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

Javier Cerna
930917
fjch@kth.se

Junhao Cui
960319
junhaoc@kth.se

Ruobing Li
950911
ruobing@kth.se

Lihao Yan
980711
lihaoy@kth.se

2020-02-24

Task 1

Know from the Homework instruction that

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

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

Therefore, one can easily get

$$u_l = v - \frac{1}{2}\omega.$$

$$u_r = v + \frac{1}{2}\omega.$$

Task 2

The discretized workspace is shown in Figure 1. The $3 \times 3m^2$ workspace is divided into 36 identical regions of $0.5 \times 0.5m^2$. The robot(represented by a gray square in Figure 1) is located at the initial position (-1.25,1.25).

The Transition System $T = \{S, S_0, \Sigma, \rightarrow, AP, L\}$ can be defined in the following way:

- $S_0 = \{R_6\}$.
- $S = \{R_1, R_2, R_3, ..., R_K\}$, where $K = 36$.
- $\Sigma = \{forward, backward, left, right\}$.

- $\rightarrow = \{$
$(R_1, forward, R_2), (R_1, right, R_{12}), (R_6, backward, R_5), (R_6, right, R_7), (R_{36}, forward, R_{35}),$
$(R_{36}, left, R_{25}), (R_{31}, backward, R_{32}), (R_{31}, left, R_{30}), (R_2, forward, R_3), (R_2, backward, R_1),$
$(R_2, right, R_6), (R_3, forward, R_4), (R_3, backward, R_2), (R_3, right, R_{10}), (R_4, forward, R_5),$
$(R_4, backward, R_3), (R_4, right, R_9), (R_5, forward, R_6), (R_5, backward, R_4), (R_5, right, R_8),$
$(R_7, left, R_6), (R_7, backward, R_8), (R_7, right, R_{18}), (R_{18}, left, R_7), (R_{18}, backward, R_{17}),$
$(R_{18}, right, R_{19}), (R_{19}, left, R_{18}), (R_{19}, backward, R_{20}), (R_{19}, right, R_{30}), (R_{30}, left, R_{19}),$
$(R_{30}, backward, R_{29}), (R_{30}, right, R_{31}), (R_{32}, left, R_{29}), (R_{32}, backward, R_{33}), (R_{32}, forward, R_{31}),$
$(R_{33}, left, R_{28}), (R_{33}, backward, R_{34}), (R_{33}, forward, R_{32}), (R_{34}, left, R_{27}), (R_{34}, backward, R_{35}),$
$(R_{34}, forward, R_{33}), (R_{35}, left, R_{26}), (R_{35}, backward, R_{36}), (R_{35}, forward, R_{34}), (R_{25}, left, R_{24}),$
$(R_{25}, right, R_{36}), (R_{25}, forward, R_{26}), (R_{24}, left, R_{13}), (R_{24}, right, R_{25}), (R_{24}, forward, R_{27}),$
$(R_{13}, left, R_{12}), (R_{13}, right, R_{24}), (R_{13}, forward, R_{14}), (R_{12}, left, R_1), (R_{12}, right, R_{13}),$
$(R_{12}, forward, R_{11}), (R_8, forward, R_7), (R_8, backward, R_9), (R_8, left, R_5), (R_8, right, R_{17}),$
$(R_9, forward, R_8), (R_9, backward, R_{10}), (R_9, left, R_4), (R_9, right, R_{16}), (R_{10}, forward, R_9),$
$(R_{10}, backward, R_{11}), (R_{10}, left, R_3), (R_{10}, right, R_{15}), (R_{11}, forward, R_{10}), (R_{11}, backward, R_{12}),$
$(R_{11}, left, R_2), (R_{11}, right, R_{14}), (R_{17}, forward, R_{18}), (R_{17}, backward, R_{16}), (R_{17}, left, R_8),$
$(R_{17}, right, R_{20}), (R_{16}, forward, R_{17}), (R_{16}, backward, R_{15}), (R_{16}, left, R_9), (R_{16}, right, R_{21}),$
$(R_{15}, forward, R_{16}), (R_{15}, backward, R_{14}), (R_{15}, left, R_{10}), (R_{15}, right, R_{22}), (R_{14}, forward, R_{15}),$
$(R_{14}, backward, R_{13}), (R_{14}, left, R_{11}), (R_{14}, right, R_{23}), (R_{20}, forward, R_{19}), (R_{20}, backward, R_{21}),$
$(R_{20}, left, R_{17}), (R_{20}, right, R_{29}), (R_{21}, forward, R_{20}), (R_{21}, backward, R_{22}), (R_{21}, left, R_{16}),$
$(R_{21}, right, R_{28}), (R_{22}, forward, R_{21}), (R_{22}, backward, R_{23}), (R_{22}, left, R_{15}), (R_{22}, right, R_{27}),$
$(R_{23}, forward, R_{22}), (R_{23}, backward, R_{24}), (R_{23}, left, R_{14}), (R_{23}, right, R_{26}), (R_{29}, forward, R_{30}),$
$(R_{29}, backward, R_{28}), (R_{29}, left, R_{20}), (R_{29}, right, R_{32}), (R_{28}, forward, R_{29}), (R_{28}, backward, R_{27}),$
$(R_{28}, left, R_{21}), (R_{28}, right, R_{33}), (R_{27}, forward, R_{28}), (R_{27}, backward, R_{26}), (R_{27}, left, R_{22}),$
$(R_{27}, right, R_{34}), (R_{26}, forward, R_{27}), (R_{26}, backward, R_{25}), (R_{26}, left, R_{23}), (R_{26}, right, R_{35}), \}.$

- $AP = \{red, blue, green, obstacle\}$

- The labeling function of each state can be assigned in the following form:
$L(R_1) = \{red\}, L(R_9) = \{green\}, L(R_{22}) = \{obstacle\}, L(R_{32}) = \{red\},$
$L(R_2) = \{green\}, L(R_{10}) = \{obstacle\}, L(R_{23}) = \{blue\}, L(R_{36}) = \{blue\},$
$L(R_3) = \{obstacle\}, L(R_{11}) = \{obstacle, red\}, L(R_{25}) = \{green\}, L(R_6) = \{green\},$
$L(R_{19}) = \{obstacle\}, L(R_{26}) = \{obstacle, red\}, L(R_7) = \{blue\}, L(R_{20}) = \{red\},$
$L(R_{29}) = \{obstacle, blue\}, L(R_8) = \{obstacle, red\}, L(R_{21}) = \{blue\},$
$L(R_{31}) = \{obstacle, red, green\}$
$L(R_i) = \varnothing, \forall i \in \{4, 5, 12, 13, 14, 15, 16, 17, 18, 24, 27, 28, 30, 33, 34, 35\}$

