Homework 3 (Tasks 1-19) in EL2450 Hybrid and Embedded Control Systems

Ching-an Wu
9210110632
cawu@kth.se

Federica Ferro
9508085942
fferro@kth.se

Simona Gugliermo
9512205023
simonagu@kth.se

Long Wang
9201083632
longw@kth.se

February 27, 2018

Task 1

The robot can be modelled as

$$v = \frac{u_r + u_l}{2} \tag{1}$$

$$\omega = u_r - u_l \tag{2}$$

where v is the input to translate the robot, while ω to translate it.
Hence, the left and right wheel input, calculated from (2) and (3), are

$$u_r = v + \frac{\omega}{2}$$

$$u_l = v - \frac{\omega}{2}$$

Task 2

A Transition System is defined as

$$\tau = (S, S_0, \Sigma, \longrightarrow_1, \Pi, \mathcal{L})$$

where

- The set of states is
$$S = \{R_1, R_2, ..., R_{36}\}$$
- The set of initial states is
$$S_0 = \{R_6\}$$

- The set of actions is

$$\Sigma = \{Left, Right, Forward, Backward\}$$

- Some transition relations are
  - $\rightarrow = \{R_1, forward, R_2\}$
  - $\rightarrow = \{R_2, backward, R_1\}$
  - $\rightarrow = \{R_1, right, R_{12}\}$
  - $\rightarrow = \{R_{12}, left, R_1\}$
  - $\rightarrow = \{R_8, forward, R_7\}$
  - $\rightarrow = \{R_8, backward, R_9\}$
  - $\rightarrow = \{R_8, right, R_{17}\}$
  - $\rightarrow = \{R_8, left, R_5\}$

  It is important to notice that:
  - from states $\{R_1, R_2, R_3, R_4, R_5, R_6\}$ the event "left" is not allowed.
  - from states $\{R_{31}, R_{32}, R_{33}, R_{34}, R_{35}, R_{36}\}$ the event "right" is not allowed.
  - from states $\{R_6, R_7, R_{18}, R_{19}, R_{30}, R_{31}\}$ the event "forward" is not allowed.
  - from states $\{R_1, R_{12}, R_{13}, R_{24}, R_{25}, R_{36}\}$ the event "backward" is not allowed.

- The set of atomic propositions is

$$\Pi = \{\emptyset, red, green, yellow, obs\}$$

  where "obs" means obstacle

- The labeling function is

$\mathcal{L}(R_1) = red$	$\mathcal{L}(R_2) = green$	$\mathcal{L}(R_3) = obs$	$\mathcal{L}(R_4) = \emptyset$
$\mathcal{L}(R_5) = \emptyset$	$\mathcal{L}(R_6) = green$	$\mathcal{L}(R_7) = blue$	$\mathcal{L}(R_8) = obs, red$
$\mathcal{L}(R_9) = green$	$\mathcal{L}(R_{10}) = yellow$	$\mathcal{L}(R_{11}) = obs, red$	$\mathcal{L}(R_{12}) = \emptyset$
$\mathcal{L}(R_{13}) = \emptyset$	$\mathcal{L}(R_{14}) = \emptyset$	$\mathcal{L}(R_{15}) = \emptyset$	$\mathcal{L}(R_{16}) = \emptyset$
$\mathcal{L}(R_{17}) = \emptyset$	$\mathcal{L}(R_{18}) = \emptyset$	$\mathcal{L}(R_{19}) = obs$	$\mathcal{L}(R_{20}) = red$
$\mathcal{L}(R_{21}) = blue$	$\mathcal{L}(R_{22}) = obs$	$\mathcal{L}(R_{23}) = obs$	$\mathcal{L}(R_{24}) = red$
$\mathcal{L}(R_{25}) = green$	$\mathcal{L}(R_{26}) = obs, red$	$\mathcal{L}(R_{27}) = \emptyset$	$\mathcal{L}(R_{28}) = \emptyset$
$\mathcal{L}(R_{29}) = obs, blue$	$\mathcal{L}(R_{30}) = \emptyset$	$\mathcal{L}(R_{31}) = obs, red, green$	$\mathcal{L}(R_{32}) = red$
$\mathcal{L}(R_{33}) = \emptyset$	$\mathcal{L}(R_{34}) = \emptyset$	$\mathcal{L}(R_{35}) = obs$	$\mathcal{L}(R_{36}) = red$

As it is possible to notice in Figures 1 and 2 the the number K of regions is 36 and the dimensions of each regions are $(0.5 \times 0.5)m^2$.

