Fundamentals of Mobile Robots

Tampere University
Tampere University of Applied Sciences

Exercise Report Number: 4

by:

Dong Le (151057054)

Niklas Kantele (H283436)

Date: 4.20.2023

THIS PAGE LEFT BLANK INTENTIONALLY

Dong Le, Niklas Kantele

Abstract

In this exercise we do some programming exercises. We implement proportional control with time-varying k and static k, track the moving goal by designing proportional control with feedforward term and design a proportional control for the orientation to reach the goal position.

Dong Le, Niklas Kantele

Table of Contents

Dong Le, Niklas Kantele

Introduction

This exercise is based on the concepts proportional and feedforward control.

Proportional control is a control system technology based on a response in proportion to the difference between what is set as a desired process variable (or set point) and the current value of the variable.

Feedforward is an element or pathway within a control system that passes a controlling signal from a source in its external environment to a load elsewhere in its external environment.

The purpose of this exercise is to learn how to design and implement a proportional control to get the robot to reach the desired goal position.

In the first problem we implement proportional control with static k and then time-varying k and plot the time series for them.
In the second problem we track the moving goal by designing proportional control with feedforward term.
In the third problem we design a proportional control for the orientation to reach the goal position and then find the minimum k in the proportional controller that ensures the robot reaches the goal.

Dong Le, Niklas Kantele

Methodology

Problem 1

The theory of proportional control was used to design control input to reach the goal. The theory of omnidirectional mobile robot.

First, the initial robot state and desired state are defined. Then using 3 zero matrices to store data from the process to plot after all. The current control input was calculated using k, which can be scalar or time-varying variable. A robot state at that time was made by adding time step times control input into the previous robot state.

Some libraries are used in the python code such as Matplotlib, numpy, a custom library called "visualize_mobile_robot" and a pre-made python code called "base_code_omnidirectional.py"

Problem 2

The theory of feedforward term was used to track the moving goal and designing proportional control. The theory of omnidirectional mobile robot.

First, the initial robot state and desired state are defined. Then using 3 zero matrices to store data from the process to plot after all. The current control input was calculated using constant k and feedforward. The feedforward was the different of the next goal state and the current goal state, the divided to the step time. A robot state at that time was made by adding time step times control input into the previous robot state.

Some libraries are used in the python code such as Matplotlib, numpy, a custom library called "visualize_mobile_robot" and a pre-made python code called "base_code_omnidirectional.py"

Problem 3

The theory of proportional control for orientation was used to reach the desired goal. The theory of unicycle mobile robot.

The purpose of the task is to design omega with a constant v to reach the desired goal. Firstly, the error between robot state and the goal can be calculated and then to find out the angle between the orientation of the robot and the goal. Secondly, omega from control input is calculated by the angle times k. Finally, by checking the different value of k, we found out the smallest value of k that able to reach the goal.

Some libraries are used in the python code such as Matplotlib, numpy, a custom library called "visualize_mobile_robot" and a pre-made python code called "base_code_unicycle.py"

Dong Le, Niklas Kantele

Results and Discussion

Problem 1

In this problem, omnidirectional mobile robot was used with the position (p_x, p_y)

$$x = \begin{bmatrix} p_x \\ p_y \\ \theta \end{bmatrix} \tag{1}$$

And the control input:

$$u = \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \tag{2}$$

Initial position $x[0] = [0\ 0\ 0]^T$

The goal $x^d = [-2\ 1\ 0]^T$

Question 1

In this task, the control input was definded as :

$$u = k(x^d - x), k > 0 \tag{3}$$

With 3 different set of k, figure (1) shows that the robot always went to the goal with the same trajectory. However there are different in the plots.

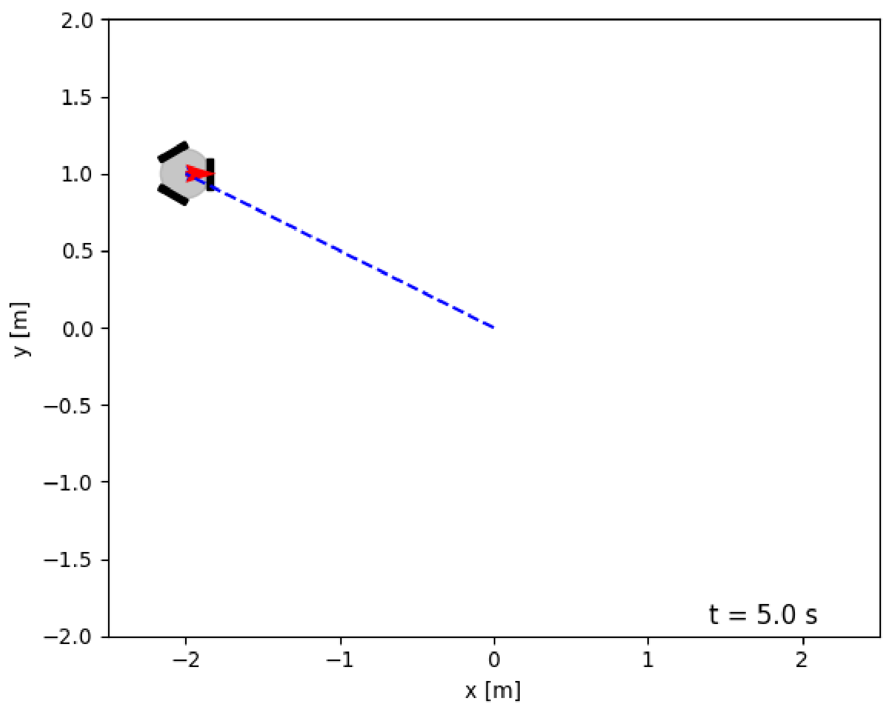

Figure 1: the trajectory of the robot

Firstly, k = 1, figure (2) and figure (3) shows the plot of time series of x and u.