Moreover, the black solid circles in Figure 1 represents "obstacles". The position with property of "blue", "red" or "green" is illustrated using "blue", "red" or "green" dot, respectively.
One can see that all the obstacles are distributed into several relatively dispersed locations with respect to the colored positions, so that the majority of the colored dots would, in the worst case, fall into the neighboring regions of obstacles instead of the "obstacle" regions, which is good for the robot to navigate. On the other hand, if we expand the area of each region to, for instance, $0.75 \times 0.75m^2$, we will lose such kind of benefit.

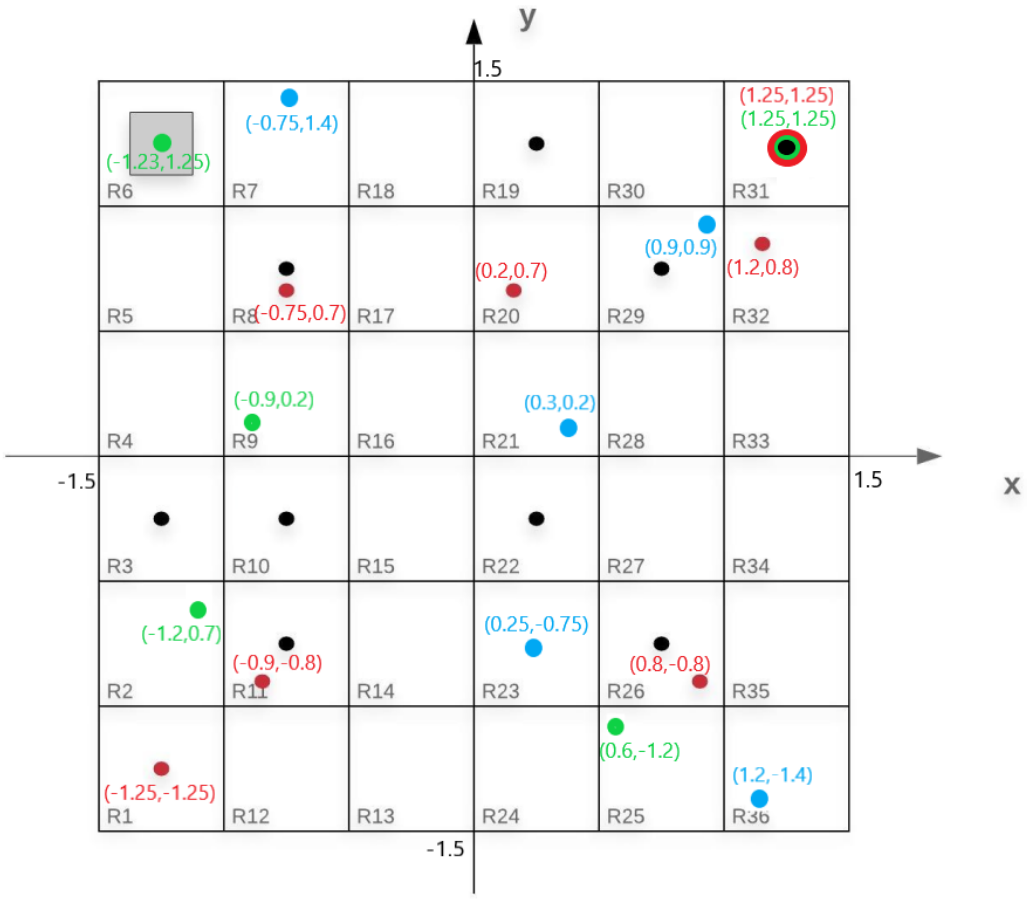


Figure 1: Graphical illustration of the discretized workspace

Task 3

The three conditions can be expressed using LTL and the atomic propositions as $\square\Diamond$ "red", $\square$ ("red" $\rightarrow \bigcirc$ "blue"), $\square$ (! "obstacles"). One possible path can be:

$$R_6 R_5 R_4 R_9 R_{16} R_{17} (R_{20} R_{21})^{\omega},$$

where

- prefix: R_6, R_5, R_4, R_9, R_{16}, R_{17}, R_{20}, R_{21};
- suffix: R_{20}, R_{21}.

This path is also shown in Figure 2.

Task 4

This choice is safe because it guarantees that the robot will not cross the "obstacle" regions during the navigation.

Turning on the orientation and distance control simultaneously might not be a good choice, since it could yield a curved trajectory, so that the robot cannot always remain in the two regions, which is contradictory to our safety requirement.

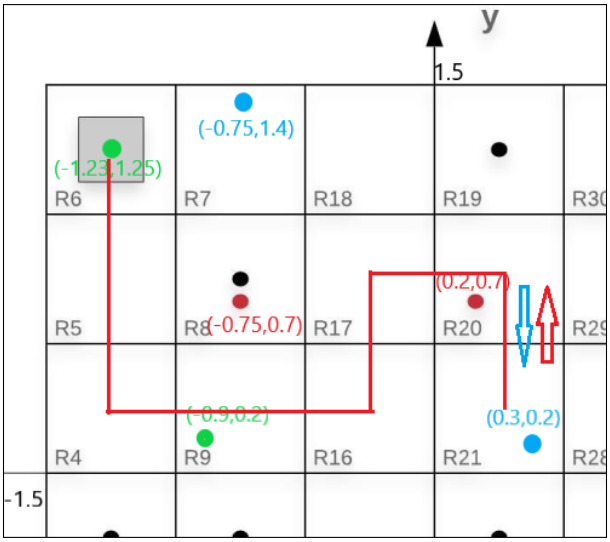


Figure 2: Possible path that satisfies the required behavior

Task 5

If only the dynamics of θ are considered, it can be treated as a linear system. The differential equation can be discretized, leading to the following result:

