Total points 42/50

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

Fereidoon Zangeneh Kamali
960614-3338
fzk@kth.se

Mohammad Saquib Alam
940226-3017
alam7@kth.se

Akash Singh
950317-6613
akashsin@kth.se

Anirvan Dutta
950419-7713
anirvan@kth.se

February 26, 2019

Lab Session 6/6

Task 1 2/2

$$u_r + u_l = 2v, \quad u_r - u_l = \omega \quad \Longrightarrow \quad u_r = \frac{2v+\omega}{2}, \quad u_l = \frac{2v-\omega}{2}$$

Task 2 7/8

K must be greater than 0.38 and it should not be large enough to contain two red regions in 1 square. We discretize the workspace into 36 (i.e. 6 by 6) squares of size 0.5m×0.5m Figure 2 shows the proposed discretization. Solid black circles represent the obstacles, red, green and blue stars correspond to the atomic propositions of the specific regions and the red hollow circle marks the initial position of the robot.

The transition system is defined as follows-

$$\mathcal{T} = (S, S_0, \Sigma, \longrightarrow, AP, \mathcal{L})$$

$$S = \{R_1, R_2, R_3, ..., R_{35}, R_{36}\}$$

$$S_0 = \{R_6\}$$

$$\Sigma = \{\text{“}up\text{”}, \text{“}down\text{”}, \text{“}left\text{”}, \text{“}right\text{”}, \text{“}stay\text{”}\}$$

$$\longrightarrow = \{ \quad (R_{12j+i}, \text{“}up\text{”}, R_{12j+(i+1)}) \text{ for } i = \{1,2,3,4,5\} \text{ and } j = \{0,1,2\} \quad ,$$

$$(R_{12j-i}, \text{“}up\text{”}, R_{12j-(i+1)}) \text{ for } i = \{0,1,2,3,4\} \text{ and } j = \{1,2,3\} \quad ,$$

$$(R_{12j+i}, \text{“}down\text{”}, R_{12j+(i-1)}) \text{ for } i = \{2,3,4,5,6\} \text{ and } j = \{0,1,2\} \quad ,$$

$$(R_{12j-i}, \text{“}down\text{”}, R_{12j-(i-1)}) \text{ for } i = \{1,2,3,4,5\} \text{ and } j = \{1,2,3\} \quad ,$$

$$(R_{12j+i}, \text{``}right\text{''}, R_{12(j+1)-(i-1)}) \text{ for } i = \{1,2,3,4,5,6\} \text{ and } j = \{0,1,2\} \quad ,$$
$$(R_{12j-(i-1)}, \text{``}right\text{''}, R_{12j+i}) \text{ for } i = \{1,2,3,4,5,6\} \text{ and } j = \{1,2\} \quad ,$$
$$(R_{12(j+1)-(i-1)}, \text{``}left\text{''}, R_{12j+i}) \text{ for } i = \{1,2,3,4,5,6\} \text{ and } j = \{0,1,2\} \quad ,$$
$$(R_{12j+i}, \text{``}left\text{''}, R_{12j-(i-1)}) \text{ for } i = \{1,2,3,4,5,6\} \text{ and } j = \{1,2\} \quad ,$$
$$(R_k, \text{``}stay\text{''}, R_k) \text{ for } k = \{1,2,3,...,36\} \quad \}$$

$$AP = \{\text{``}red\text{''}, \text{``}green\text{''}, \text{``}blue\text{''}, \text{``}obstacle\text{''}\}$$ The empty element is missing