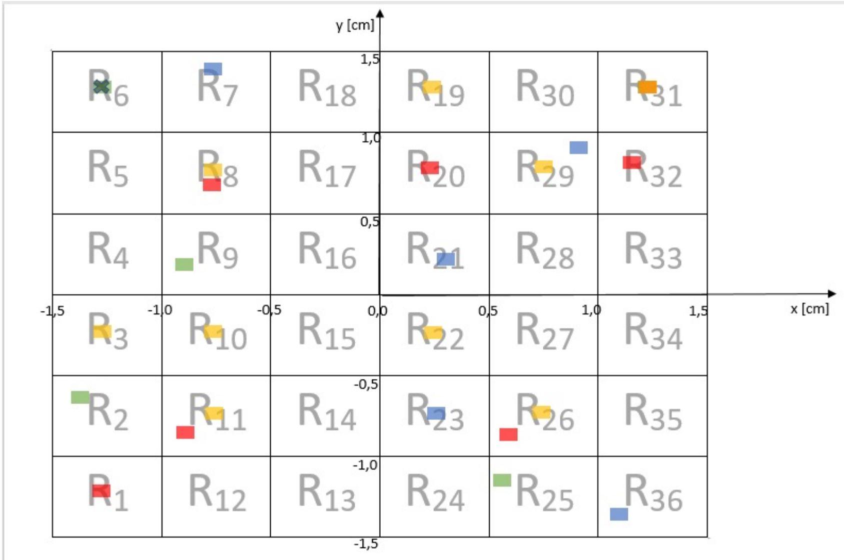

Figure 1: Workspace discretization in rectangular regions in which the obstacles are represented by yellow square

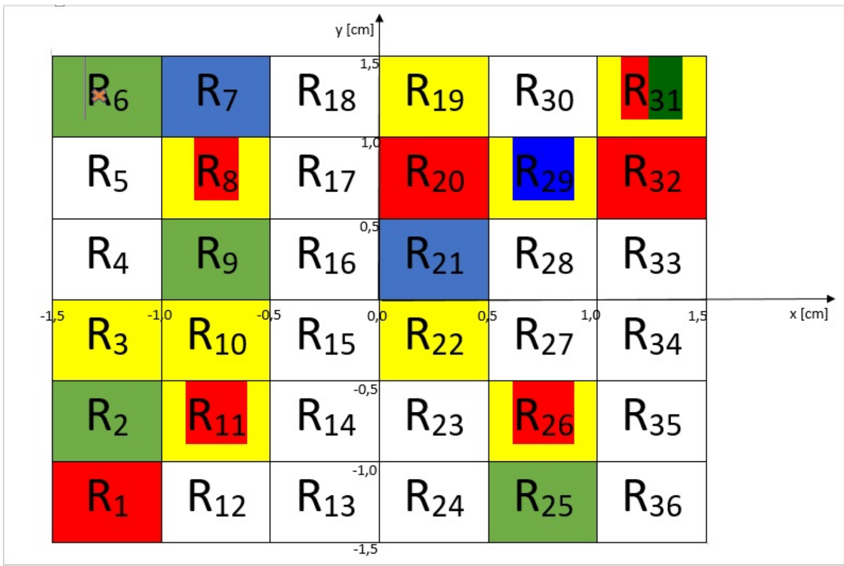

Figure 2: Workspace discretization in rectangular regions

Task 3

In order to satify the desidered robot behavior a path can be:

$$R_6, R_7, R_{18}, R_{17}(R_{20}, R_{21})^*$$

In Figure 3 the desidered path is indicated by the dashed orange lines.

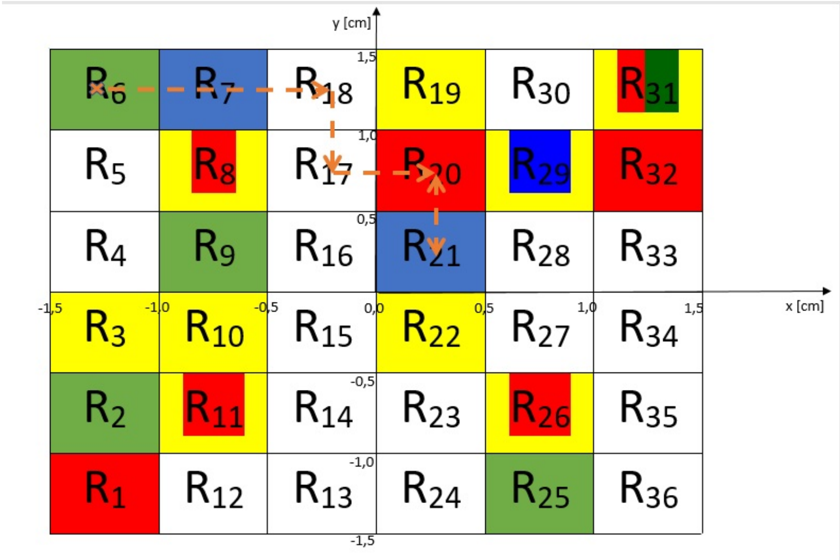

Figure 3: Path that satisfies the desidered behavior

Task 4

The reason is that if we enable both the rotation controller and the go-to-goal controller simultaneously, the trajectory of the robot might be a curve, which incur the possibility of visiting some unintended regions.