$$\theta(kh+h) = \theta(kh) + \frac{Rh}{L} w(kh) \tag{3}$$

Where $w(kh) = K_\psi(\theta^R - \theta(kh))$. If the state e_θ is defined as

$$e_\theta(kh) = \theta^R - \theta(kh) \tag{4}$$

Then $w(kh)$ can be expressed as $w(kh) = K_\psi e_\theta$. To determine if e_θ can approach 0 asymptotically, a time shift of one unit can be applied to determine the dynamics of this error. This leads to the following:

$$e_\theta(kh+h) = \theta_R - \theta(kh+h) \tag{5}$$

Replacing equation 3 and the definition of $w(kh)$ into equation 5 leads to the following result:

$$e_\theta(kh+h) = (1 - \frac{Rh}{L} K_\psi) e_\theta(kh) \tag{6}$$

To guarantee that the error can approach 0 asymptotically, the eigenvalue needs to lie inside the unit circle. This is expressed as

$$|1 - \frac{Rh}{L} K_\psi| < 1, \tag{7}$$

which results in the boundaries for K_ψ

$$0 < K_\psi < \frac{2L}{Rh} \tag{8}$$

Task 6

The simulation result is illustrated in Figure 3. The position holds the initial value of $\theta = 0°$ and our desired value $\theta^R = -135°$. One can see from the curve that the value of θ finally becomes stable and the error is $0.6°$. In theory, the final value of θ should be exactly the same as θ^R, because the controller is designed such that θ is asymptotically stable, which means the error in the end should be zero. Also, there is no non-holonomic constraint imposed in θ, which means that this variable can take any value (from $0°$ to $360°$).

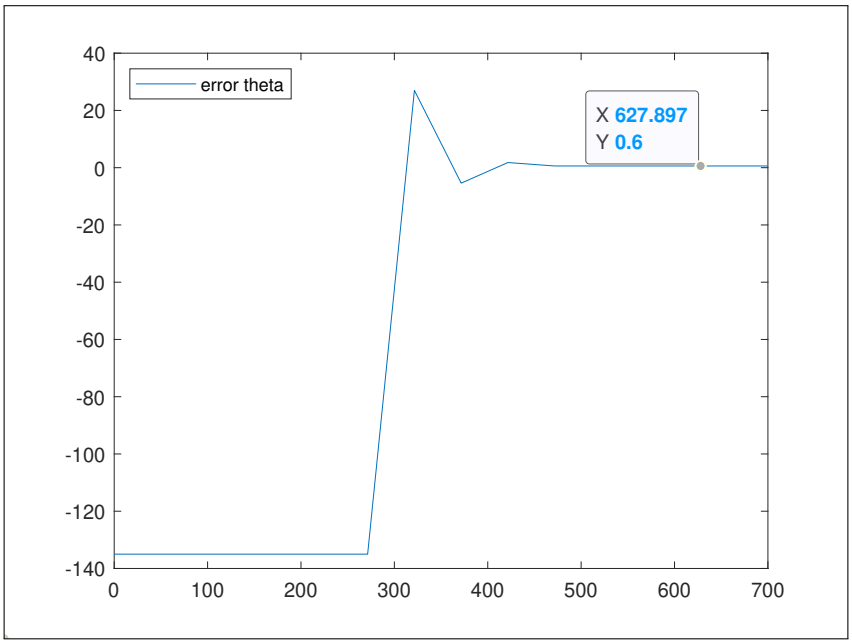


Figure 3: The error of θ with respect to θ^R, where $\theta^R = -135°$

Task 7

Assuming that θ doesn't change allows to treat the dynamics of x and y as a linear system. The differential equation can be discretized, leading to the following result:

$$x(kh+h) = x(kh) + Rh\cos(\theta)v(kh) \tag{9}$$
$$y(kh+h) = y(kh) + Rh\sin(\theta)v(kh) \tag{10}$$

Where $v(kh) = K_w d_0(kh)$. To determine if d_0 can approach 0 asymptotically, a time shift of one unit can be applied to determine the dynamics of this error. This leads to the following:

$$d_0(kh+h) = (x_0 - x(kh+h))\cos(\theta) + (y_0 - y(kh+h))\sin(\theta) \tag{11}$$

Replacing equations 9 and 10, and the definition of $v(kh)$ into equation 11 leads to the following result:

$$d_0(kh+h) = (1 - hRK_w)d_0(kh) \tag{12}$$

To guarantee that the error can approach 0 asymptotically, the eigenvalue needs to lie inside the unit circle. This is expressed as

$$|1 - hRK_w| < 1 \tag{13}$$

, which results in the boundaries for K_w

$$0 < K_w < \frac{2}{Rh} \tag{14}$$

Task 8

The previous task shows that it's possible to make the error approach 0 asymptotically, so x and y can be maintained almost exactly at x0, y0 if w is set to 0. The case when θ is equal to θ^R is shown in Figure 4 (the error for both x and y is less than 0.01 m).

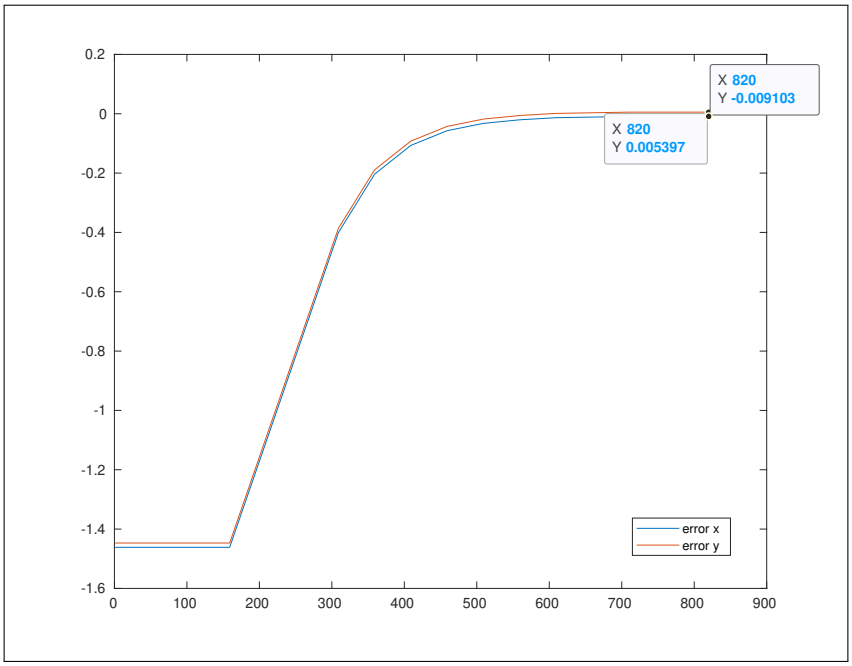


Figure 4: Error of x and y when θ equal (or close by an epsilon) than θ_R

However, when θ is different than the desired θ^R, x and y can get close to x0, y0, but they will never be at exactly that position, because the controller can't change the value of θ. This is shown in Figure 5 (the error for both x and y is less than 0.07).

