Homework 3 in EL2450 Hybrid and Embedded Control Systems

Alexander Ramm
931005
aleram@kth.se

Filip Stenbeck
930702
fstenb@kth.se

Gabriel Andersson Santiago
910706
gabras@kth.se

Anders Levin
910729
anle@kth.se

March 7, 2016

Task 1

By using the formulas given the following equations were derived.

$$u_r = \frac{2(u_\omega + u_\psi)}{2} = u_w + \frac{u_{Psi}}{2} \tag{1}$$

$$u_l = u_\omega - \frac{u_\Psi}{2} \tag{2}$$

Task 2

Rewriting the equations 3 from the problem formulation on the following form:

$$R = \frac{\dot{x}}{u_\omega cos\theta} \tag{3}$$

yields the possibility to calculate a good estimate of the model parameter R. Approximating the speed $\dot{x}$ and then calculating the best R for different values of θ. The same holds for the approximation of L. The equation for R,l is rewritten accordingly.

$$L = \frac{Ru_\Psi}{\dot{\theta}} \tag{4}$$

The rotational speed $\dot{\theta}$ is approximated from the obtained data and L is then calculated in the same say as R. The calculated values of R and L are presented in Table 1 below.

R	L
0.0010085	0.0050935

Task 3

-2

WRONG

$\dot{\theta} = R/Lu_\psi$ won't be asymptotically stable due to that the output signal will oscillate with constant amplitude after a set time. It will not have Zeno behaviour due there is no finite time limit when the output signal is stable.

→ DOES THAT HAPPEN IN THEORY?

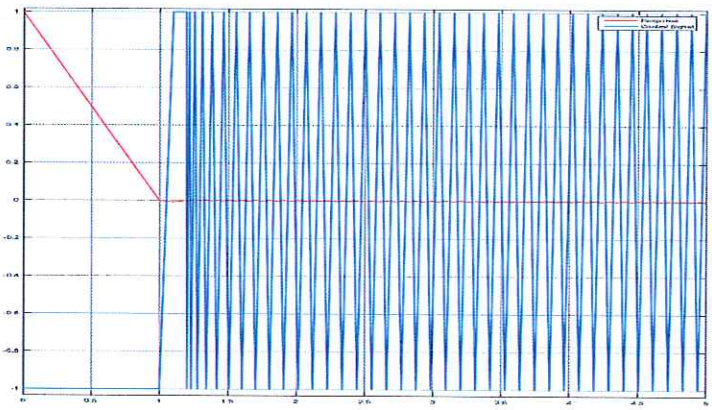


Figure 1: Performance of the controller

Task 4

The system is asymptotically stable due to that the output signal will constantly converge to the desired value. Though by doing this will quicker and quicker reactions from the controller which will not be possible in real life. The system has Zeno behaviour due to infinite switches in finite time.

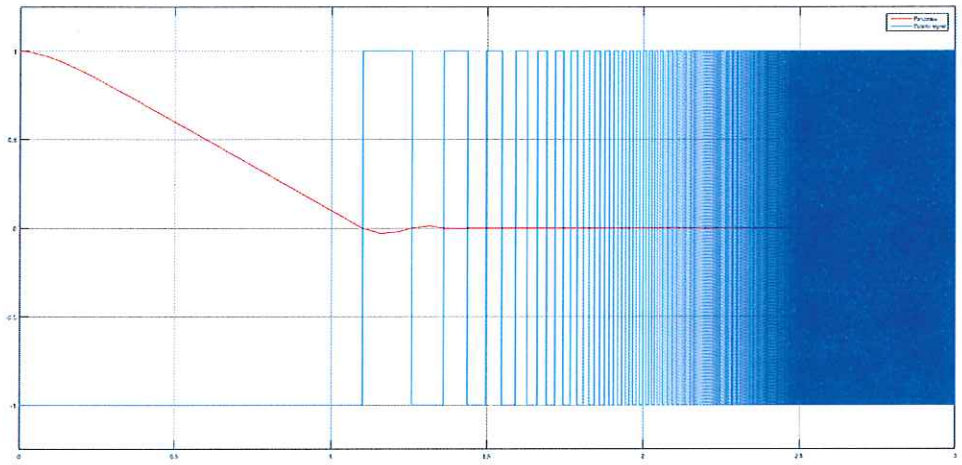


Figure 2: Performance of the controller

Task 5

The system is stable but not asymptotically stable and therefore does also not exhibit Zeno behaviour. We see that the system is not asymptotically stable due to that the output signal does not converge to the desired value but will oscillate with a constant amplitude after infinite time.

