}{T_{Ix} \cdot s^2} \tag{2.7}$$

We would like to have a good response, without overshoots, this means:

$$\phi_m \geq 60^\circ \tag{2.8}$$

to have this we want the $L(s)$ to cut the $0dB$ axis with -1 slope, therefore the zero must be before w_c at low frequency, let's choose:

$$T_{Ix} = 1 \tag{2.9}$$

$$T_{Iy} = 1 \tag{2.10}$$

Now to have $w_c = 20Hz$ we need $K_{Px,y}$ to be equal to:

$$K_{Px} = 20 \tag{2.11}$$

$$K_{Py} = 20 \tag{2.12}$$

So we get the following loop transfer functions:

$$L_{x,y}(s) = \frac{20 \cdot (1 + \cdot s)}{\cdot s^2} \tag{2.13}$$

with the bode plot in fig. 2.1 Getting the following results that satisfy the requirements:

$$w_c = 20Hz \tag{2.14}$$

$$\phi_m = 87.1^\circ \tag{2.15}$$

Moreover thanks to the integrators we get 0 steady state error, while regarding the actuator effort we will check that in simulation. We still need to choose the

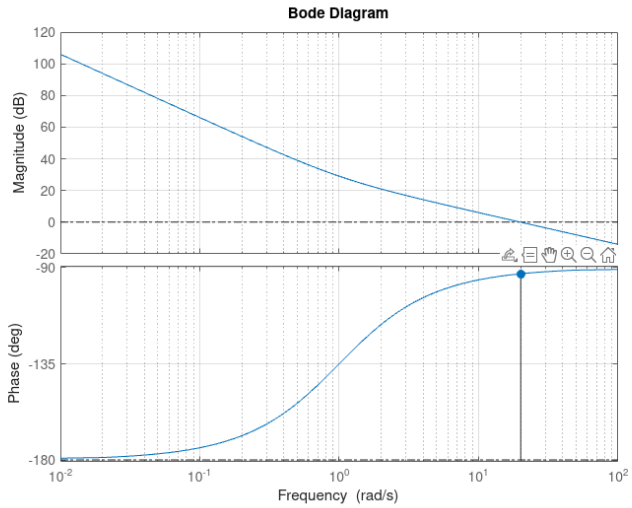


Figure 2.1: Bode plot of $L_{x,y}(s)$ with $w_c = 20Hz$

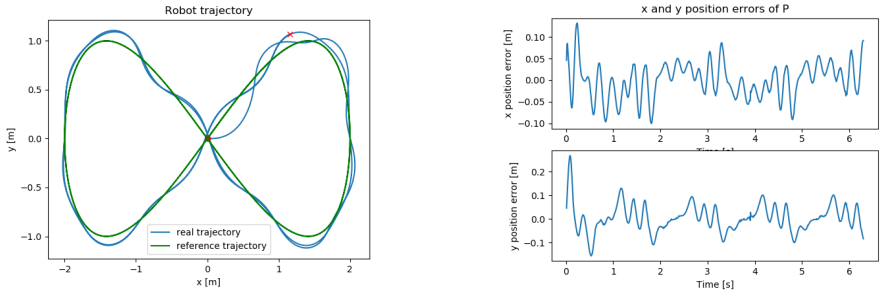
P_{dist} parameter used for the feedback linearization, let's start setting P_{dist} on the front wheel axle:

$$P_{dist} = 0.12 \tag{2.16}$$

Testing the regulators in simulation on the dynamic single-track model we get the results in fig.2.2:
Looking at fig. 2.2b we can compute these maximum errors:

$$max(x_{error}) = 0.13662298038746834 \tag{2.17}$$

$$max(y_{error}) = 0.27120581460181514 \tag{2.18}$$



(a) Trajectory followed.