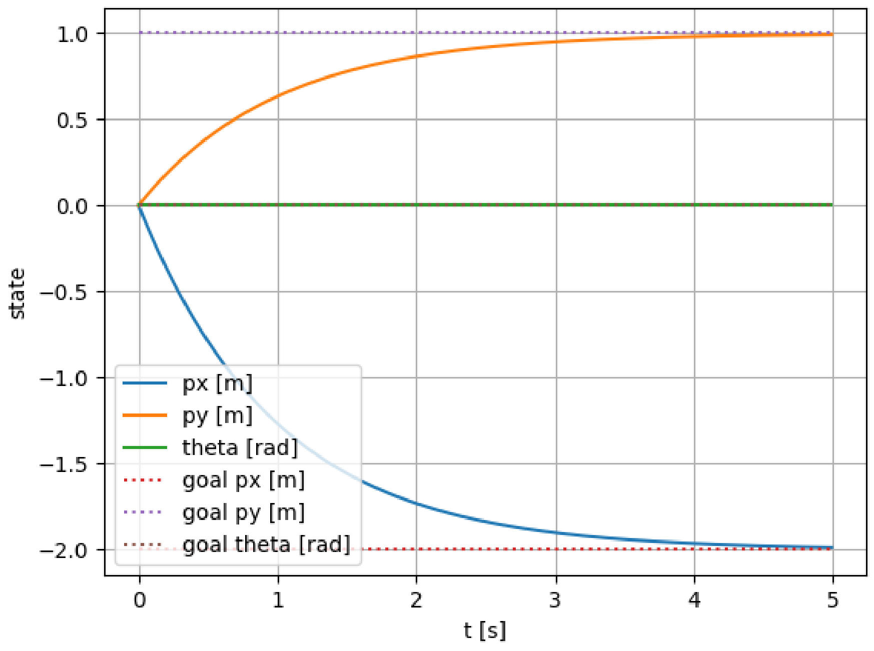

Figure 2: the state of robot and desired goal at k = 1

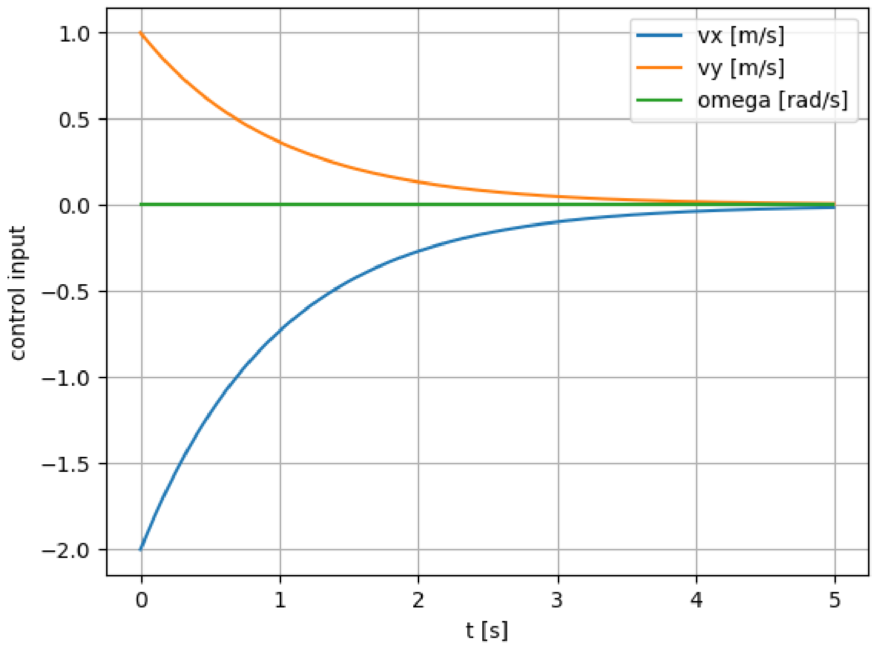

Figure 3: the control input of the robot at k =1

Secondly, k = 2, figure (4) and figure (5) shows the plot of time series of x and u.

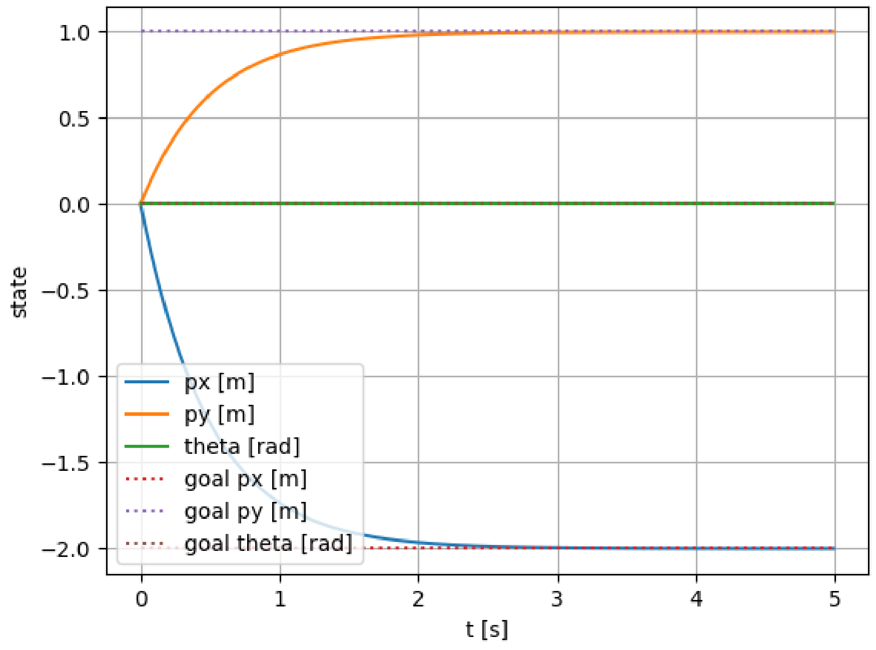

Figure 4: the state of robot and desired goal at k = 2

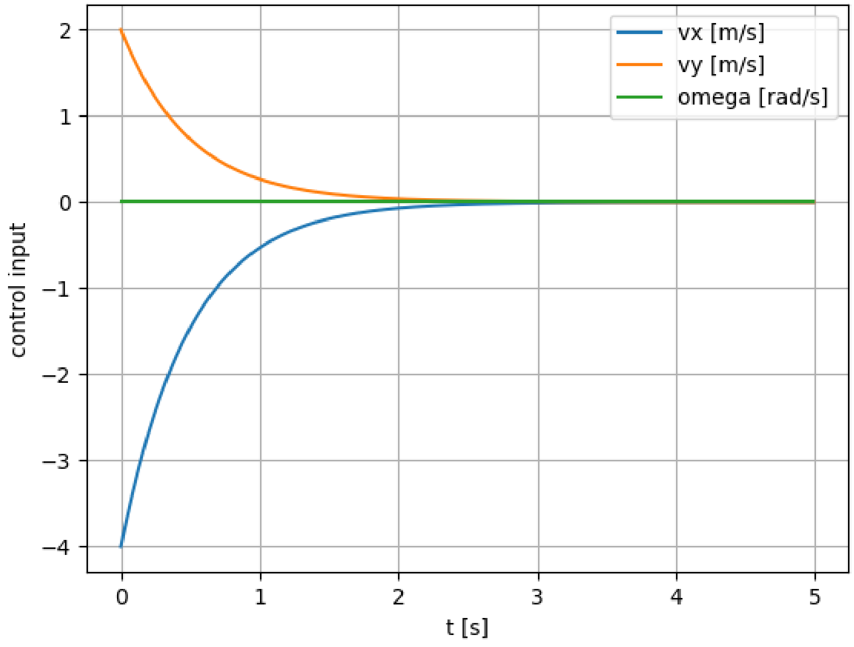

Figure 5: the control input of the robot at k =2

Finally, k = 3, figure (6) and figure (7) shows the plot of time series of x and u.