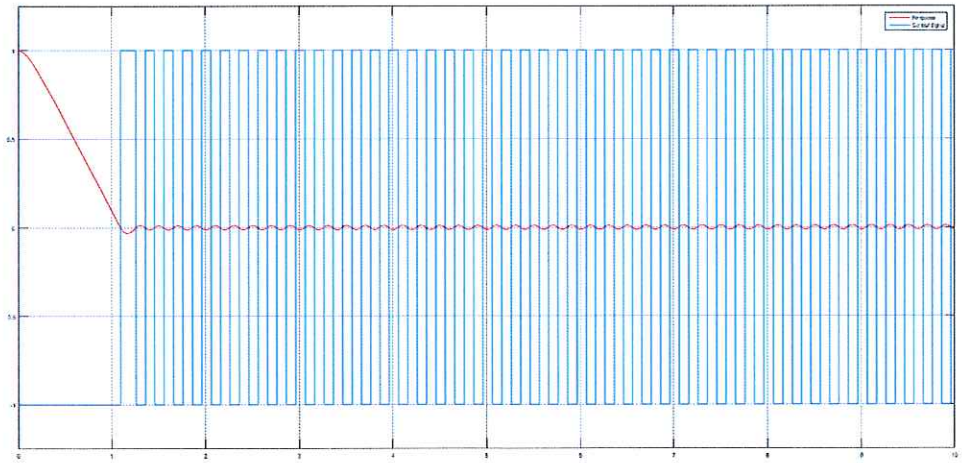


Figure 3: Performance of the controller

Task 6

The discretized system can be seen in Equations (5),(6) and(7)

$$\frac{z-1}{T_s}x[k] = Ru_\omega[k]cos(\theta[k]) \tag{5}$$

$$\frac{z-1}{T_s}y[k] = Ru_\omega[k]sin(\theta[k]) \tag{6}$$

$$\frac{z-1}{T_s}\theta[k] = \frac{R}{L}u_\Psi[k] \tag{7}$$

Task 7

With Euler forward method we have that

$$\theta[h] \approx \theta[k] + \dot{\theta}[k]\tau_s$$

We have $\dot{\theta}$ from equation (3) and control signal u_Ψ in the assignment. This give the following equation for the system.

$$\theta[k+h] \approx \frac{RK_\Psi\tau_s(\theta^R - \theta[k])}{L} + \theta[k]$$

$$\theta[k+h] \approx (1 - \frac{RK_\Psi\tau_s)}{L})\theta[k] + \frac{RK_\Psi\tau_s)}{L}\theta^R$$

This is stable i.e the absolute value of the eigenvalues are less than 1 for the Φ matrix. The eigenvalue is

$$\lambda = |1 - \frac{RK_\Psi\tau_s)}{L}| < 1$$

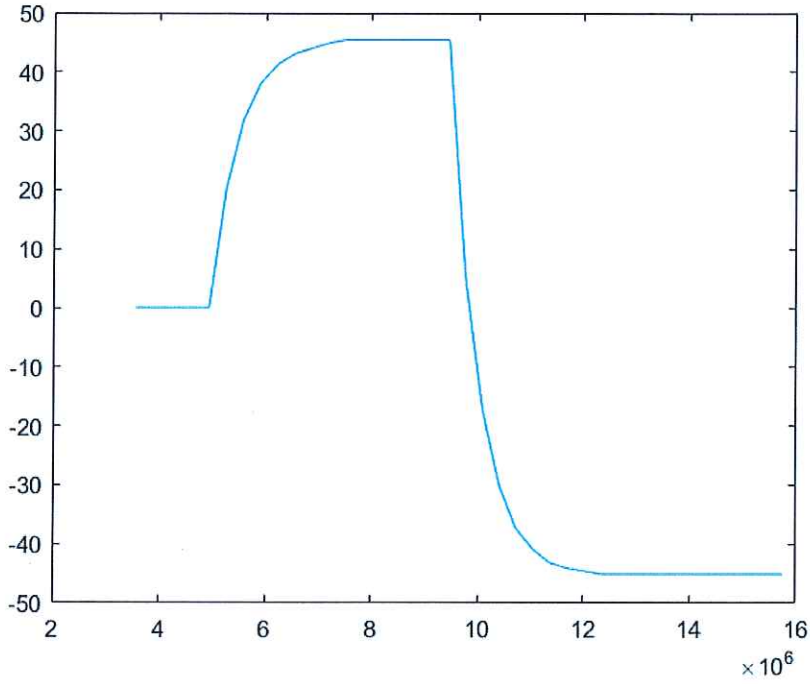
This gives the upper/lower boundary of $0 < K_\Psi < \frac{2L}{R\tau_s}$.