Task 5

We know that in the rotational controller,

$$\dot{\theta} = \frac{R}{L}\omega$$

In a discrete time system whose sampling time $h = 1\mu s$,

$$\dot{\theta} = \theta[k+1] - \theta[k]$$

The proportional controller is

$$\omega[k] = K_\Psi(\theta^R - \theta[k])$$

So

$$\theta[k+1] = \theta[k] + \dot{\theta} = \theta[k] + h\frac{R}{L}K_\Psi(\theta^R - \theta[k])$$

It can be expressed as

$$\theta[k+1] - \theta^R = (\theta[k] - \theta^R)(1 - h\frac{R}{L}K_\Psi)$$

In order to make $\theta[k]$ approach θ^R asymptotically, we need to fulfill the following condition

$$|1 - h\frac{R}{L}K_\Psi| < 1$$

We can get

$$0 < K_\Psi < \frac{2L}{Rh}$$

Task 6

It is possible to maintain $\theta[k]$ at θ^R with smaller and smaller errors according to the chosen K_Ψ. The simulation of the controller $\omega[k] = K_\Psi(\theta^R - \theta[k])$ is shown in figure 4. In the simulation, we set desired angle $\theta^R = 90°$ and $K_\Psi = 3$. We can see that the real angle θ approaches the desired angle θ_R and stays the same as θ^R finally. In the practical situation, the robot will rotate to the target orientation and then stop rotating.

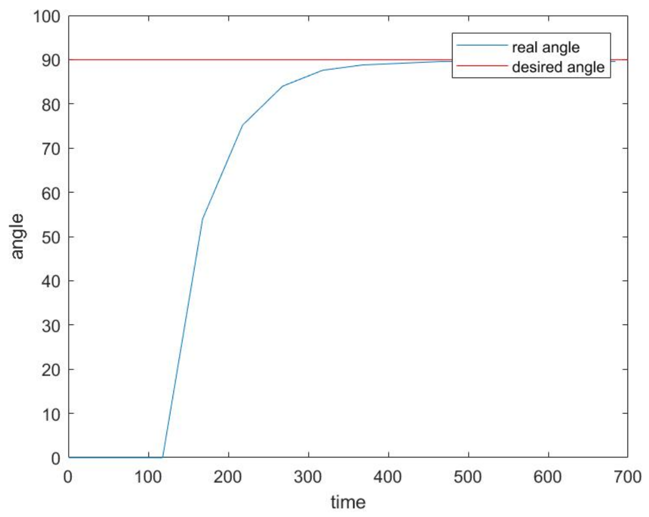

Figure 4: The simulation of the controller $\omega[k] = K_\Psi(\theta^R - \theta[k])$

Task 7

From the instruction, we know that in the continuous system,

$$d_0 = \begin{bmatrix} cos\theta & sin\theta \end{bmatrix} \begin{bmatrix} x_0 - x \\ y_0 - y \end{bmatrix} = (x_0 - x)cos\theta + (y_0 - y)sin\theta$$

Now we differentiate both sides. As θ does not change, we can get

$$\dot{d}_0 = -\dot{x}cos\theta - \dot{y}sin\theta$$

From the instruction, we know $\dot{x} = Rvcos\theta, \dot{y} = Rvsin\theta$. Then

$$\dot{d}_0 = -Rvcos\theta * cos\theta - Rvsin\theta * sin\theta = -Rv$$

In the discrete system,

$$d_0[k+1] = d_0[k] - hRv[k]$$

We know $v[k] = K_\omega d_0[k]$, so

$$d_0[k+1] = d_0[k] - hR * K_\omega d_0[k] = (1 - hRK_\omega)d_0[k]$$

In order to make $d_0[k]$ converge to 0, we need to fulfill the condition $|1 - hRK_\omega| < 1$. It is easy to get

$$0 < K_\omega < \frac{2}{Rh}$$

Task 8

It is possible to maintain $[x[k], y[k]]$ at $[x_0, y_0]$ exactly in certain situations, but not always. We set $K_\omega = 500$ and the initial position $(x_0, y_0) = (0, 0)$. In the first simulation, we make $(x, y) = (1, 0)$. The result is shown in figure 5. It is easy to see that the robot goes from $(x, y) = (1, 0)$ to $(x_0, y_0) = (0, 0)$ exactly . In the second simulation, we make $(x, y) = (1, 1)$. The result is shown in figure 6. It is easy to see that the robot goes from $(x, y) = (1, 1)$ to $(0, 1)$ instead of the initial position $(0, 0)$.
Now we can conclude that if the initial orientation is aligned with the the line connecting the initial position and real position, the robot will go to and maintain at the initial position. While if the initial orientation is not aligned with the the line connecting the initial position and real position, the robot can't go to and maintain at the initial position.