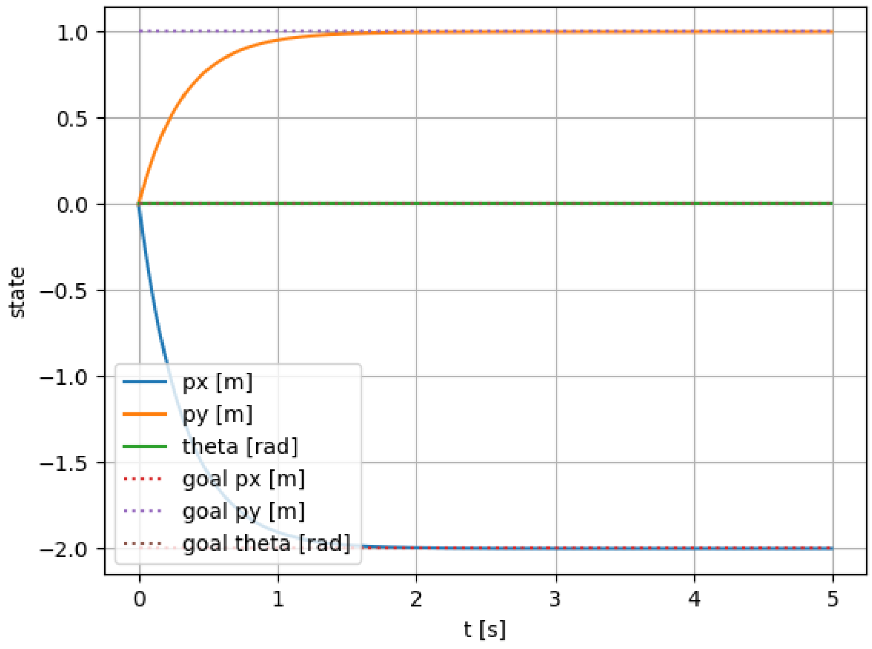

Figure 6: the state of robot and desired goal at k = 3

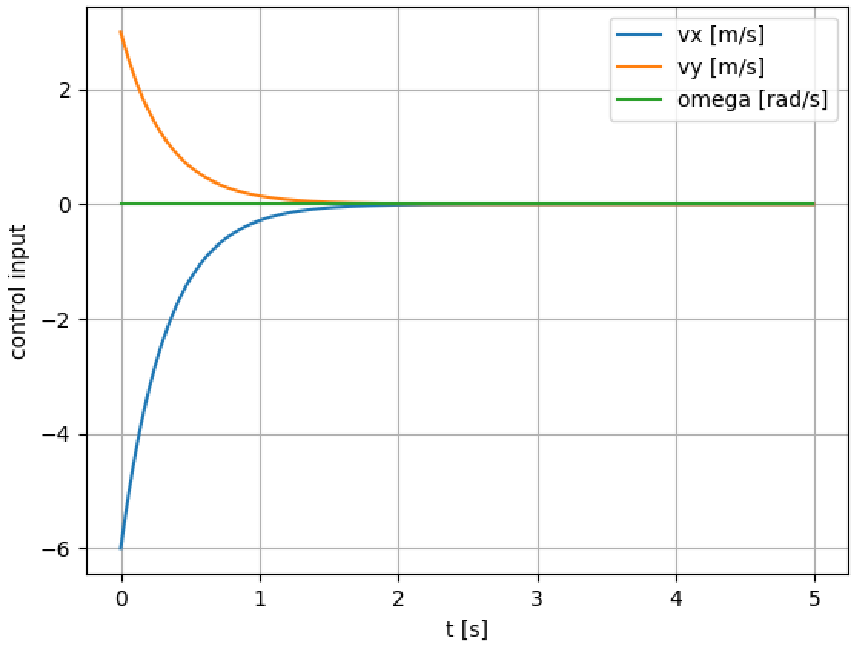

Figure 7: the control input of the robot at k =3

The proportional control with static k was changed with 3 different values, which affected the robot by increasing the speed to the desired goal when k is increased.

Question 2

The proportional control k was changed based on the equation.

$$k = \frac{v_0\left(1 - e^{-\beta||\bar{e}||}\right)}{||\bar{e}||} \tag{4}$$

With $||\bar{e}||$ is the magnitude of the error between the desired state and the current state

$$||\bar{e}|| = \theta_d - \theta \tag{5}$$

In this case, the robot goes to the goal during the 10 second times max.

Firstly, v_0 was set to 3, and β was set to 3, figure (8) and figure (9) show the plot of time series of the robot state x and control input u.