Task 8

It is not possible to reach the goal angle due to we only have a proportional controller and therefore always have a small static error.

K = 0.7*L/R

NOT TRUE



What are you plotting?

Figure 4: Performance of the controller

Task 9

SOMETHING WRONG IN YOUR COMPUTATION

2

With euler forward method we have that for the x variable

$$x[h] \approx x[k] + \dot{x}[k]\tau_s$$

We have $\dot{x}$ from equation (3) and control signal u_ω.

$$u_\omega[k] = K_\omega cos(\theta[k])(x_0 - x[k]) + K_\omega sin(\theta[k])(y_0 - y[k])$$

$$\dot{x[k]} = Ru_\omega[k]cos(\theta[k])$$

$$x[k+h] \approx (1 - K_\omega \tau_s R cos^2(\theta[k]))x[k] + K_1 + \Gamma y[k]$$

Where K_1 a constant of how $x[k+h]$ depends on x_0, y_0 and Γ is how it depends on the position i y direction $y[k]$. Assuming that $y[k]$ could be considered as an input signal we have stability if the absolute value of the eigenvalues in the stability equation is less than 1.

$$|\lambda| = |1 - K_\omega \tau_s R cos^2(\theta[k])| < 1$$

This gives the the upper/lower boundary of $0 < K_\omega < \frac{2}{R\tau_s}$ since the minumum upper boundary is found when $cos^2(\theta[k]) = 1$.

If the controller is analysed in the y direction the following eigenvalues are obtained.

$$|\lambda| = |1 - K_\omega \tau_s R sin^2(\theta[k])| < 1$$

And since the maximum of $sin^2(\theta[k]) = cos^2(\theta[k]) = 1$ gives the same upper and lower bounds.

Task 10

In the general case no due to the direction of the robot is not lined up with the point it wants to go to.

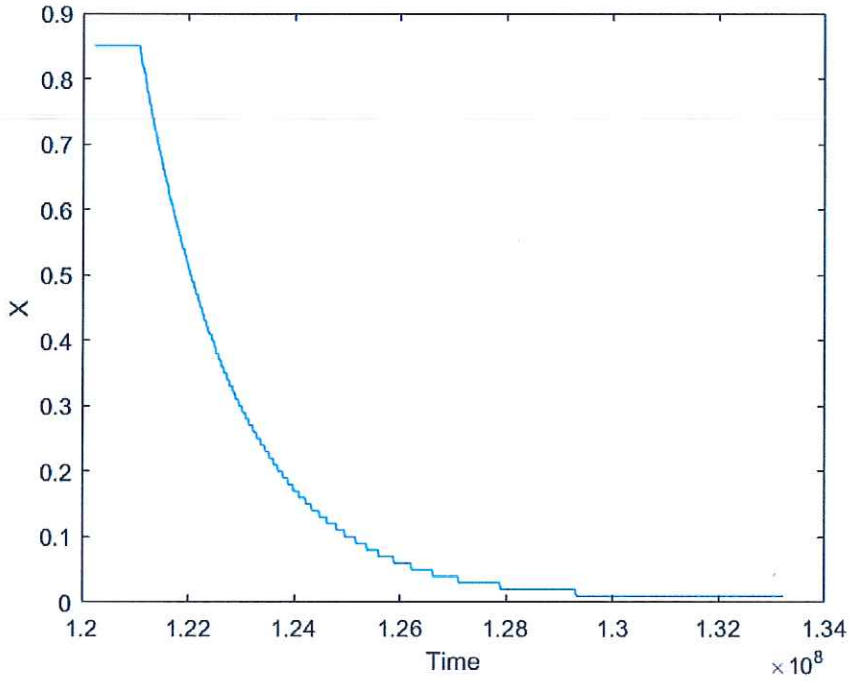


Figure 5: Controller performance for translational movement where the desired point is directly in front of the robot

Task 11

With both controllers enabled we get both the correct angle and keeping the same position. This is kind of dependent on the speed of the controllers that systems are in similar speeds.