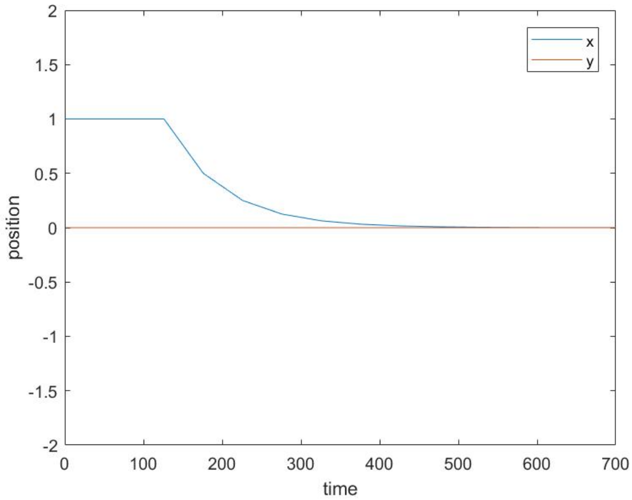

Figure 5: The change of position from (1,0) to the initial position (0,0)

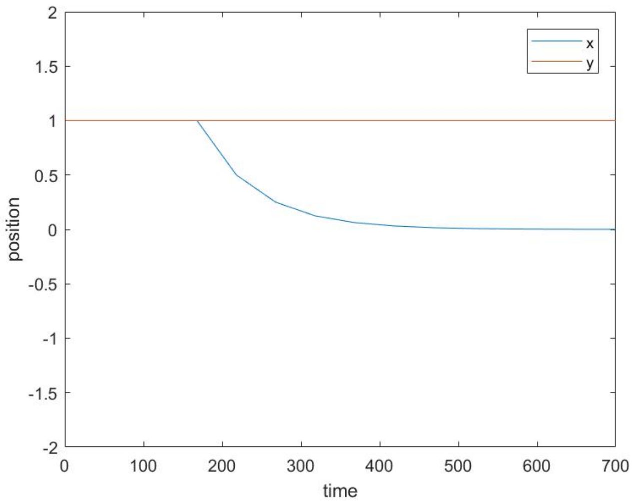

Figure 6: The change of position from (1,1) to the initial position (0,0)

Task 9

The errors $\theta^R - \theta[k]$ and $d_0[k]$ evolve the same compared with when only one controller is enabled. The simulation results are shown in 7 and 8. We can see that the errors both maintain at 0 finally. In the simulation, we make the robot go to the initial position (x_0, y_0) first and then adjust its orientation to the desired angle θ^R. So the two controllers don't affect each other and perform the same as before.

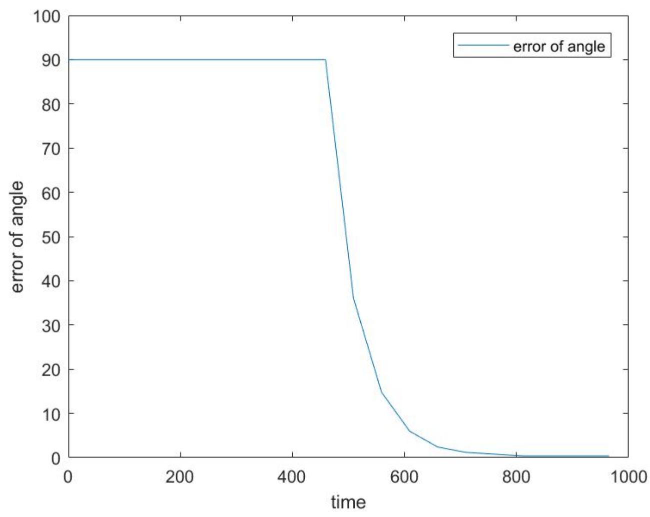

Figure 7: The change of the angle of error $\theta^R - \theta[k]$

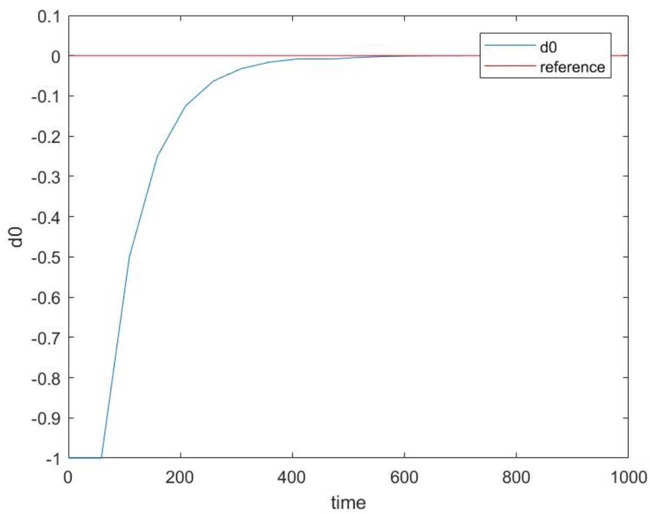

Figure 8: The change of the inner product $d_0[k]$

Task 10

From the instruction, we know that in the continuous system,

$$d_g = \begin{bmatrix} cos\theta_g & sin\theta_g \end{bmatrix} \begin{bmatrix} x_g - x \\ y_g - y \end{bmatrix} = (x_g - x)cos\theta_g + (y_g - y)sin\theta_g$$