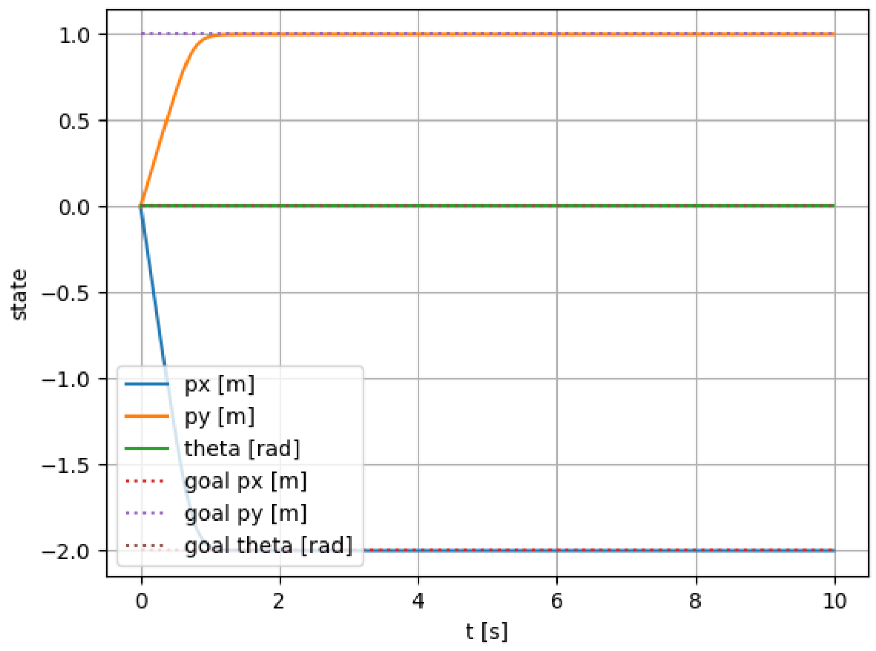

Figure 8: the state of robot and desired goal at v0 = 3 and beta =3

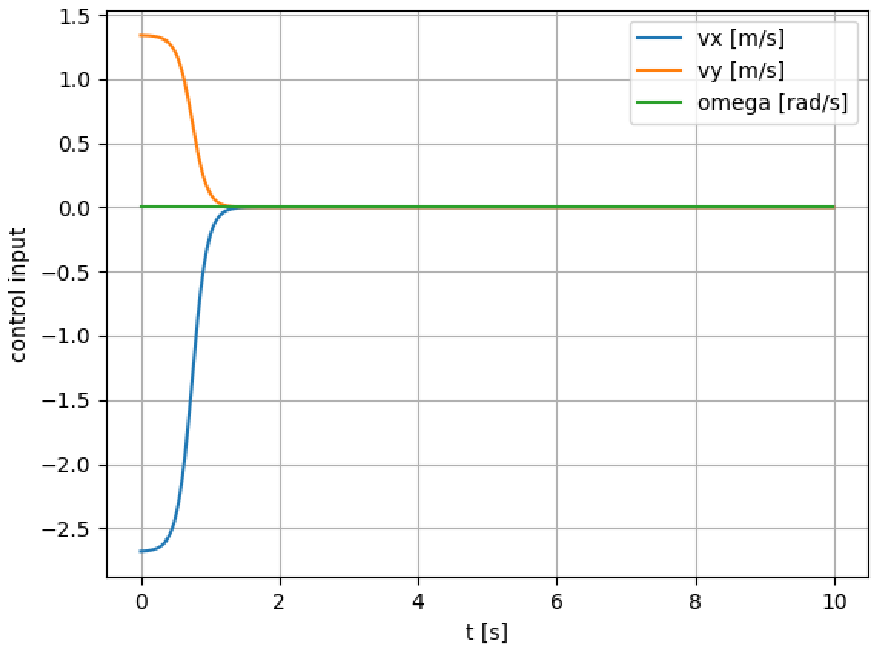

Figure 9: the control input of the robot at v0 = 3 and beta = 3

Dong Le, Niklas Kantele

Secondly, v_0 was set to 0.4, and β was set to 3, figure (10) and figure (11) show the plot of time series of the robot state x and control input u.

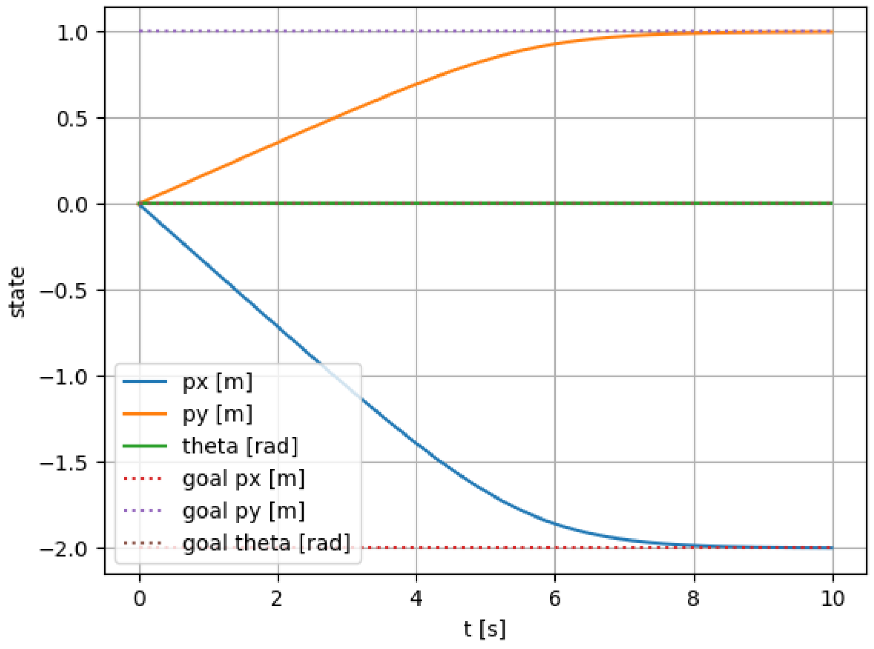

Figure 10: the state of robot and desired goal at v0 = 0.4 and beta =3

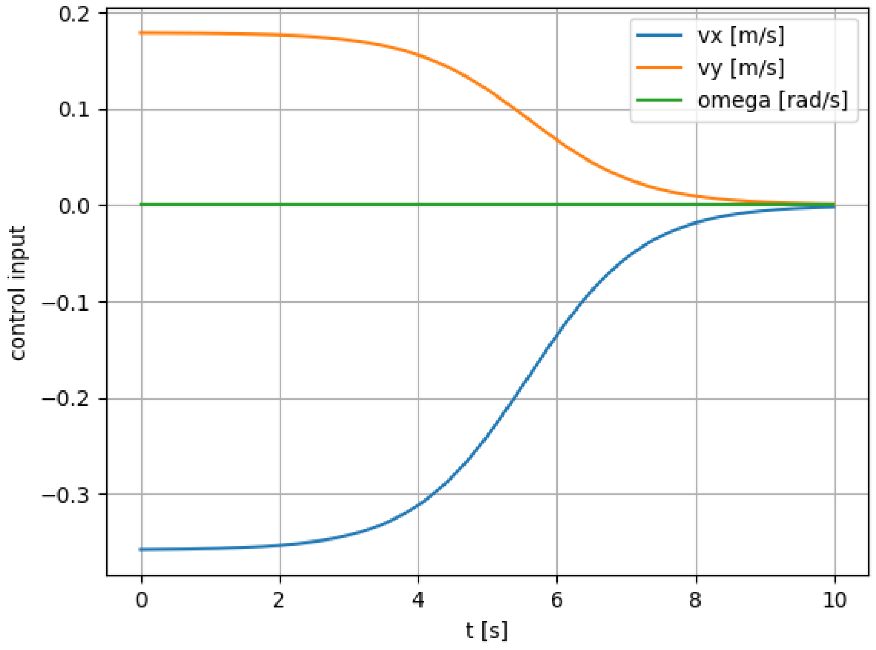

Figure 11: the control input of the robot at v0 = 0.4 and beta = 3

Finally, v_0 was set to 3, and β was set to 0.4, figure (12) and figure (13) show the plot of time series of the robot state x and control input u.