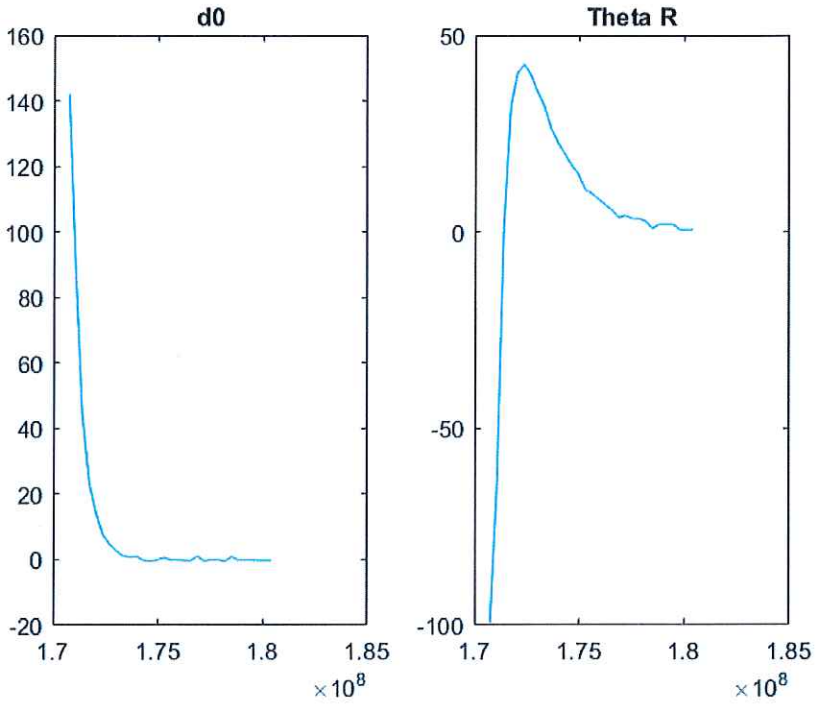


Figure 6: We see that the control signal to the transient controller is zero during the time of the plot as well as the desired θ is achieved.

Task 12 SEE TASK 9

This section is very similar to task 9 whilst we have a slightly different controller thus this will not change the poles of the controller. Again the position approximated with euler forward.

$$x[h] \approx x[k] + \dot{x}[k]\tau_s$$

We have $\dot{x}$ from equation (3) and control signal u_ω.

$$u_\omega[k] = K_\omega cos(\theta[k])(x_g - x[k]) + K_\omega sin(\theta[k])(y_g - y[k])$$

$$\dot{x[k]} = Ru_\omega[k]cos(\theta[k])$$

$$x[k+h] \approx (1 - K_\omega \tau_s R cos^2(\theta[k]))x[k] + K_1 + \Gamma y[k]$$

Where K_1 a constant of how $x[k+h]$ depends on x_g, y_g and Γ is how it depends on the position i y direction $y[k]$. Assuming that $y[k]$ could be considered as an input signal we have stability if the absolute value of the eigenvalues in the stability equation is less than 1.

$$|\lambda| = |1 - K_\omega \tau_s R cos^2(\theta[k])| < 1$$

This gives the the upper/lower boundary of $0 < K_\omega < \frac{2}{R\tau_s}$ since the minumum upper boundary is found when $cos^2(\theta[k]) = 1$. This is just like the controller in task 9 but we control to a different position.

If the controller is analysed in the y direction the following eigenvalues are obtained.

$$|\lambda| = |1 - K_\omega \tau_s R sin^2(\theta[k])| < 1$$

And since the maximum of $sin^2(\theta[k]) = cos^2(\theta[k]) = 1$ gives the same upper and lower bounds.

Task 13

With the control value: K = 30 the performance can be seen in Figure 7.