Now we differentiate both sides. As θ_g does not change, we can get

$$\dot{d}_g = -\dot{x}cos\theta_g - \dot{y}sin\theta_g$$

From the instruction, we know $\dot{x} = Rvcos\theta_g, \dot{y} = Rvsin\theta_g$. Then

$$\dot{d}_g = -Rvcos\theta_g * cos\theta_g - Rvsin\theta_g * sin\theta_g = -Rv$$

In the discrete system,

$$d_g[k+1] = d_g[k] - hRv[k]$$

We know $v[k] = K_\omega d_g[k]$, so

$$d_g[k+1] = d_g[k] - hR * K_\omega d_g[k] = (1 - hRK_\omega)d_g[k]$$

In order to make $d_g[k]$ converge to 0, we need to fulfill the condition $|1 - hRK_\omega| < 1$. It is easy to get

$$0 < K_\omega < \frac{2}{Rh}$$

Task 11

It is possible for the robot to arrive at $[x_g, y_g]$ exactly. We set $K_\omega = 500$, the initial position $(x_0, y_0, \theta_0) = (0, 0, 45°)$, and the goal position $(x_g, y_g) = (1, 1)$. The result of the simulation is shown in figure 9. We can see that the position of x approaches to $x_g = 1$ from $x_0 = 0$, and the same goes for the position of y.

Task 12

From the instruction, we know that in the continuous system,

$$d_p = \begin{bmatrix} sin\theta_g & -cos\theta_g \end{bmatrix} \begin{bmatrix} x + pcos\theta - x_0 \\ y + psin\theta - y_0 \end{bmatrix} = (x - x_0)sin\theta_g - (y - y_0)cos\theta_g + psin(\theta_g - \theta)$$

We know that $x - x_0 = \sqrt{(x-x_0)^2 + (y-y_0)^2} * cos(\theta_g)$, $y - y_0 = \sqrt{(x-x_0)^2 + (y-y_0)^2} * sin(\theta_g)$, and we assume that θ is close to θ_g. So

$$d_p = psin(\theta_g - \theta) \approx p(\theta_g - \theta)$$

Now we differentiate both sides. We can get

$$\dot{d}_p = -p\dot{\theta} = -\frac{pR}{L}\omega$$

In the discrete system,

$$d_p[k+1] = d_p[k] - \frac{hpR}{L}\omega[k]$$

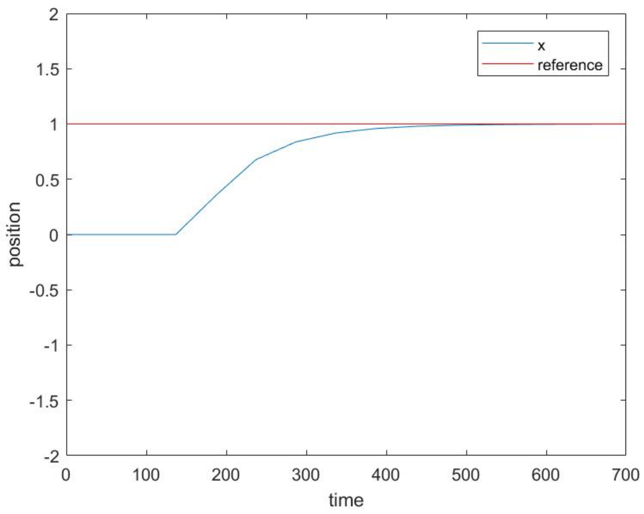

Figure 9: The change of position x (y)

We know $\omega[k] = K_\Psi d_p[k]$, so

$$d_p[k+1] = d_p[k] - \frac{hpR}{L} * K_\Psi d_p[k] = (1 - \frac{hpR}{L} K_\Psi) d_p[k]$$

In order to make $d_p[k]$ converge to 0, we need to fulfill the condition $|1 - \frac{hpR}{L} K_\Psi| < 1$. It is easy to get

$$0 < K_\Psi < \frac{2L}{hpR}$$

Task 13

When we approximate $d_p \approx p(\theta_g - \theta)$, we can see p as the gain of the proportional controller. Increasing p, the accuracy of θ to θ_g will be higher. However if p is chosen too high, θ will fluctuate around θ_g and can't stop at a point: this due to the fact that the stability region becomes smaller while increasing the value of p.

Task 14

It is possible to maintain $d_p[k]$ at 0 exactly. In the simulation, we set the initial angle $\theta_0 = 0°$ and goal angle $\theta_g = 90°$. The result of the simulation is shown in 17. We can see that d_p approaches to 0 and maintain to 0 finally.

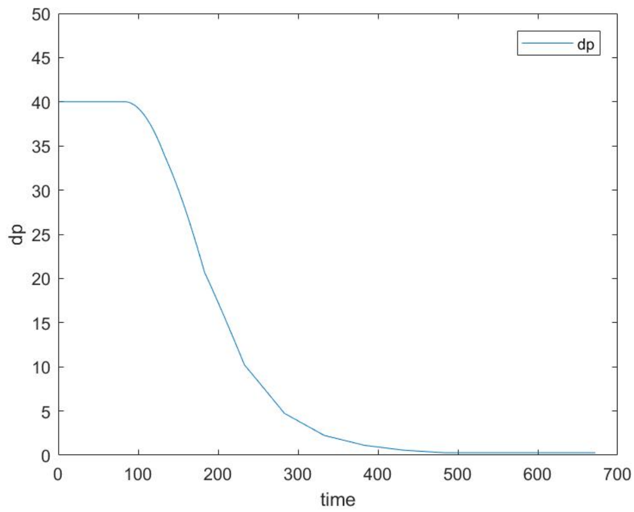

Figure 10: The change of d_p

Task 15

When both controllers are executed, the simulation results of $d_g[k]$ and $d_p[k]$ are shown in figure 11 and 12. In this simulation, we make the initial and goal angle 85° and 90°. We can see that d_g approaches to and maintain at 0 finally while d_p does not maintain at 0.