$$\mathcal{L}(R_1) = \{\text{``}red\text{''}\}, \quad \mathcal{L}(R_2) = \{\text{``}green\text{''}\}, \quad \mathcal{L}(R_3) = \{\text{``}obstacle\text{''}\},$$
$$\mathcal{L}(R_6) = \{\text{``}red\text{''}, \text{``}green\text{''}\}, \quad \mathcal{L}(R_7) = \{\text{``}blue\text{''}\}, \quad \mathcal{L}(R_8) = \{\text{``}red\text{''}, \text{``}obstacle\text{''}\},$$
$$\mathcal{L}(R_9) = \{\text{``}green\text{''}\}, \quad \mathcal{L}(R_{10}) = \{\text{``}obstacle\text{''}\}, \quad \mathcal{L}(R_{11}) = \{\text{``}red\text{''}, \text{``}obstacle\text{''}\},$$
$$\mathcal{L}(R_{19}) = \{\text{``}obstacle\text{''}\}, \quad \mathcal{L}(R_{20}) = \{\text{``}red\text{''}\}, \quad \mathcal{L}(R_{21}) = \{\text{``}blue\text{''}\},$$
$$\mathcal{L}(R_{22}) = \{\text{``}obstacle\text{''}\}, \quad \mathcal{L}(R_{23}) = \{\text{``}blue\text{''}\}, \quad \mathcal{L}(R_{25}) = \{\text{``}green\text{''}\},$$
$$\mathcal{L}(R_{26}) = \{\text{``}red\text{''}, \text{``}obstacle\text{''}\}, \quad \mathcal{L}(R_{29}) = \{\text{``}blue\text{''}, \text{``}obstacle\text{''}\},$$
$$\mathcal{L}(R_{31}) = \{\text{``}green\text{''}, \text{``}obstacle\text{''}\}, \quad \mathcal{L}(R_{32}) = \{\text{``}red\text{''}\}, \quad \mathcal{L}(R_{36}) = \{\text{``}blue\text{''}\},$$
$$\mathcal{L}(R_i) = \{\} \quad \text{for all other i}$$ This should be the empty element

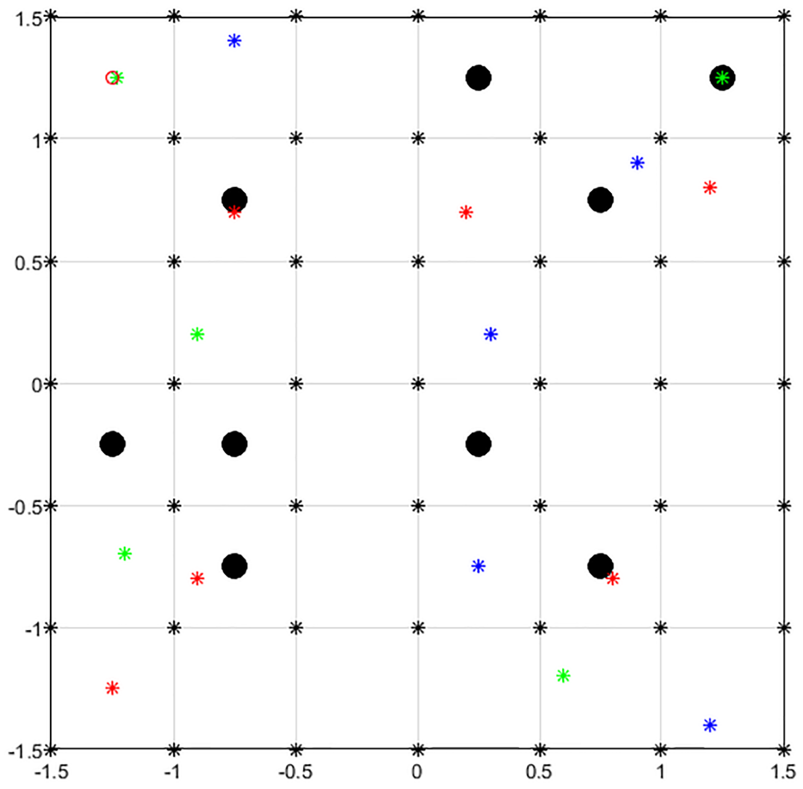


Figure 1: Discretized workspace

Task 3 4/4

$$(R_6, \text{“}right\text{”}, R_7)(R_7, \text{“}right\text{”}, R_{18})(R_{18}, \text{“}down\text{”}, R_{17})(R_{17}, \text{“}right\text{”}, R_{20})$$
$$(R_{20}, \text{“}down\text{”}, R_{21})[(R_{21}, \text{“}up\text{”}, R_{20})(R_{20}, \text{“}down\text{”}, R_{21})]^*$$

Task 4 1/1

“Safety” essentially means that there is no possible transition that can drive the system (i.e. the robot) to a ”bad” state (i.e. regions with obstacles). This manner of having a separate controller for aligning the orientation of the robot towards the goal, then driving it towards the goal helps with safety, as it eliminates the chance of a transition to any undesired regions/states (that might have obstacles in them), which may occur of the robot starts its motion at a wrong orientation. If both orientation and distance to the goal are controlled together, this may indeed happen. For example, in the case shown in Figure 5 of the homework instructions, if orientation is not corrected, the robot moves to the top-right region, followed by bottom-right region, then reaches the goal region, and as the top-right and bottom-right regions might have obstacles in them, ”safety” criterion is not met. However, if orientation of the robot is first aliened towards the goal, the transition from the initial region to the goal region is direct.