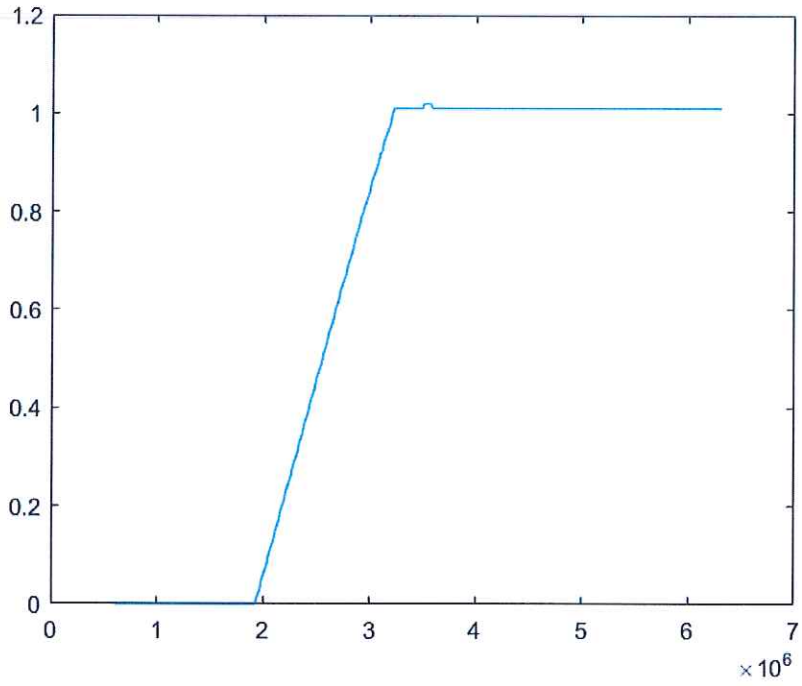


Figure 7: Controller performance for translational movement

It is not possible to arrive at x_g y_g exactly, mainly because the performance depends on the aim by the rotational controllers, both before and during travel.

↳ WHAT HAPPENS IF THE ROBOT STARTS WITH THE RIGHT ANGLE?

Task 14

With euler forward method we have that

$$\theta[h] \approx \theta[k] + \dot{\theta}[k]\tau_s$$

We have $\dot{\theta}$ from equation (3) and control signal u_Ψ in the assignment. This give the following equation for the system.

$$\theta[k+h] \approx \frac{pRK_\Psi\tau_s(\theta_g - \theta[k])}{L} + \theta[k]$$

$$\theta[k+h] \approx (1 - \frac{pRK_\Psi\tau_s}{L})\theta[k] + \frac{RK_\Psi\tau_s}{L}\theta_g$$

This is stable i the absolute value of the eigenvalues are less than 1 for the Φ matrix. The eigenvalue is

$$\lambda = |1 - \frac{pRK_\Psi \tau_s}{L}| < 1$$

This gives the the upper/lower boundary of K_Ψ due to.

$$\frac{pRK_\Psi \tau_s}{L} > 0$$

and

$$\frac{pRK_\Psi \tau_s}{L} < 2$$

and therefore $0 < K_\Psi < \frac{2L}{pR\tau_s}$.

Task 15

K_Ψ depends on the chosem p), not the other way around.

Given the inequality in task 14 we get that the maximum value for the value p to keep the system stable is $p < \frac{2L}{K_\Psi R\tau_s}$ given a controller gain K_Ψ. Generally the larger the value set on p is, the more punishing a angle error becomes. The optimal value seemed to be approximately 0.5 meters given our value K_Ψ.

Task 16

We see in figure 8 that the error goes to zero, meaning that we can maintain an exact angle with no error. This is when only rotation, completely disabling the translational controller.

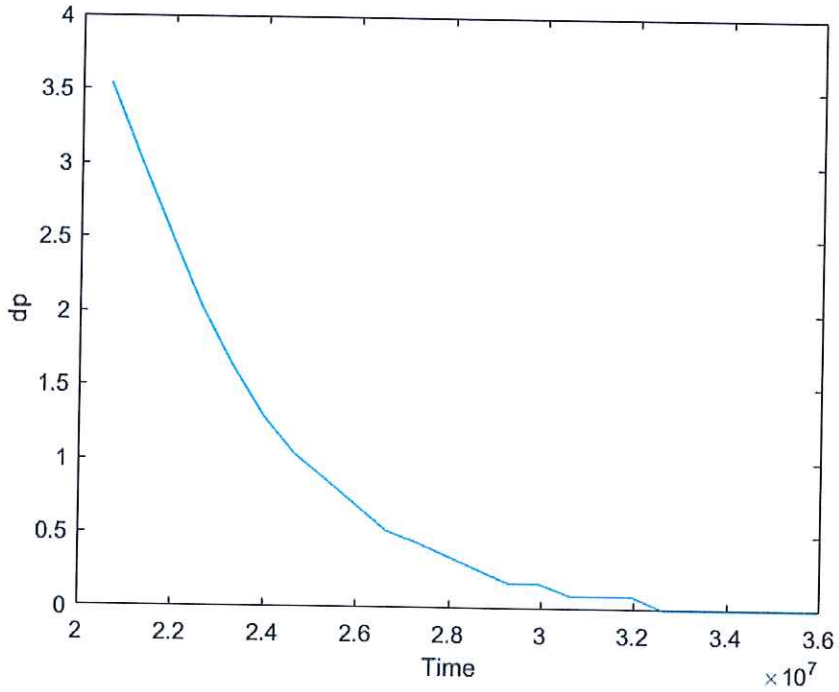


Figure 8: Plot of d_p staying at 0

Task 17

Now the line following controller gets disturbed by the translational controller and vice versa. We can see the performance in figure 9. We can see that they differ in behaviour form the case whne only one controller is active at the time.