(b) x,y position errors

Figure 2.2: First tuning.

and mean absolute errors:

$$MAE_x = 0.031162930444451453 \tag{2.19}$$

$$MAE_y = 0.04134550132062547 \tag{2.20}$$

in particular looking at the actual trajectory followed, fig. 2.2a, we can notice a shaking behaviour, this may be related to the feedback linearization, indeed we are neglecting the dynamic behaviour of the robot using the kinematic model to linearize the non-linear model but then we test it on the dynamical one. This reduce the phase margin ϕ_m and cause the shaking behaviour, to fix it we can try to reduce the cut-off frequency w_c to:

$$w_c = 9Hz \tag{2.21}$$

we get approximately same phase margin, fig. 2.3:

$$\phi_m = 83.7° \tag{2.22}$$

While as regard the sampling frequency we can in principle reduce it, having

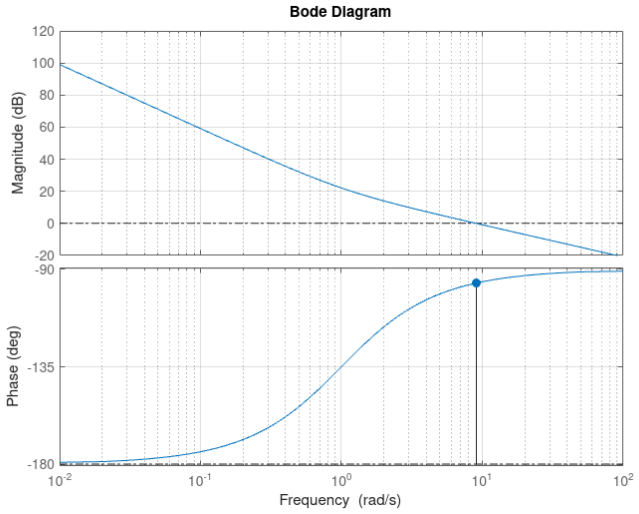
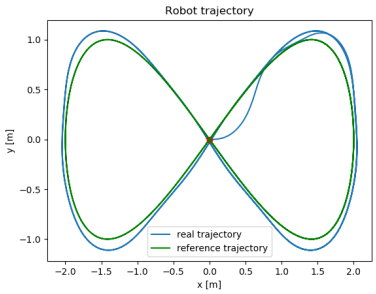


Figure 2.3: Bode plot of $L_{x,y}(s)$ with $w_c = 9Hz$

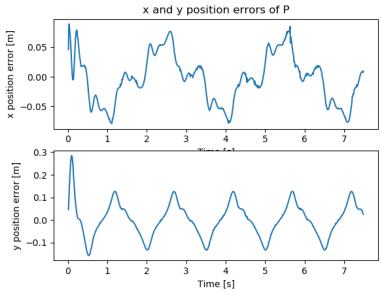
reduced the cut-off frequency, but for convenience we leave it unchanged.
With this controller we get the following results: From fig. 2.4a we can see that there isn't shaking anymore and the errors, fig. 2.4b, are almost unchanged:

$$max(x_{error}) = 0.08905151213336476 \tag{2.23}$$

$$max(y_{error}) = 0.2874923899640151 \tag{2.24}$$



(a) Trajectory followed.



(b) x,y position errors

Figure 2.4: Second tuning.

and mean absolute errors of:

$$MAE_x = 0.03161931003458963 \tag{2.25}$$

$$MAE_y = 0.05943784679968786 \tag{2.26}$$

Regarding the actuator effort, for the steering position, it oscillates between $0.8rad$ and $-0.8rad$ reaching a maximum value of $1.2rad$, that corresponds to $\delta_{max} = 68.75°$, and given that usually in mobile robotics angles up to $45°$ are allowed this may be a problem (a possible solution will be given later), while for the longitudinal velocity, maximum value reached is slightly less than $8m/s$, but we don't have any information on the actuators provided, hence we cannot comment this.

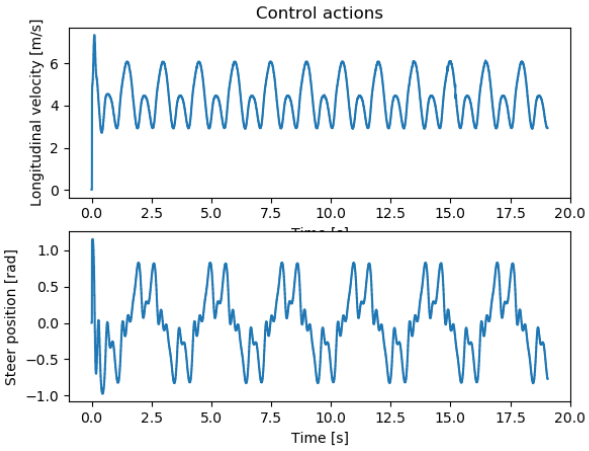


Figure 2.5: Control actions

Question 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) Figure showing the reference and actual robot trajectory.
- (b) Figure showing the x and y components of the tracking error.
- (c) Figure showing the two control signals (velocity and steering).