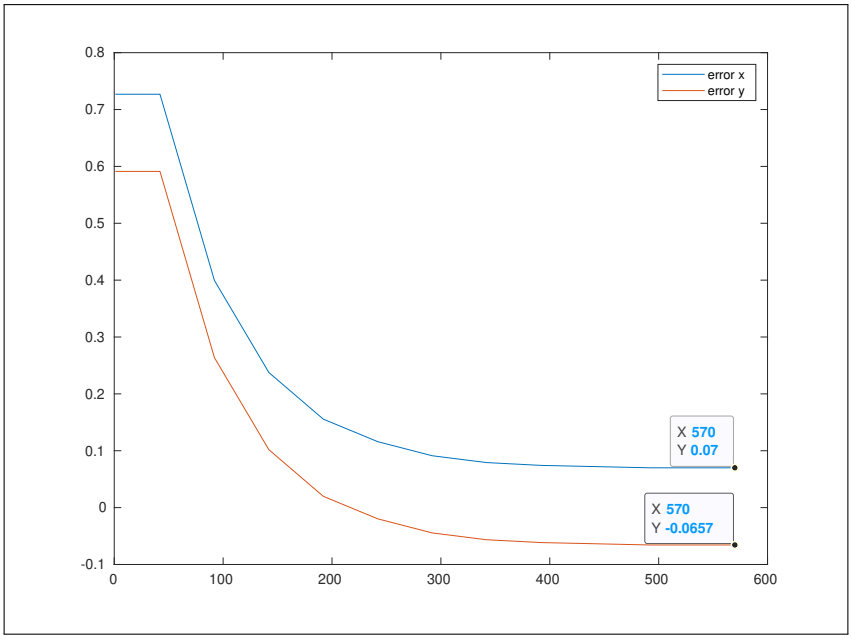


Figure 5: Error of x and y when θ different than θ_R

The reason for this two different behaviors is the non-holonomic constraint imposed in x and y (they depend on θ to achieve certain poses, because the vehicle can't move laterally).

Task 9

When both controllers are enabled, now the limitation from Task 8 is not present anymore. As it can be seen in Figure 6, both the error in θ and d_0 achieve very small values, even when the initial angle is different (the plot shows that the initial angle error was set to more than 100° to test the performance). The final errors are -0.08° and -0.00051 m, respectively.

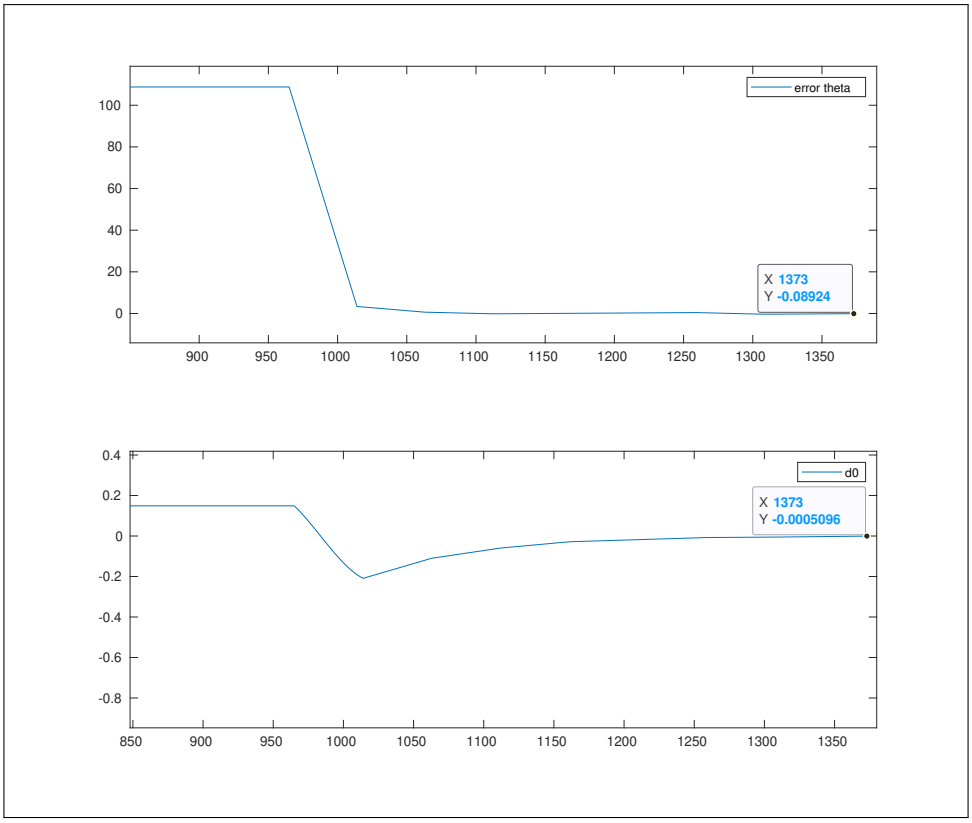


Figure 6: Error of θ and d_0

Task 10

Assuming that $\theta = \theta_g$ (constant) allows to treat the dynamics of x and y as a linear system. The differential equation can be discretized, leading to the following result:

$$x(kh+h) = x(kh) + Rh\cos(\theta_g)v(kh) \tag{15}$$
$$y(kh+h) = y(kh) + Rh\sin(\theta_g)v(kh) \tag{16}$$