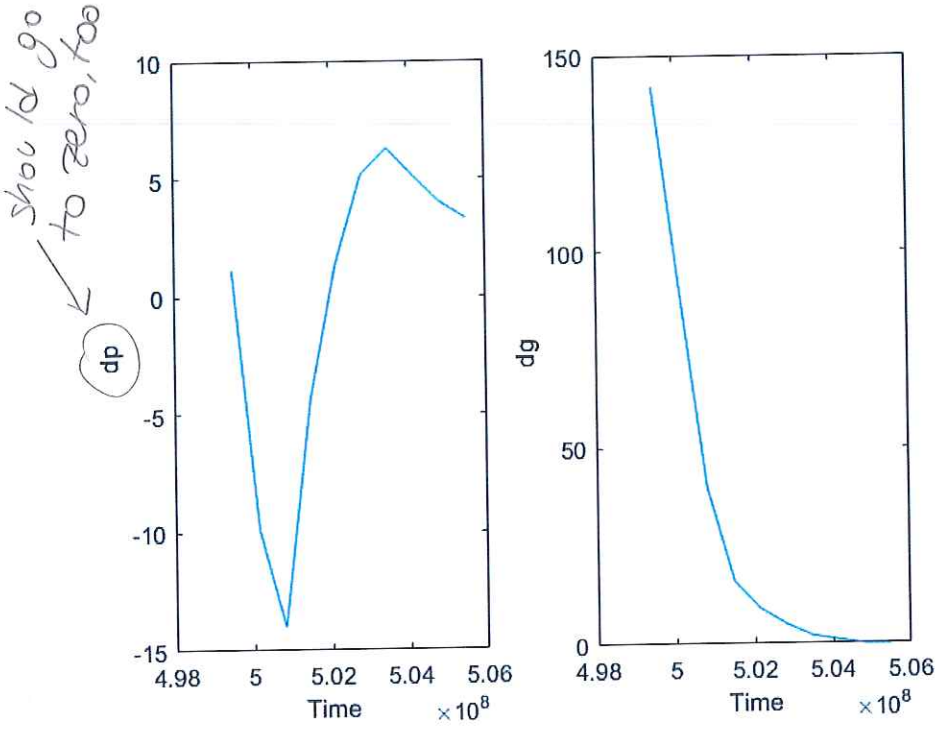


Figure 9: Plot of p_d and p_g when desired point is directly ahead of the robot

Task 18

The hybrid automata can be represented as follows: $Q = \{Rotate, Line, Wait\}$
$X = \mathbb{R}^3$
$Init = (Rotation, (x_0, y_0, \theta_0))$
$f(Rotation, (x, y, \theta)) = (Ru_\omega * cos(\theta), Ru_\omega * sin(\theta), \frac{R}{L}u_\Psi)$
$f(Line, (x, y, \theta)) = (Ru_\omega * cos(\theta), Ru_\omega * sin(\theta), \frac{R}{L}u_\Psi)$
$f(Wait, (x, y, \theta)) = (0, 0, 0)$

$D(Rotation) = \{(x, y, \theta)\epsilon\mathbb{R}, -180 < \theta \leq 180\}$
$D(Line) = \{(x, y, \theta)\epsilon\mathbb{R}, -180 < \theta \leq 180\}$
$D(Wait) = \{(x, y, \theta)\epsilon\mathbb{R}, -180 < \theta \leq 180\}$

$E = \{(Rotation, line), (Line, Wait), (Line, Rotation), (Wait, Rotation)\}$

$G(Rotation, Line) = \{(x, y, \theta)\epsilon\mathbb{R} : |\theta^R - \theta| < a, |x_0 - x| < b, |y_0 - y| < c\}$
$G(Line, Rotation) = \{(x, y, \theta)\epsilon\mathbb{R} : |x_g - x| < d, |y_g - y| < e\}$

$R((Rotation, Line), ((x, y, \theta)\epsilon\mathbb{R})) = R((Line, Wait), ((x, y, \theta)\epsilon\mathbb{R})) =$

$= R((Line, Rotation), ((x, y, \theta)\epsilon\mathbb{R})) = R((Wait, Rotation), ((x, y, \theta)\epsilon\mathbb{R})) = \{(x, y, \theta)\epsilon\mathbb{R}\}$

the same model as ain figure 10.