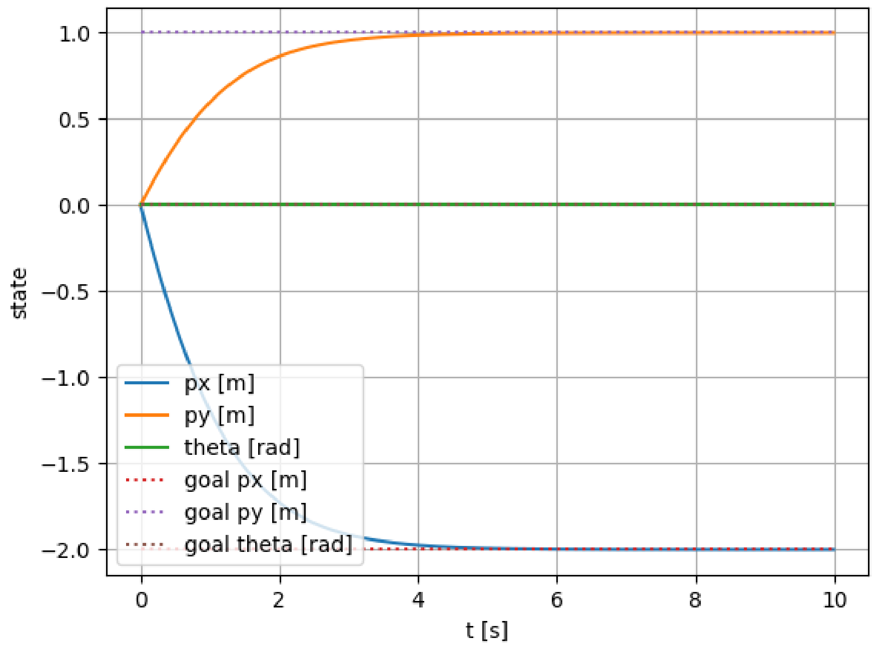

Figure 12: the state of robot and desired goal at v0 = 3 and beta = 0.4

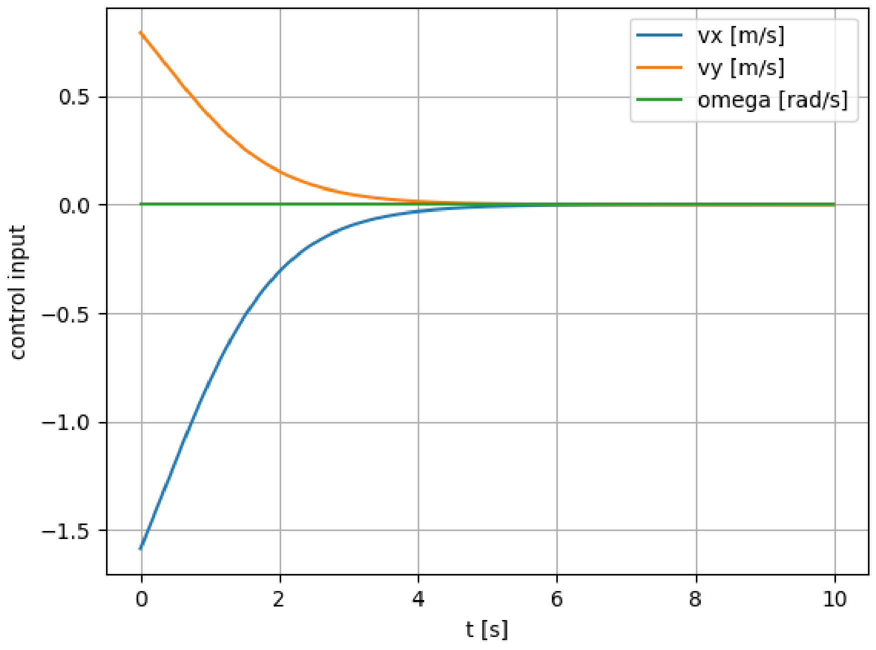

Figure 13: the control input of the robot at v0 = 3 and beta = 0.4

Dong Le, Niklas Kantele

Question 3

The value of k was depended on the value of v_0 and β. From the equation of k, it can be re-written as

$$k = \frac{v_0\left(1 - \frac{1}{e^{\beta\|\bar{e}\|}}\right)}{\|\bar{e}\|} \tag{6}$$

So, we increased the value of v_0 and β, k increased and vice versa, to be specific, v_0 will affect directly to the value of k. Three variations of v_0 and β in the previous task demonstrated the effect of it to control input, but figure (14) shows the robot trajectory was the same in 3 cases.

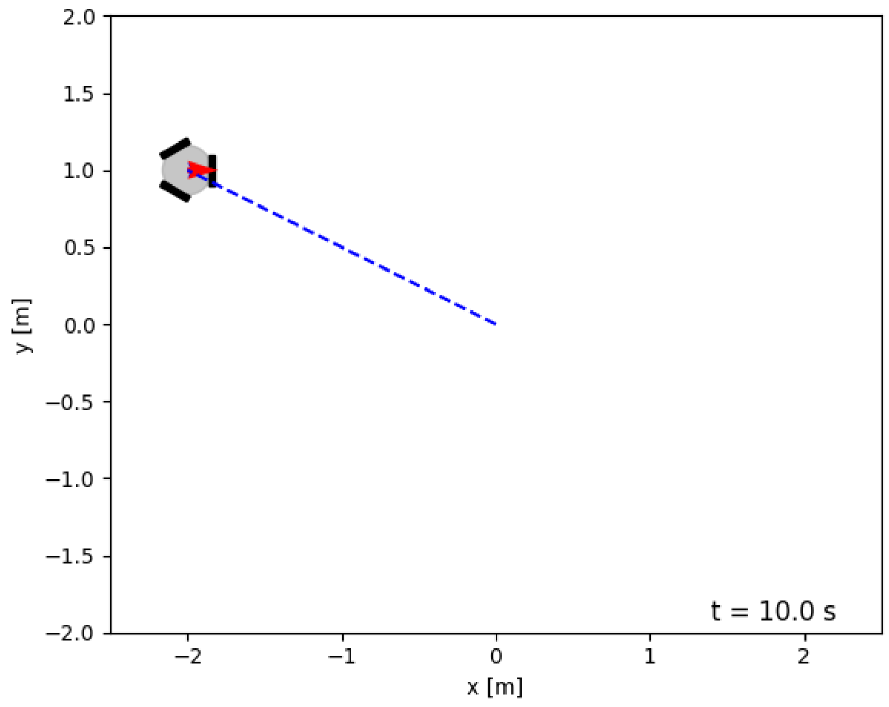

Figure 14: Robot trajectory at time-varying k

With $v_0 = 3$ and $\beta = 3$, the robot went to the desired goal quickly in about 1.5 seconds. The slowest version of robot was $v_0 = 0.4$ and $\beta = 3$, it took the robot nearly 8 seconds to reach the goal. Finally, the last version with $v_0 = 3$ and $\beta = 0.4$ took about 5 seconds.