Where $v(kh) = K_w d_g(kh)$. To determine if d_g can approach 0 asymptotically, a time shift of one unit can be applied to determine the dynamics of this error. This leads to the following:

$$d_g(kh+h) = (x_g - x(kh+h))\cos(\theta_g) + (y_g - y(kh+h))\sin(\theta_g) \tag{17}$$

Replacing equations 15 and 16, and the definition of $v(kh)$ into equation 17 leads to the following result:

$$d_g(kh+h) = (1 - RhK_w)d_g(k) \tag{18}$$

To guarantee that the error can approach 0 asymptotically, the eigenvalue needs to lie inside the unit circle. This is expressed as

$$|1 - RhK_w| < 1 \tag{19}$$

, which results in the boundaries for K_w

$$0 < K_w < \frac{2}{Rh} \tag{20}$$

Task 11

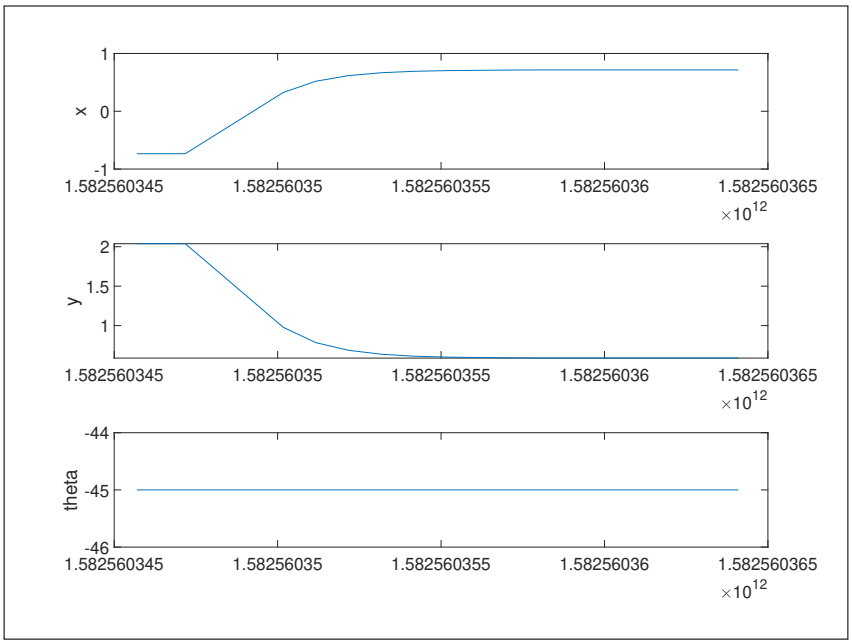


Figure 7: Value of x, y and θ (reference from waypoint 1 to waypoint 5 in the simulator)

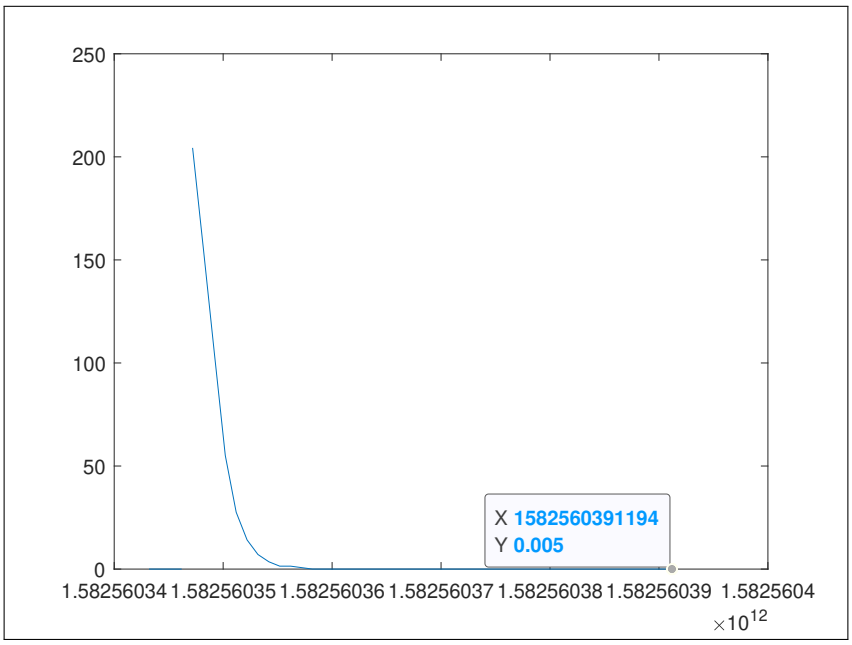


Figure 8: Value of d_g (reference from waypoint 1 to waypoint 5 in the simulator)

It is not possible to arrive at $[x_g, y_g]$ exactly, but it is possible to arrive very close to $[x_g, y_g]$ because of the numerical approximation. We can notice from figure 7 that x and y do not change much. According to figure 8, d_g is very large in the beginning, which means that v is quite large and the robot moves very fast. As the robot moves close to the goal, d_g gradually decreases to 0.005. Thus, the robot still has a very small velocity v eventually, which means that the robot arrives close to $[x_g, y_g]$ at last.

Task 12

Similar to Task 5, equation 3 can be used. However, in this case, $w(kh)$ is defined as $w(kh) = K_\psi d_p(kh)$. Also, $d_p(hk)$ is approximated as follows:

$$d_p(kh) = p(\theta_g - \theta(kh)) \tag{21}$$

To determine if d_p can approach 0 asymptotically, a time shift of one unit can be applied to determine the dynamics of this error. This leads to the following:

$$d_p(kh + h) = p(\theta_g - \theta(kh + h)) \tag{22}$$

Replacing equation 3 and the definition of $w(kh)$ into equation 22 leads to the following result:

$$d_p(kh + h) = (1 - \frac{pRh}{L} K_\psi) d_p(kh) \tag{23}$$

To guarantee that the error can approach 0 asymptotically, the eigenvalue needs to lie inside the unit circle. This is expressed as

$$|1 - \frac{pRh}{L} K_\psi| < 1 \tag{24}$$

, which results in the boundaries for K_ψ

$$0 < K_\psi < \frac{2L}{pRh} \tag{25}$$

Task 13

The value of p influences the maximum value of K_ψ that can be used, as proved by Task 12. It acts as some sort of proportial control, in the sense that larger values of p will allow smaller values of K_ψ and viceversa.