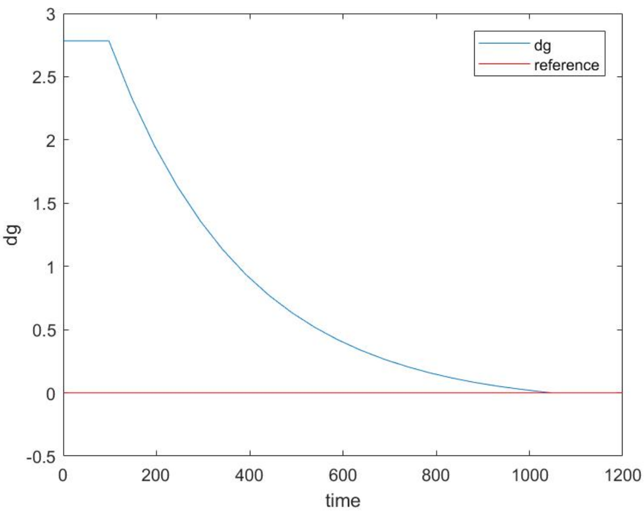

Figure 11: The change of d_g

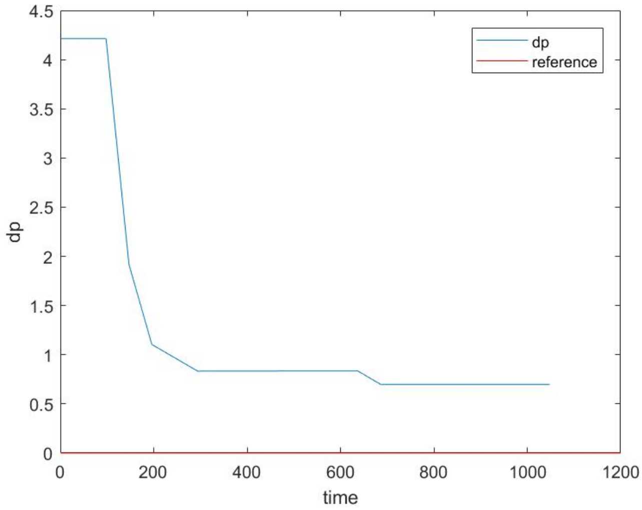

Figure 12: The change of d_p

Task 16

To reach the desired position the robot first rotates and then translates. As tail of this, the hybrid controller has been designed to switch from the discrete state of rotation (rotation controller) to the state of translation (line following controller) until the goal position $[x_g, y_g]$ is reached in the goal state. It is formally described as $H = (Q, X, Init, f, D, E, G, R)$.

- Q are the discrete states of the controller: $Q = \{q_1, q_2, q_3\} = \{rotation, translation, goal\}$. The robot is in state q_1 when executing a rotation, in state q_2 when is translating, and in state q_3 when it stops in the goal position.

- X are the continuous states of the controller which represents all the possible position the robot can have: $X = \{(x, y) \in R^2, \theta \in (-\pi, \pi]\}$

- $Init$ represents the initial states (both continuous and discrete) of the controller: $Init = \{rotation, x_0, y_0, \theta_0\}$

- f are the vector fields. Indicate with $X = [x, y, \theta]$

$$f(rotation, X) = f(q_1, X) = \{\frac{R}{2}(u_l + u_r)cos\theta, \frac{R}{2}(u_l + u_r)sin\theta, \frac{R}{L}(u_r - u_l)\}$$

$$f(traslation, X) = f(q_2, X) = \{\frac{R}{2}(u_l + u_r)cos\theta, \frac{R}{2}(u_l + u_r)sin\theta, \frac{R}{L}(u_r - u_l)\}$$

$$f(goal, X) = f(q_3, X) = \{0, 0, 0\}$$

- D is the domain of the discrete states (what conditions need to be satisfied in order for the automaton to stay in a state):

$$D(q_1) = \{[x, y] \in R^2, \theta \in (-\pi, \pi] : |[x, y] - [x_0, y_0]| \leq [1, 1]\ [cm]\ , |\theta - \theta_g| \geq 1\ [deg]\}$$

$$D(q_2) = \{[x, y] \in R^2, \theta \in (-\pi, \pi] : |[x_g, y_g] - [x, y]| \geq [1, 1]\ [cm]\ , |\theta - \theta| \leq 1\ [deg]\}$$

$$D(q_3) = \{[x, y] \in R^2, \theta \in (-\pi, \pi] : |[x_g, y_g] - [x, y]| \leq [1, 1]\ [cm]\ , |\theta - \theta_g| \leq 1\ [deg]\}$$