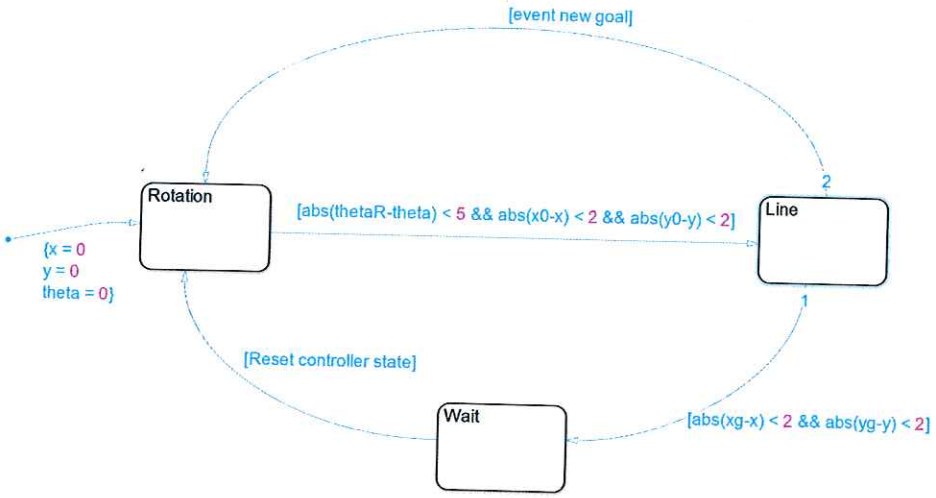


Figure 10: Statechart of the hybrid controller with guards

Task 19

The state changes when we fulfil the guards in the state machine. When we are in the zero state only the angle changes, while x and y errors remains constant, while in state one both x, y and θ error changes with time. We never reach the state two in this simulation, probably because our guards are a bit harsh or not letting the robot settle. If we would reach state two neither position or angle would change. But we can see the effect of the guard when changing form state zero to state one, once θ is less than ± 5 degrees (and the robots position is close to the initial position) the state changes to state one.