Comment on the quality of the results, with particular reference to your tuning procedure or tuning requirements.

Answer: Testing the 9 Hz bandwidth controller on the simulator based on the kinematic model we get the following results:

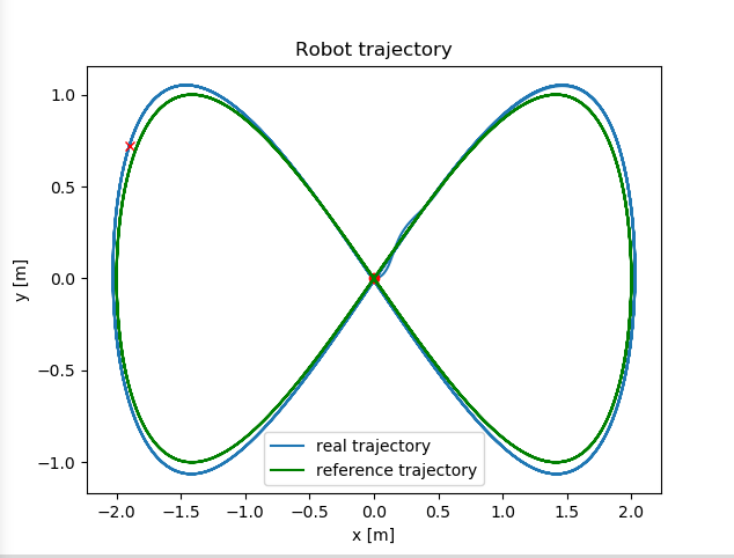


Figure 2.6: Trajectory followed.

Looking at fig. 2.7, we have maximum errors of:

$$max(x_{error}) = 0.11355266396726799 \tag{2.27}$$

$$max(y_{error}) = 0.14080395596987583 \tag{2.28}$$

the y maximum is definitely smaller thanks to the improved initial behaviour of the controller while the x one is more or less the same and mean absolute errors of:

$$MAE_x = 0.018089481433225533 \tag{2.29}$$

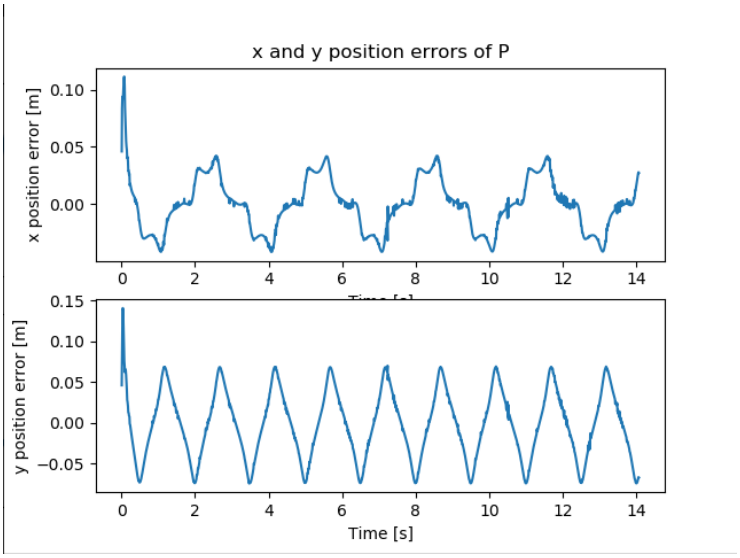
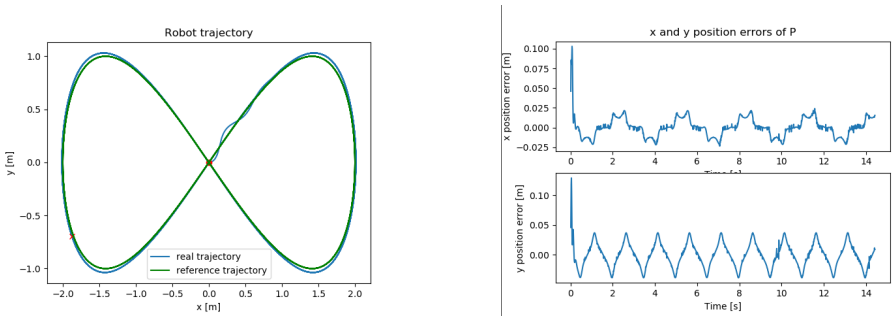