The choice of v_0 and β affects the proportional control k, so with the big value of v_0 and β, k is big, and the robot travels quickly. Therefore, the value of v_0 should be big and β is small, or vice versa to get a good design.

Problem 2

In this problem, omnidirectional mobile robot was used with the position (p_x, p_y)

$$x = \begin{bmatrix} p_x \\ p_y \\ \theta \end{bmatrix} \tag{7}$$

And the control input:

$$u = \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \tag{8}$$

Initial position $x[0] = [0\ 0\ 0]^T$

The goal $x^d[t] = [-\cos(2t) - \sin(2t)\ \ 0]^T$

The purpose of the task is to track the moving goal, so at the end the robot moves in a circle.

In this case, we applied the feedforward term to design proportional control.

$$u = k(x^d(t) - x) + \dot{x}^d(t)\ with\ k > 0$$

Without feedforward, there is a small tracking error because the feedback was a bit late to react.

For the sake of the program, we set k = 5, Ts = 0.01, and t_max = 10

To calculate the feedforward $\dot{x}^d(t)$, we stored the 2D array of $x^d(t)$ with iteration + 1 rows and 3 columns, so:

$$\dot{x}^d(t) = \frac{x^d(t+1) - x^d(t)}{Ts} \tag{9}$$

Figure (15) shows the trajectory of robot, it started from the initial position, then followed a circle path.

Dong Le, Niklas Kantele

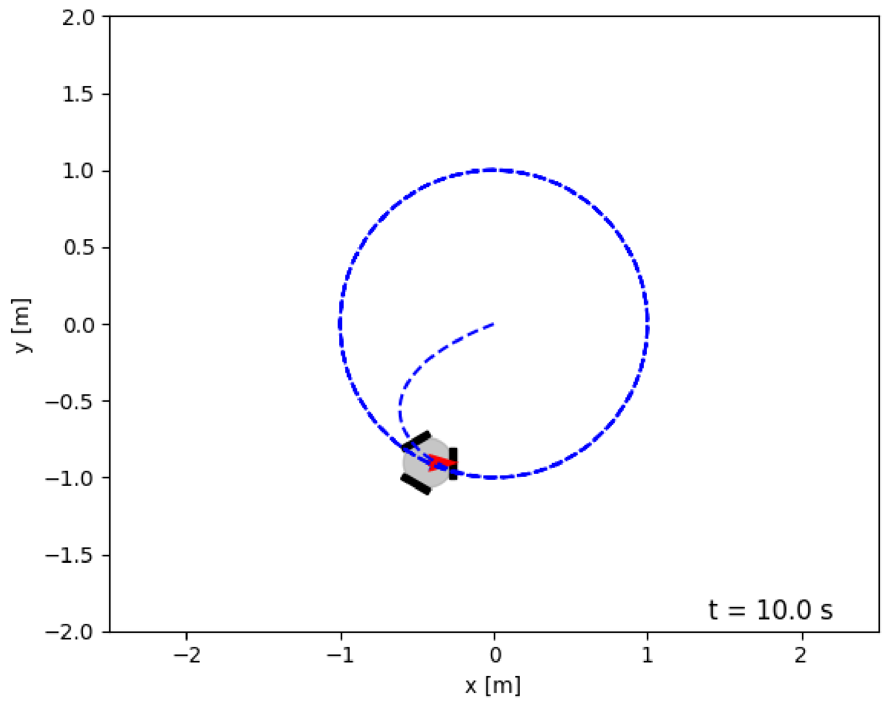

Figure 15: Robot trajectory travels in a circle

Figure (16) shows the control input, vx and vy were changing based on the equation:

$$u = k(x^d(t) - x) + \dot{x}^d(t) \qquad (10)$$

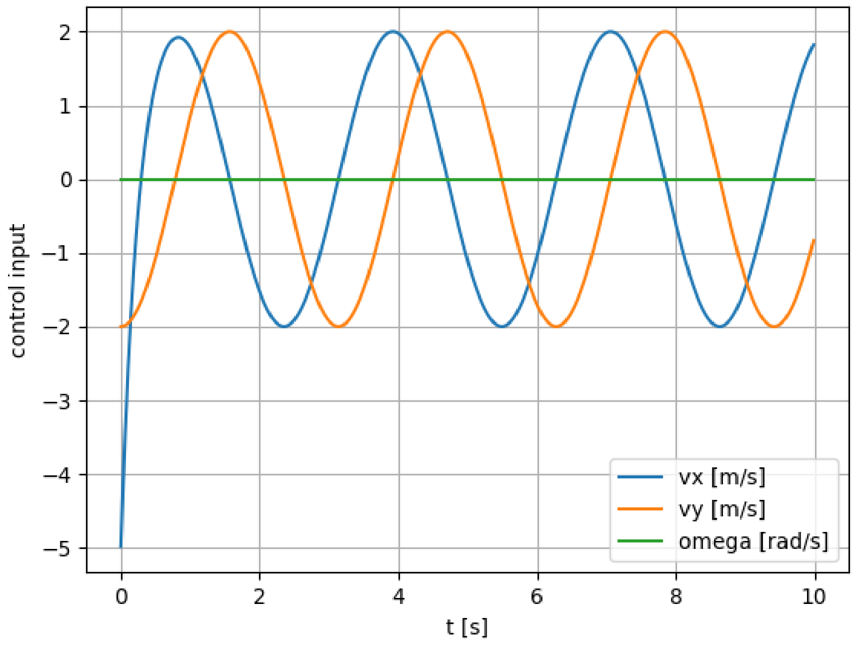

Figure 16: control input with feedforward

Figure (17) shows the state of robot and state of desired goal during time.