Task 5 2/2

$$\dot{\theta} = \frac{R}{L}\omega \tag{1}$$

We discretize this system with $A = 0$ and $B = \frac{R}{L}$, and sampling period h:

$$\Phi = 1, \quad \Gamma = \frac{Rh}{L} \quad \Longrightarrow \quad \theta[k+1] = \theta[k] + \frac{Rh}{L}\omega[k] \tag{2}$$

We now plug in the control law $\omega[k] = K_\Psi(\theta^R - \theta[k])$:

$$\theta[k+1] = \theta[k] + \frac{RhK_\Psi}{L}(\theta^R - \theta[k]) = (1 - \frac{RhK_\Psi}{L})\theta[k] + \frac{RhK_\Psi}{L}\theta^R \tag{3}$$

For the system to be stable, its pole must lie within the unit circle:

$$|1 - \frac{RhK_\Psi}{L}| < 1 \quad \Longrightarrow \quad -1 < 1 - \frac{RhK_\Psi}{L} < 1$$

$$0 < \frac{RhK_\Psi}{L} < 2 \quad \Longrightarrow \quad 0 < K_\Psi < \frac{2L}{Rh} \tag{4}$$

The value of K_Ψ is tuned experimentally such that it’s large enough so that convergence is not too slow, and at the same time not too large that would cause large oscillations about the steady-state value. Another fact to take into account while tuning K_Ψ is that it should not be too large to avoid requesting for physically impossible actions by the wheel motors.

Task 6 1/2

From Fig 2, it can be seen that theta converges asymptotically. However, it fails to reach the exact value of θ^R in finite time due to discretization and the fact that a perfect value of gain is required for the controller to completely cancel the error, which we do no have here.

The perfect gain is not required since you derived a range of gains which stabilize the system. Also the discretization does not make it impossible

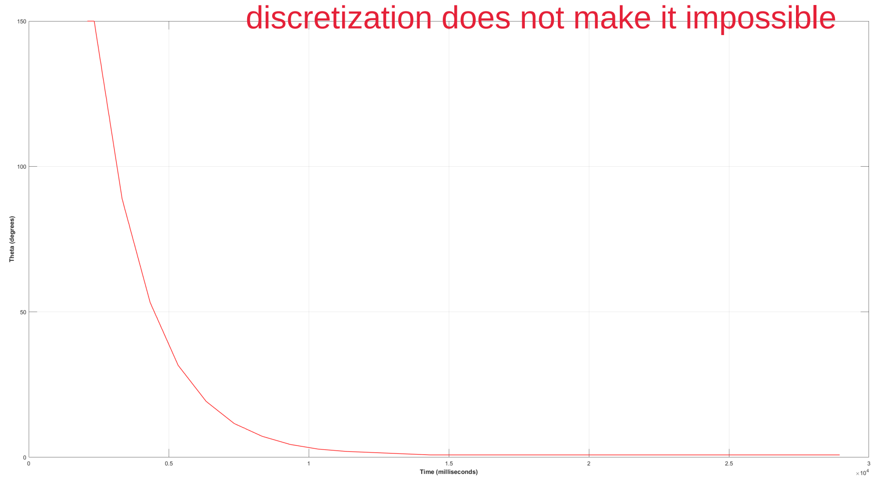


Figure 2: **Performance of rotation controller at $K_\Psi = 0.2 * \frac{2L}{Rh}$**

Task 7 2/2

$$\dot{x} = Rvcos\theta, \quad \dot{y} = Rvsin\theta \tag{5}$$

We discretize the system with $A_x = A_y = 0$, $B_x = Rcos\theta$ and $B_y = Rsin\theta$ (assuming constant θ) and with sampling period h:

$$x[k+1] = x[k] + hRcos\theta[k]v[k], \quad y[k+1] = y[k] + hRsin\theta[k]v[k] \tag{6}$$