Figure 2.7: x,y position errors.

$$MAE_y = 0.03493699416850828 \tag{2.30}$$

both definitely smaller than the dynamic case as we expected, because the linearization and the testing are both based on the kinematic model, however we still have errors in the turns, fig. 2.6, this is due to the change in curvature of the reference that can be seen as a disturbance from the control perspective, hence increasing the bandwidth we are able to mitigate this effect. Testing the first controller tuned with bandwidth $w_c = 20Hz$ (loop transfer function(2.8)) in fact we get, fig. 2.8.



(a) Trajectory followed.

(b) x,y position errors

Figure 2.8: First tuning.

Can be easily perceived that the error along the turns is smaller through fig.

2.8a, computing the mean errors in fact we get:

$$MAE_x = 0.008765788411950899 \tag{2.31}$$

$$MAE_y = 0.016884848796767802 \tag{2.32}$$

Therefore the controller with 20Hz of bandwidth behaves better as expected on the kinematic simulator but will result in undesired oscillations when tested on the dynamic model therefore from now one we consider the 9 Hz bandwidth controller, because it has a better dynamic behaviour. For both the controllers we can see an initial oscillating behaviour this is related to the look ahead distance P_{dist} increasing this parameter in fact we reduce it, actually i have decided keep it unchanged $P_{dist} = 0.12$ because the oscillations are quite small with the 9 Hz bandwidth controller. Moreover we don't reach zero error due to the fact that the reference is changing so we are not in a steady state scenario. Regarding the actuator effort we get the following control actions:

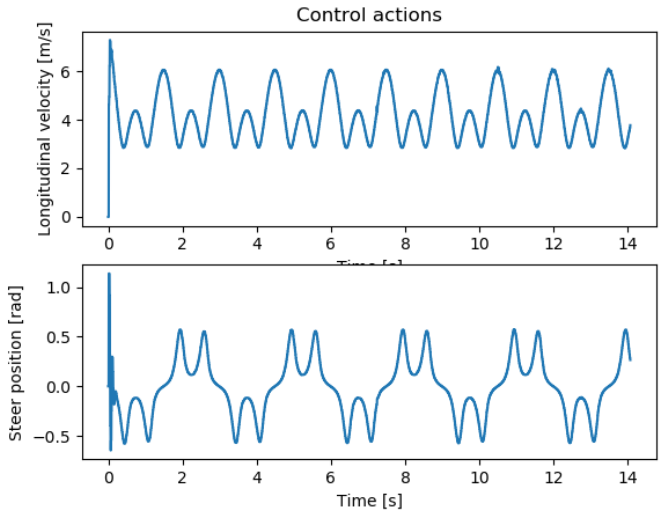


Figure 2.9: Control actions

Longitudinal velocities are limited approximately to $7m/s$, therefore not a very high velocity, this depends on the actuators but should not create problems, while for the steering angle during the initial maneuver to align the robot with the reference trajectory we reach $\delta_{max} \approx 1.2rad$ that means $\delta_{max} \approx 68.7°$, this may cause problems cause we are reaching the steering actuator limits, hence we should take this into account creating an anti wind-up control scheme on the steering angle (not so easy due to the control transformation).

Question 3

Report the results of a test of the trajectory tracking controller with the single-track dynamic model, using a linear tyre model. In particular, the following pictures should be reported:

- (a) Figure showing the reference and actual robot trajectory.
- (b) Figure showing the x and y components of the tracking error.
- (c) Figure showing the two control signals (velocity and steering).

Comment on the quality of the results, with particular reference to your tuning procedure or tuning requirements.