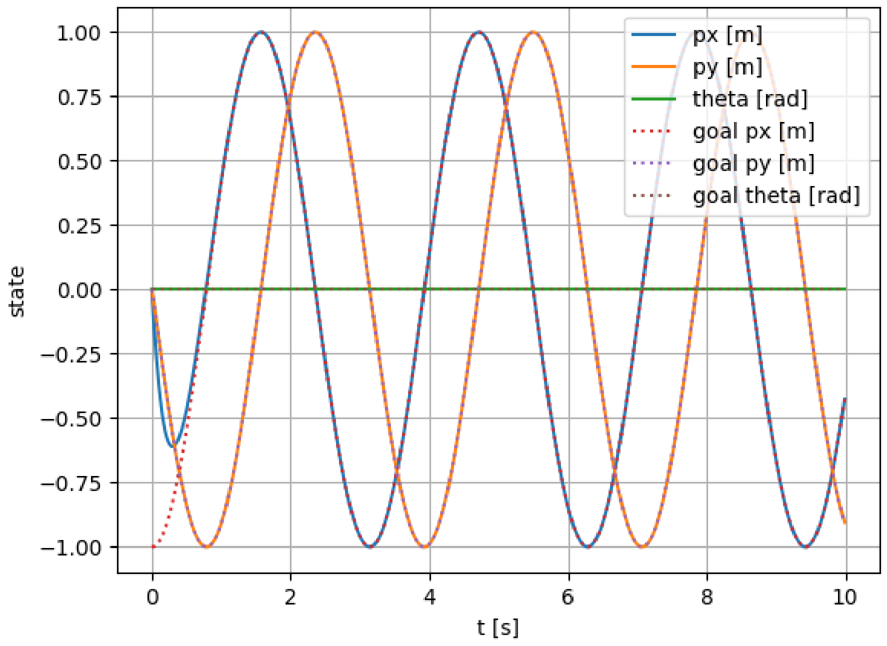

Figure 17: State of the robot and the desired goal against time

Dong Le, Niklas Kantele

Figure (18) shows the error between $x_d - x$,

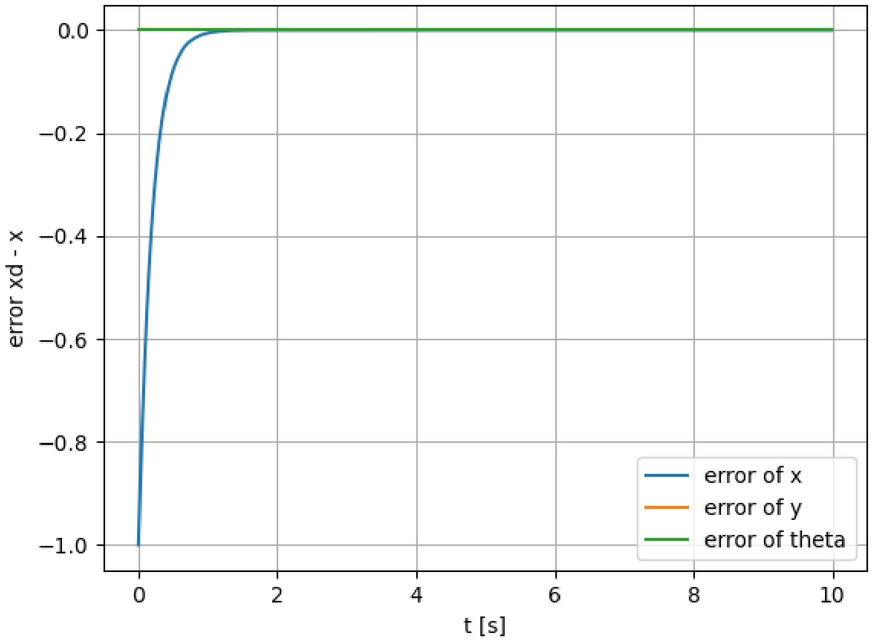

Figure 18: Error between xd and x

Problem 3

In this problem, unicycle mobile robot was used with the position (p_x, p_y)

$$x = \begin{bmatrix} p_x \\ p_y \\ \theta \end{bmatrix} \tag{11}$$

And the control input:

$$u = \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{12}$$

Initial position $x[0] = [0\ 0\ \frac{-\pi}{2}]^T$

The goal $x^d = [-1\ 1\ \frac{\pi}{2}]^T$

In this problem, the velocity v is fixed and the robot only stops if it is very close less than 0.05m to its goal x_d. The robot can only control its angular velocity $\omega = k * \bar{e}_\theta$

With $\bar{e}_\theta$ acts as a control input, $\bar{e}_\theta = \theta_d - \theta$

For the sake of the program, we set k = 3, t_max = 5, Ts = 0.01, and the orientaion of the desired goal is $\frac{\pi}{2}$

Figure (19) displays the trajectory of the robot, which was changing omega during the travel based on the change of robot state.

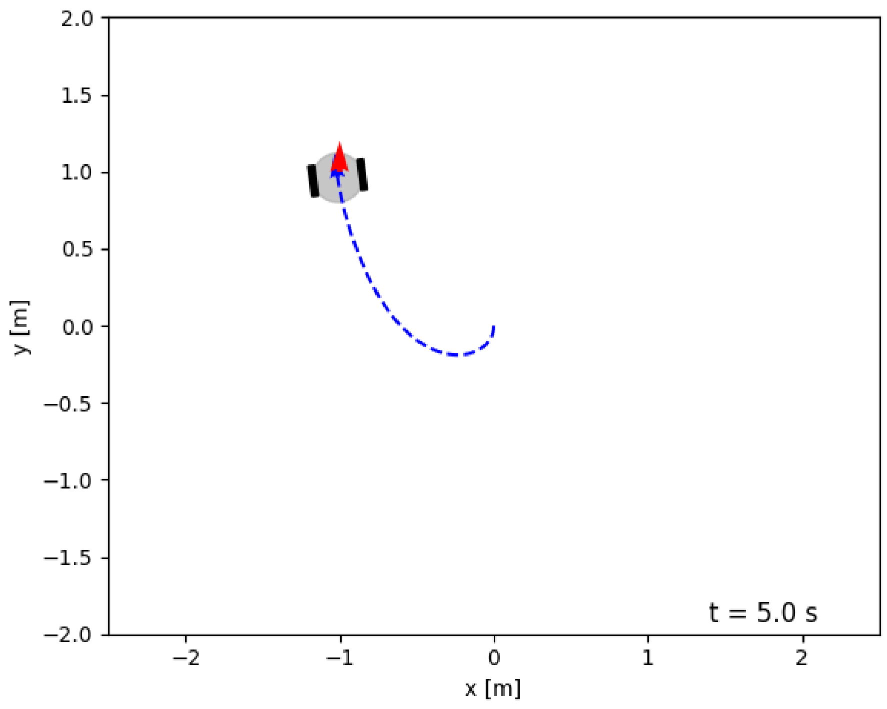

Figure 19: Robot trajectory to the goal

Figure (20) shows the control input u, which based on the a constant v = 1 when distance is more than 0.05m