Task 14

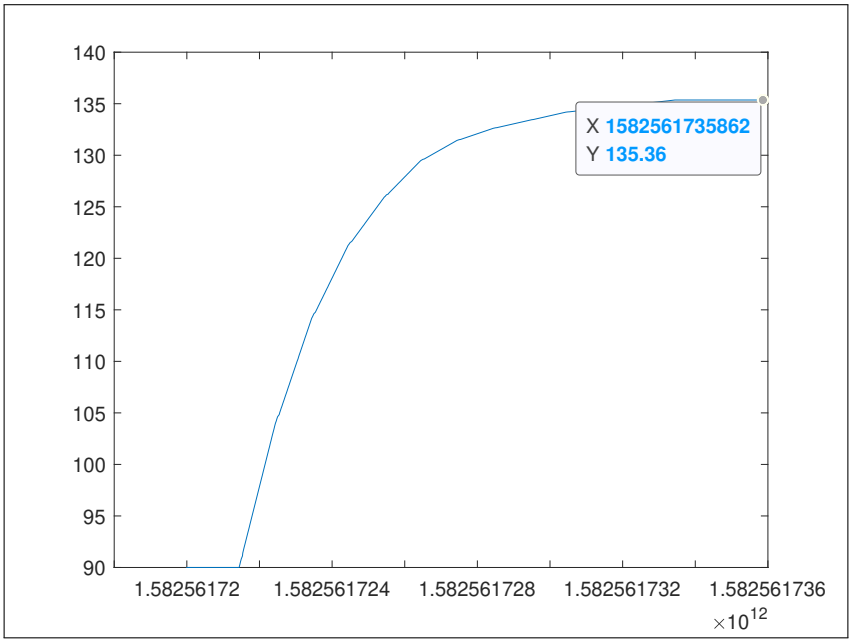


Figure 9: Value of θ (reference from waypoint 6 to waypoint 2 in the simulator)

It is possible to maintain $d_p[k]$ close to 0, but not exactly at 0. In this case shown in Figure 9, $\theta_g = 135°$, we set θ to 90 degrees in the beginning. We can find $\theta = 135.36°$ and the robot almost points to the goal eventually. From Figure 10, we can see that d_p gradually decreases to 0.007 as the robot rotates close to the reference line, which means that the robot still has a very small velocity w finally. Thus, we can't maintain $d_p[k]$ at 0 exactly, but we can maintain it very close to 0.

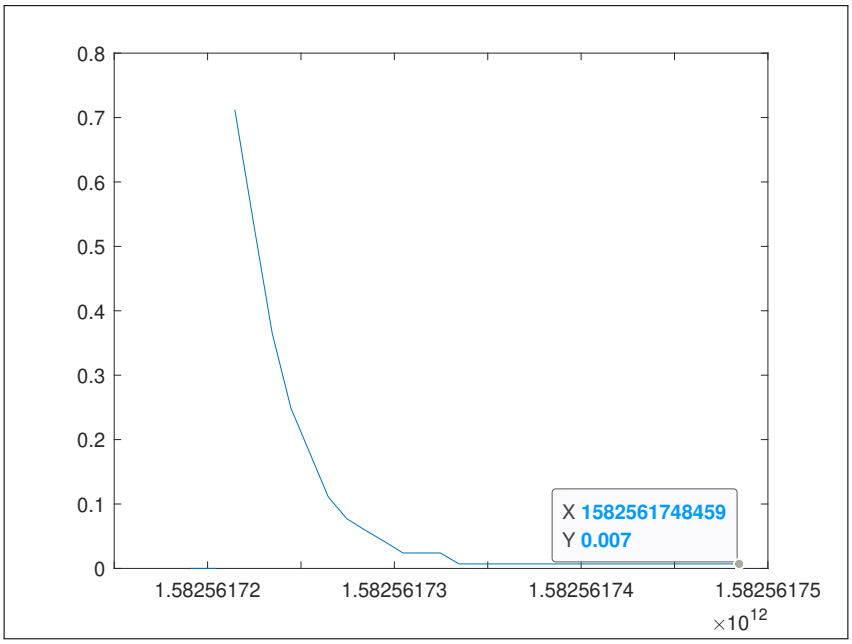


Figure 10: Value of d_p (reference from waypoint 6 to waypoint 2 in the simulator)

Task 15

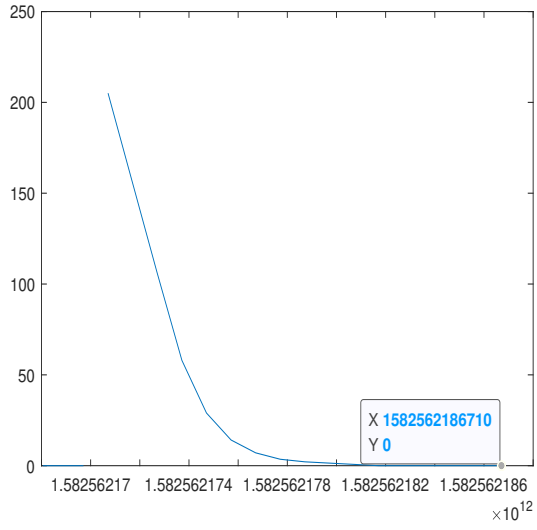


Figure 11: Value of d_g

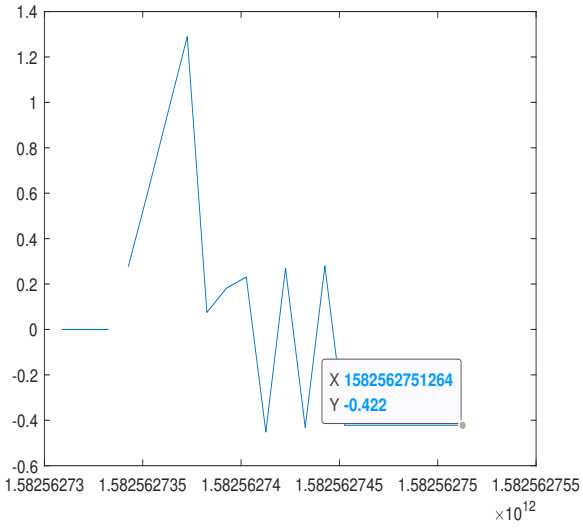


Figure 12: Value of d_p

Comparing Figure 8 and Figure 11, we can find that d_g in the case when both controllers are enabled is the same as d_g in the case when only one controller is enabled. However, comparing Figures 10 and 12, d_p is different in those two cases.