- are the edges, the possible switches between the states:

$$E = \{(rotation, translation), (translation, goal)\}$$

- G are the guards, the conditions that allow to switch from a state to another:

$$G(q_1, q_2) = \{[x, y] \in R^2, \theta \in (-\pi, \pi] : |\theta - \theta_g| \leq 1\ [deg]\}$$

$$G(q_2, q_3) = \{[x, y] \in R^2, \theta \in (-\pi, \pi] : |[x_g, y_g] - [x, y]| \leq [1, 1]\ [cm] \theta - \theta_g| \leq 1\ [deg]\}$$

- R are the reset maps of the controller:

$$R(q_1, q_2, X) = \{[x, y, \theta]\}$$

$$R(q_2, q_3, X) = \{[x, y, \theta]\}$$

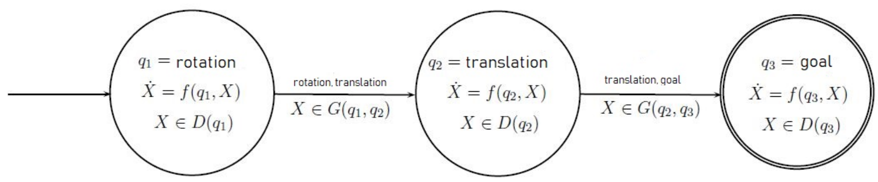

Figure 13: The hybrid automaton that controls the robot

Task 17

When the hybrid controller has been implemented, the following simulation results are achieved with the robot. In the simulation the initial position $(x_0; y_0; 0)$ is set to be (0; 0; 0) while the goal position $(x_g; y_g)$ is set to be (1; 1). Figures 14, 15, 16 show the change of coordinate x, y and angle θ respectively. When the hybrid controller is active, firstly controls the rotation and then, when the desired angle is reached, the robot starts translating through the goal position. Consequently, the hybrid controller is in the discrete state of translation and finally, when the robot arrives at the goal point, the goal state is activated. Therefore, the robot maintains its position.