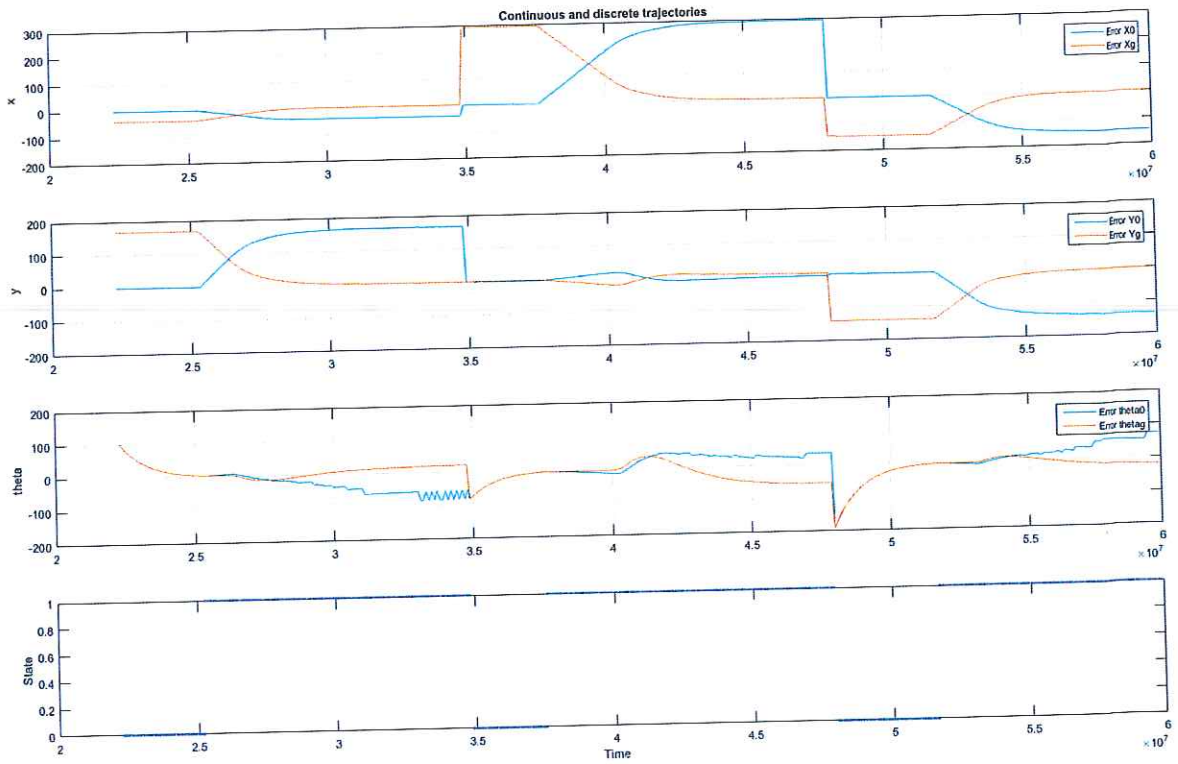


Figure 11: Plot of errors and state trajectories when the full hybrid controller is running.

Task 20

Done

Task 21

It was pretty hard to reach the exact point manually. This was probably due to the short delay in communication and that you needed to hold down the button for a short time for the robot to react.

Task 22

The path of the robot compared to the simulation with the same controllers can be seen in Figure 12. This is due to many reasons, here are a few: The smallest input given by our controllers might not be able to move the robot thus not having any effect. This might further be disturbed by static friction in the wheels and the robot battery charge level. The motion capture seem very accurate but might still cause some errors int both positioning and angle perception. We can see in figure 12 that the aiming in state zero is not ass good as in our simulations thus making the robot more reliant on the state one controllers the manage to achieve its goal state.

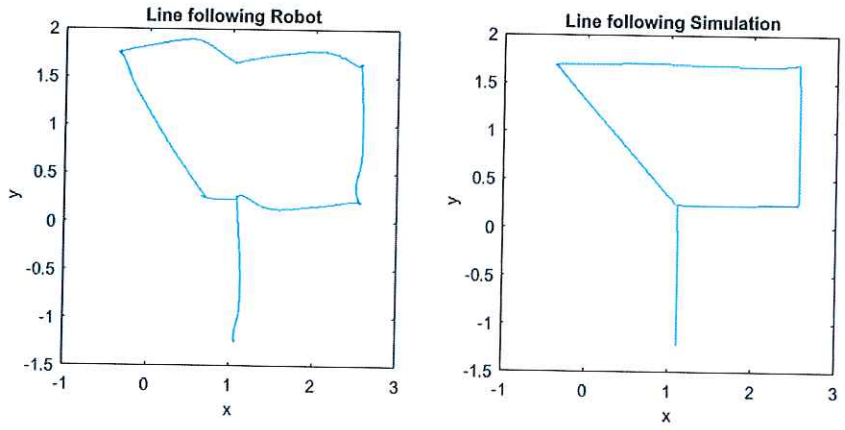


Figure 12: Comparison of line following between the real robot and the simulation

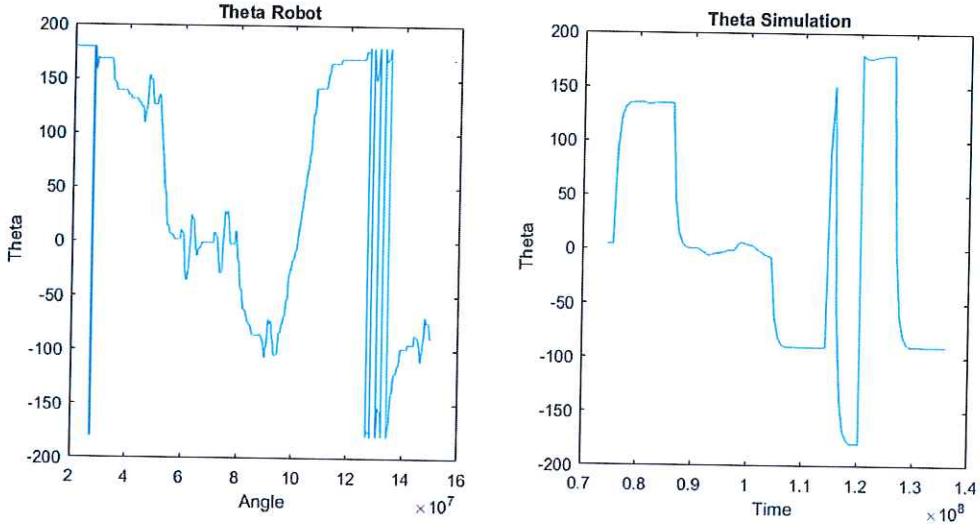


Figure 13: Comparison of θ between the real robot and the simulation