Task 16

The hybrid automaton is

$$H = (Q, X, Init, f, D, E, G, R)$$

- Q=$\{q_1, q_2, q_3\}$, where q_1 is the "stop" state, q_2 is the "rotate" state and q_3 is the "go-to-goal" state.
- X=$\mathbb{R}^3$;
- Init=$\{q_1\} \times \{x_0, y_0, \theta_0\}$;
- $f(q_1, x, y, \theta) = \begin{bmatrix} \dot{x} = 0 \\ \dot{y} = 0 \\ \dot{\theta} = 0 \end{bmatrix}$

$$f(q_2, x, y, \theta) = \begin{bmatrix} \dot{x} = Rvcos(\theta) \\ \dot{y} = Rvsin(\theta) \\ \dot{\theta} = \frac{R}{L}w \end{bmatrix}$$

where $w = K_\Psi(\theta^R - \theta)$, $\theta^R = arctan(\frac{y_g - y_0}{x_g - x_0})$
$v = K_\omega d_0 = K_\omega[cos(\theta)(x_0 - x) + sin(\theta)(y_0 - y)]$

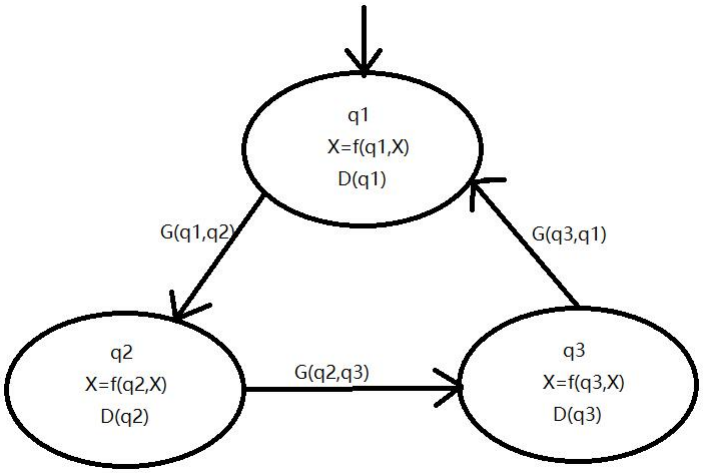
$f(q_3, x, y, \theta) = \begin{bmatrix} \dot{x} = Rvcos(\theta) \\ \dot{y} = Rvsin(\theta) \\ \dot{\theta} = \frac{R}{L}w \end{bmatrix}$
where $w = K_\Psi d_p = K_\Psi[sin(\theta_g)(x + pcos(\theta) - x_0) - cos(\theta_g)(y + psin(\theta) - y_0)]$,
$v = K_\omega d_g = K_\omega[cos(\theta_g)(x_g - x) + sin(\theta_g)(y_g - y)]$

- $D(q_1) = \{x, y, \theta \in \mathbb{R}^3 \mid |d_g| \leq \epsilon_1\}$
  $D(q_2) = \{x, y, \theta \in \mathbb{R}^3 \mid |\theta - \theta_g| > \epsilon_2\}$
  $D(q_3) = \{x, y, \theta \in \mathbb{R}^3 \mid |d_g| > \epsilon_1\}$
  where ϵ_1 is a threshold value used for the minimum acceptable error in the goal to goal controller and ϵ_2 is a threshold value used for the minimum acceptable error in the rotation controller.

- E=$\{(q_1, q_2), (q_2, q_3), (q_3, q_1)\}$

- $G(q_1, q_2) = \{x, y, \theta \in \mathbb{R}^3 \mid |d_g| > \epsilon_1\}$
  $G(q_2, q_3) = \{x, y, \theta \in \mathbb{R}^3 \mid |\theta - \theta_g| \leq \epsilon_2\}$
  $G(q_3, q_1) = \{x, y, \theta \in \mathbb{R}^3 \mid |d_g| \leq \epsilon_1\}$

- R=$\{x, y, \theta\}$
  The diagram is shown below.



Task 17

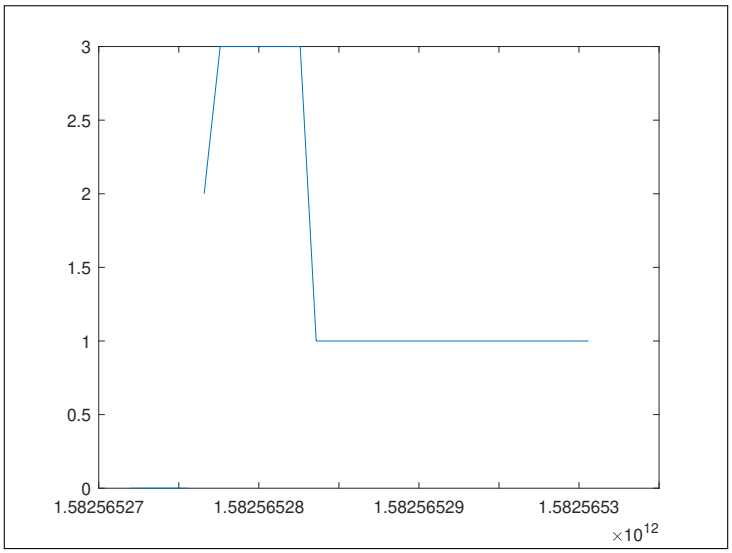


Figure 13: Discrete state trajectories

Our expected discrete state trajectory is: The robot starts at state q_1, the "stop" state, and changes to state q_2, the "rotate" state immediately. After changing the angle θ to θ_g, it moves to the state q_3, the "go-to-goal" state and begins to move forward to the goal. When the robot is arriving at the goal, it changes to state q_1, the "stop" state, and remains in that state. Thus, the discrete state evolves as we expected.