Answer: As already in part commented in the first question, testing the controller on the dynamic simulator with linear friction model, we get the following results: From fig. 2.10 we can notice no shaking behaviour, but a bad initial

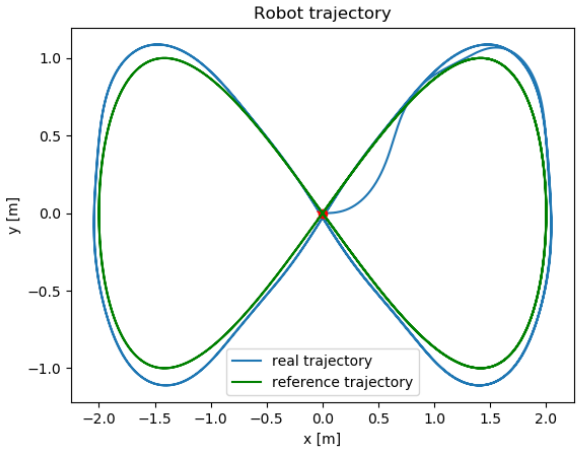


Figure 2.10: Trajectory followed.

behaviour, this is related to the dynamic simulator, it takes in fact into account the side-slip angle of the tyres not considered from the kinematic one, therefore the vehicle will take more to get aligned to the trajectory. Same reasoning can be applied to the turns, we have in fact bigger overall errors than the kinematic case, fig. 2.11a:

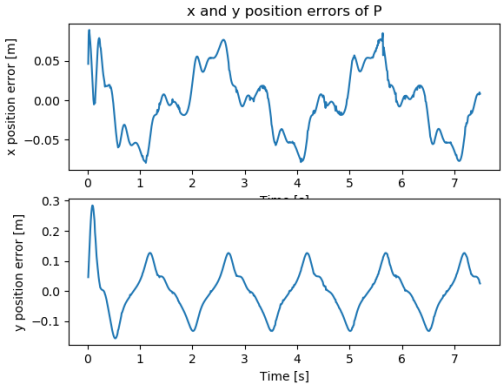
$$max(x_{error}) = 0.08905151213336476 \tag{2.33}$$

$$max(y_{error}) = 0.2874923899640151 \tag{2.34}$$

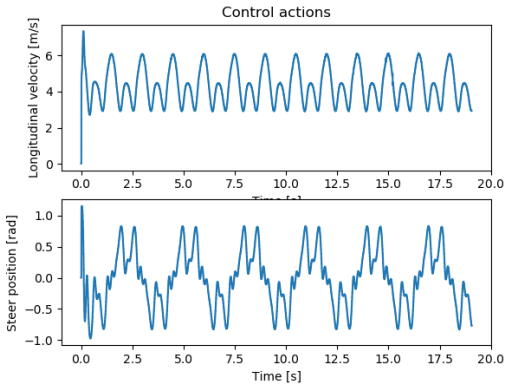
and mean absolute errors of:

$$MAE_x = 0.03161931003458963 \tag{2.35}$$

$$MAE_y = 0.05943784679968786 \tag{2.36}$$



(a) x,y position errors.



(b) Control actions.

Figure 2.11: Second tuning.

Analyzing instead the control actions, fig.2.11b, as already mentioned, the maximum steering angle reached is $\delta_{max} \approx 1.2rad = 68.75°$, and given that usually in mobile robotics angles up to 45° are allowed this will be a problem, hence as the kinematic case we should use an anti wind-up control scheme, moreover it can be noticed that overall are reached steering angle bigger than in the kinematic case and this is related to the side-slip angle of the vehicle, we need bigger steering angles to steer of the same amount than the kinematic case, while for the longitudinal velocity, maximum value reached is slightly less than $8m/s$, similarly to the kinematic case should not be a problem but it depends on the actuators mounted on the robot. Knowing the actuators, beyond the max velocity requested,it should be also checked the maximum accelleration deliverable against the one requested.

Question 4

Report the results of a test of the trajectory tracking controller with the single-track dynamic model, using a Fiala tyre model and reducing of 25 % the value of T . In particular, the following pictures should be reported:

- (a) Figure showing the reference and actual robot trajectory.
- (b) Figure showing the x and y components of the tracking error.
- (c) Figure showing the two control signals (velocity and steering).
- (d) Figure showing the front and rear lateral tyre forces, and their maximum values.