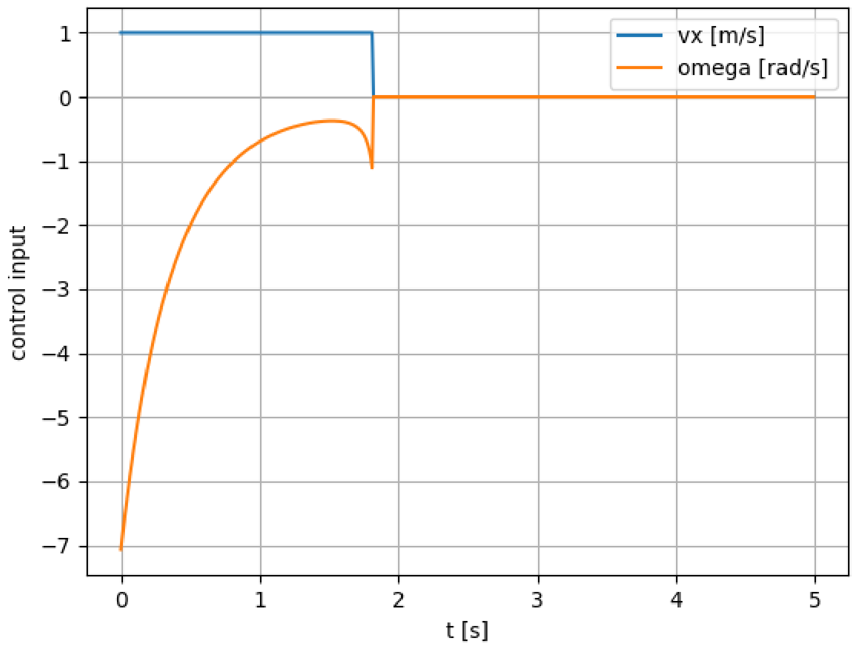

Figure 20: the control input of the robot with v and omega

So, after nearly 2 seconds, the robot went close to the goal, so the control input became 0 to stop.

Figure (21) shows the error of $x_d - x$ against time, because of the theta of the desired goal is not needed to reach, so the error of $\theta_d - \theta$ does not go to 0

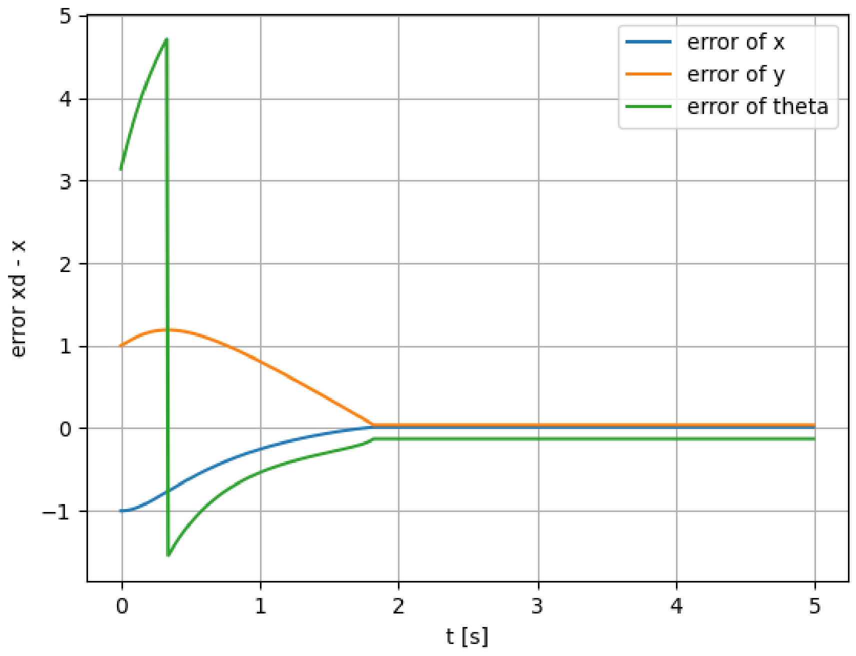

Figure 21: error of desired goal and robot state

Figure (22) shows the state of $x_d - x$ during time, so lines lie on each other because the robot reach the goal.

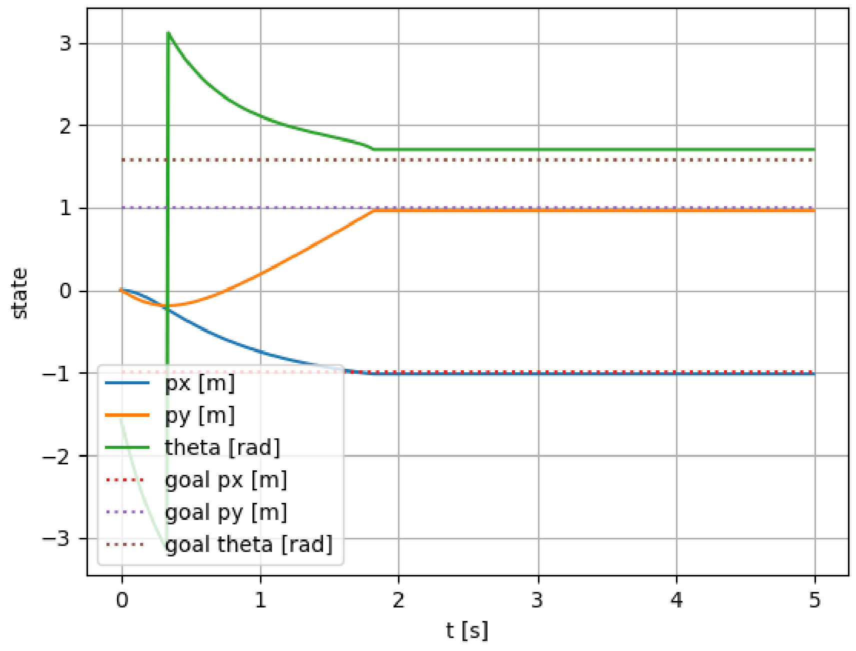

Figure 22: the state of robot and desired goal

During the process, we checked several variants of k, we found out that the smallest possible k is 2.3 to reach the goal. Otherwise, if k is chosen too small, the robot will not be able to change its orientation fast enough to be directed towards the goal.

Dong Le, Niklas Kantele

Conclusion

In the first problem we found out that with static k the robot always went to the goal with the same trajectory, the changes of k will change the speed of the robot to reach the goal. The choice of v_0 and β affects k, therefore, the value of v_0 should be big and β is small, or vice versa to get a good design.

In the second problem we found out that without feedforward, there is a small tracking error because the feedback was late to react. So, by design a forecast in the proportional control based on the goal state will help the system get a perfect tracking.

In the third problem we found out by changing the ω in the proportional control, the robot can reach the desired goal with a constant speed. The smallest possible k is 2.3 to reach the goal, if k is chosen too small, the robot will not be able to change its orientation fast enough to be directed towards the goal.

Appendices

Appendix A

Program code is attached to the zip file.

THIS PAGE LEFT BLANK INTENTIONALLY