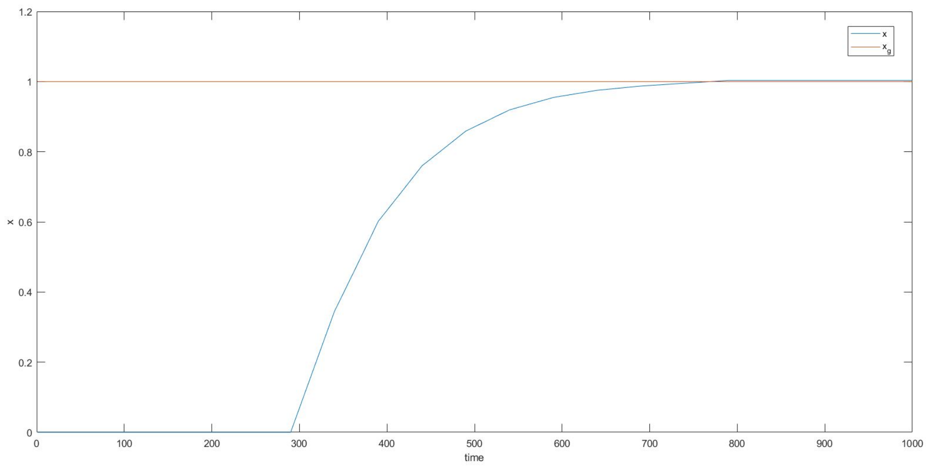

Figure 14: continuous trajectory of x

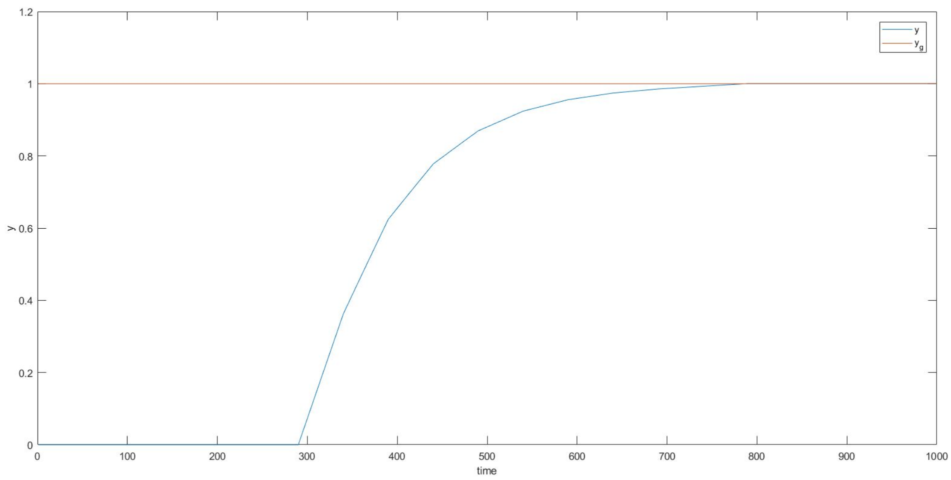

Figure 15: continuous trajectory of y

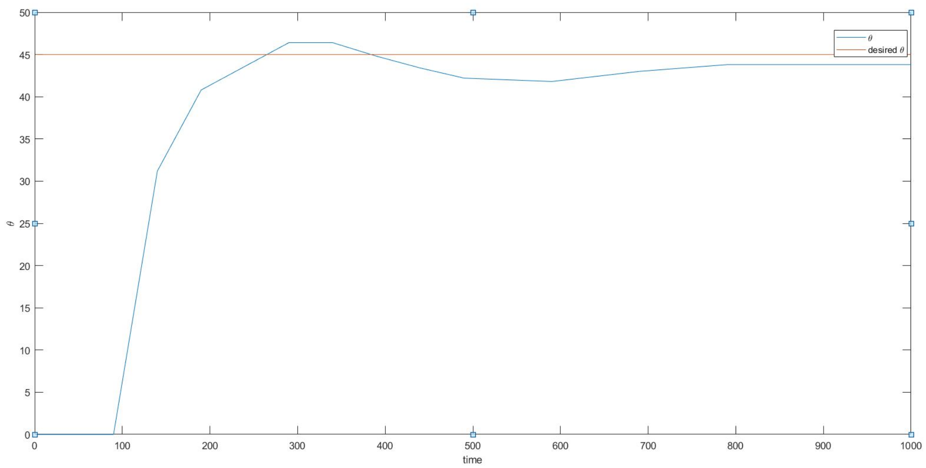

Figure 16: continuous trajectory of θ

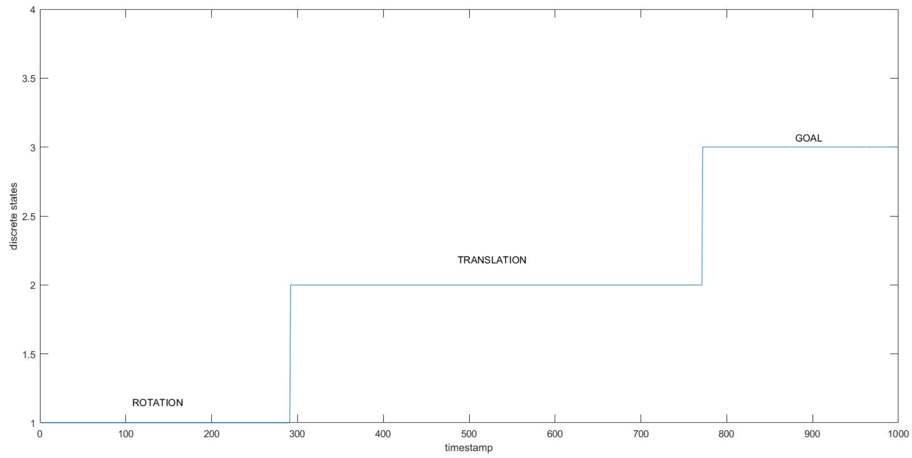

Figure 17: discrete trajectory

Task 18

Executing the plan proposed in Task 3 to the controller, the robot is able to follow a list of way-points. We set the coordinates as:

$$(-1.25; 1.25) \rightarrow (-0.25; 1.25) \rightarrow (-0.25; 0.75) \rightarrow (0.25; 0.75) \rightarrow (0.25; 0.25)$$

The trajectory of the robot to follow the target way-points can be saw in the following figure. The controller is effective since the robot is able to pass through each way-point with a very small error.

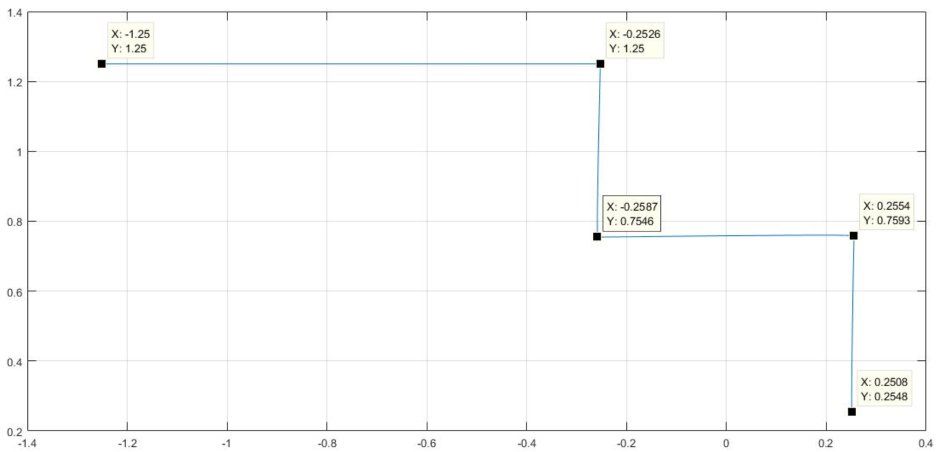

Figure 18: The trajectory of the robot following waypoints

Task 19

In order to make sure that each transitions to be safe, the continuous controllers should maintain the initial position and the final position of the robot. In other words, if the robot stays in the two regions as exact as possible, then the transition might be more stable.