Compare these results with the ones obtained in Question 3.

Answer: Reducing of 25% T we get $T = 2.25$ but this means that $w_c \geq \omega = 2pi/T = 2.79 rad/s$. If we look at the simulations we have:

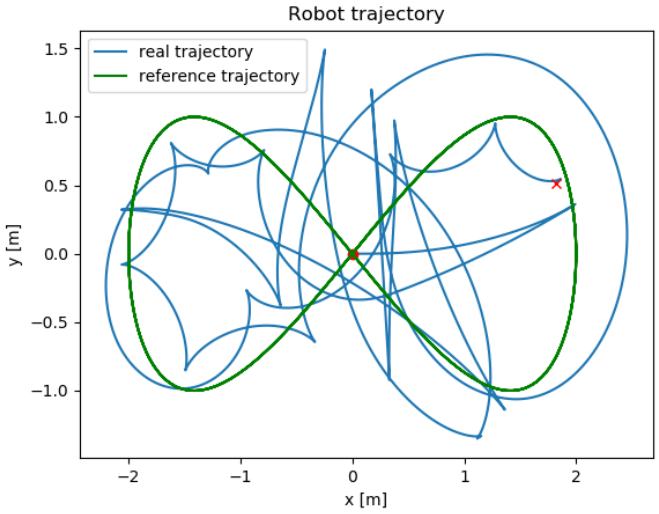


Figure 2.12: Trajectory followed.

The controller is not working at all, fig.2.12, hence we get very big errors, way bigger than the question 3, fig.2.13:

$$max(x_{error}) = 2.198321820652436 \tag{2.37}$$

$$max(y_{error}) = 0.8359359697557186 \tag{2.38}$$

The bandwidth here is not the problem because $w_c = 9Hz \geq \omega$, if we look to the lateral forces generated with the ground, fig.2.14, we have that the lateral forces

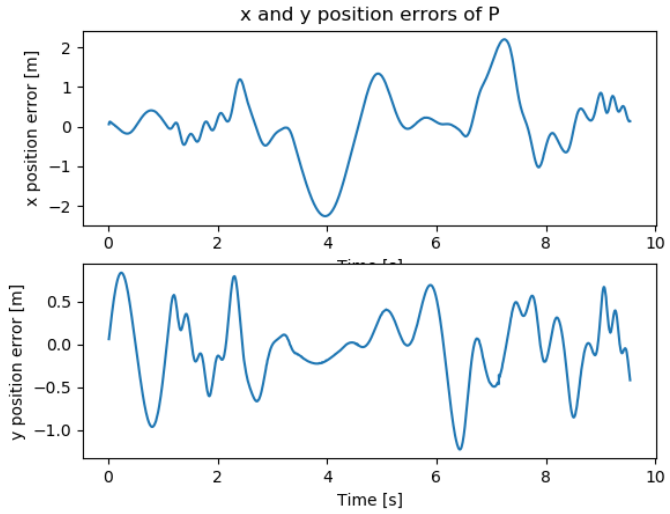


Figure 2.13: x,y position errors

are almost everywhere saturated to $F_{f,r_{max}} = 4$ or $F_{f,r_{min}} = -4N$, hence the regulators try to steer more,in fact steering angle is almost everywhere very high greater than 1 or smaller than -1 fig.2.15, but the robot is not able to generate enough lateral force with the ground to make the curves, in particular if we run a simulation with $T = 2.25$ and the linear tyre model we can look into the lateral forces generated to understand the real values needed, 2.16. In conclusion with these ground robot interaction parameters and with that trajectory the robot cannot follow the reference.