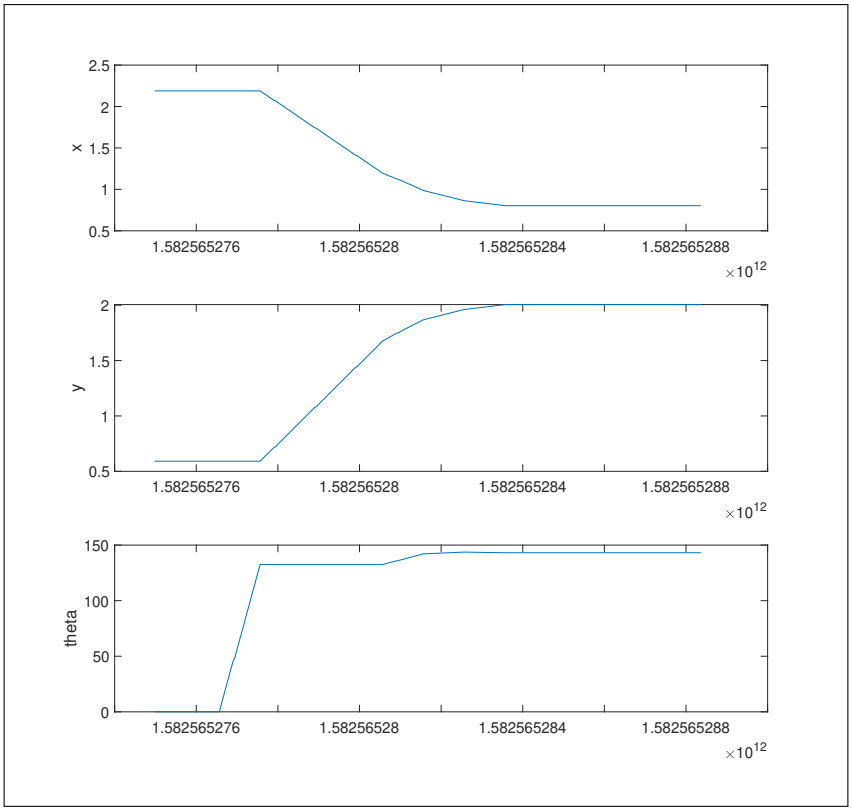


Figure 14: Continuous state trajectories

Task 18

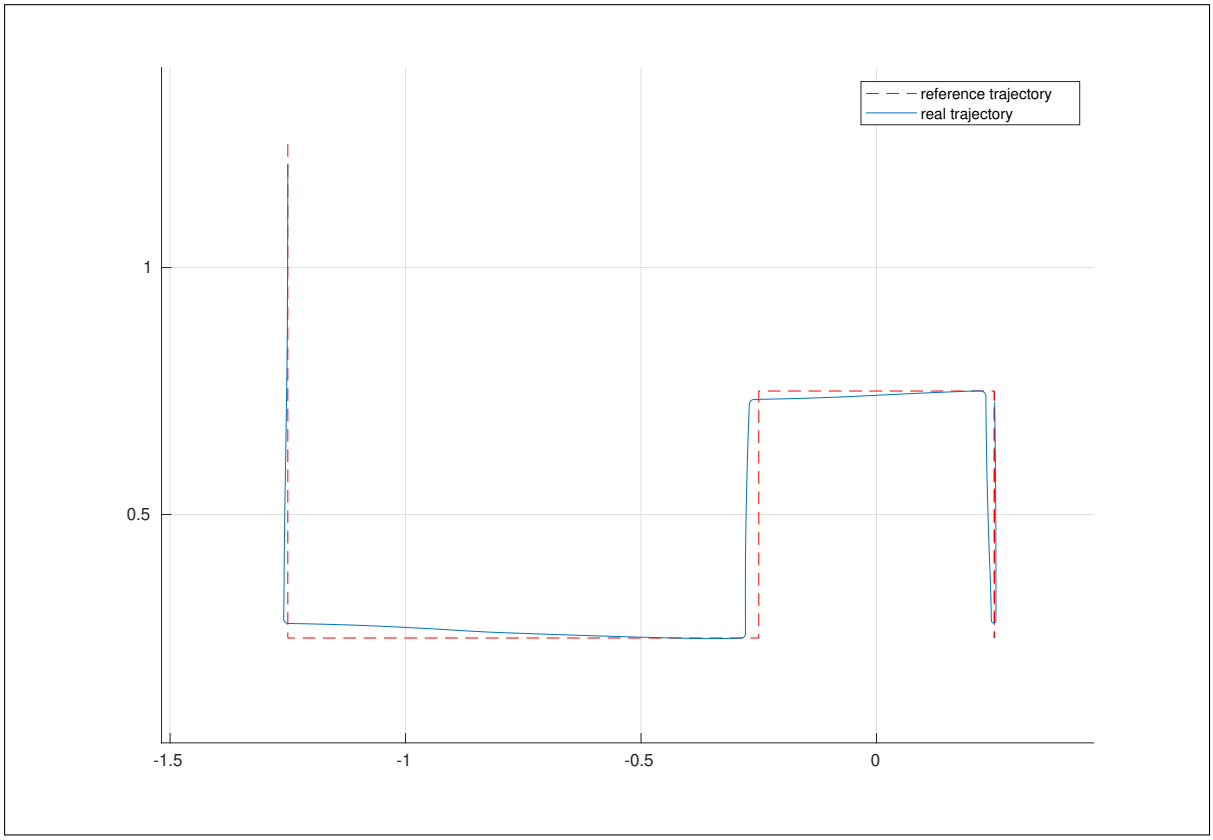


Figure 15: Reference and actual trajectory of the robot

The robot regularly follows and repeats the order: q_2 "rotate" $\rightarrow$ q_3 "move forward" $\rightarrow$ q_1 "stop". As shown in figure 15, the robot generally follows the reference trajectory without going into any obstacle regions. It doesn't follow the reference trajectory with absolute precision (particularly in the corners), but it's good enough for our purposes.

Task 19

Rotation controller ensures that the robot walks in the correct direction, and the line following controller ensures that the robot follows the correct path.
During the transition, firstly rotation controller makes the robot rotate to the correct direction, and then line following controller makes the robot walk from one region to another region. The robot cannot enter another region.