Question 5

Report:

- (a) the values of the parameters $m = 2.2kg$, $Iz = 0.019kgm^2$, and $T = 3.0s$;
- (b) the names of the launch files and Python or MATLAB scripts one have to run in order to generate the results required in questions 2, 3 and 4:
  - **Question 2:**
    - launch file: car_kin_trajctrl.launch
    - python file: plot_result_kin_trajctrl.py
  - **Question 3:**
    - launch file: car_dyn_trajctrl.launch

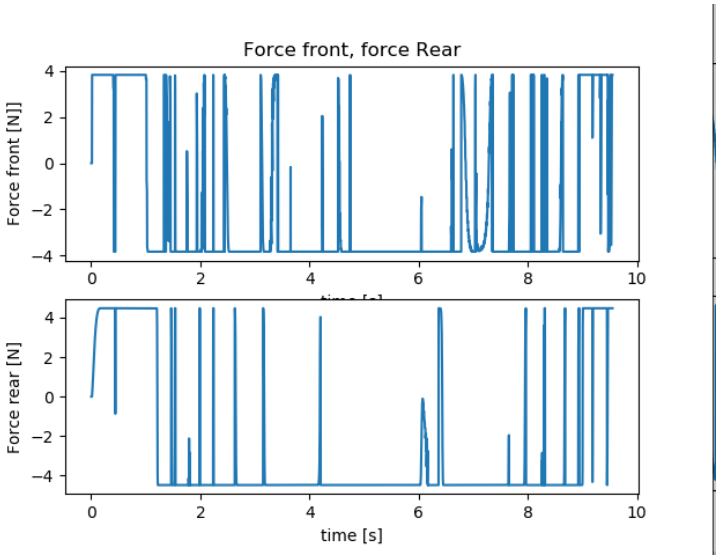


Figure 2.14: Front and rear lateral forces

  - python file: plot_result_trajctrl.py

- **Question 4:**
  - launch file: car_dyn_trajctrl.launch
  - python file: plot_result_trajctrl.py
    Here you must change T in the file at the path /car_traj_ctrl/config/car_kin_trajctrl_param.yaml to 2.25 and the parameter tyre_model to 1 at the path car_simulator/config/car_param.yaml

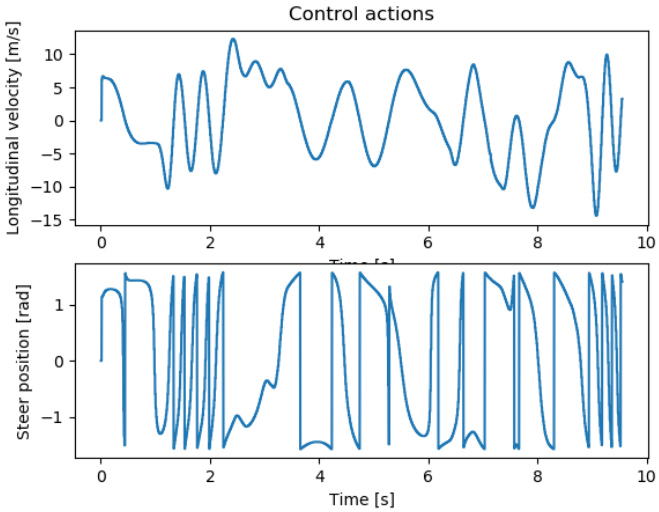


Figure 2.15: Control actions

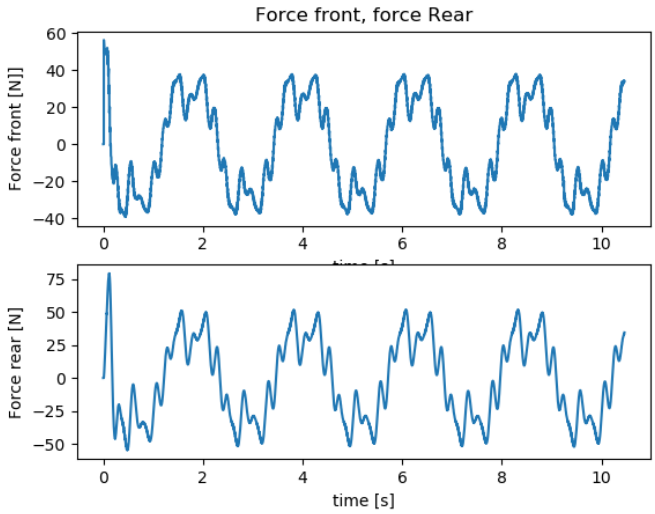


Figure 2.16: Front and rear lateral forces