Now we plug in the control law $v[k] = K_\omega d_0[k]$:

$$x[k+1] = x[k] + hRK_\omega cos\theta[k]d_0[k], \quad y[k+1] = y[k] + hRK_\omega sin\theta[k]d_0[k] \tag{7}$$

Now we write the expression for $d_0[k]$ and $d_0[k+1]$:

$$d_0[k] = cos\theta[k]x_0 - cos\theta[k]x[k] + sin\theta[k]y_0 - sin\theta[k]y[k] \tag{8}$$

$$d_0[k+1] = cos\theta[k+1]x_0 - cos\theta[k+1]x[k+1] + sin\theta[k+1]y_0 - sin\theta[k+1]y[k+1] \tag{9}$$

We now replace all $\theta[k+1]$ with $\theta[k]$ (as $\theta[k] = \theta[k+1]$), and also replace $x[k+1]$ and $y[k+1]$ with their forms found in (7):

$$d_0[k+1] = cos\theta[k]x_0 - cos\theta[k]x[k] - cos^2\theta[k]hRK_\omega d_0[k]$$

$$+sin\theta[k]y_0 - sin\theta[k]y[k] - sin^2\theta[k]hRK_\omega d_0[k] \tag{10}$$

We can see the expression for $d_0[k]$ which could be factorized:

$$d_0[k+1] = d_0[k] - (cos^2\theta[k] + sin^2\theta[k])hRK_\omega d_0[k] = (1 - hRK_\omega)d_0[k] \tag{11}$$

Now for stability the pole must lie within the unit circle:

$$|1 - hRK_\omega| < 1 \quad \implies \quad -1 < 1 - hRK_\omega < 1$$

$$0 < hRK_\omega < 2 \quad \implies \quad 0 < K_\omega < \frac{2}{Rh} \tag{12}$$

The value of K_ω is tuned experimentally such that it's large enough so that convergence is not too slow, and at the same time not too large that would cause large oscillations about the steady-state value. Another fact to take into account while tuning K_ω is that it should not be too large to avoid requesting for physically impossible actions by the wheel motors.

Task 8 2/2

Since $\omega = 0$, robot will be unable to rotate. Therefore, goal position should in the direction of heading theta for this case. For small K_ω, controller was able to maintain $[x[k], y[k]]$ at $[x_0, y_0]$ Fig 3, but for higher values, it oscillates at the $[x_0, y_0]$. If initial θ is not aligned with the direction to the goal, the robot fails to reach the goal.

What about d0? Would that converge if the heading is wrong?

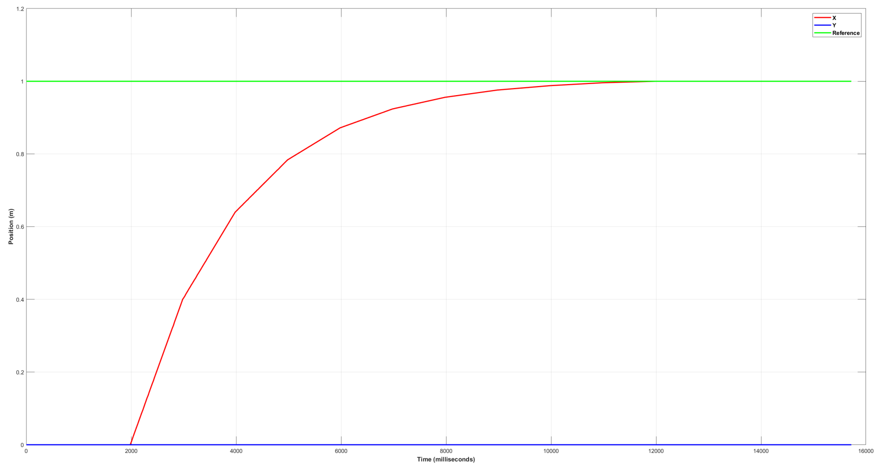


Figure 3: **Performance of straight line controller at** $K_\omega = 0.2 * \frac{2}{Rh}$

Task 9 1/2

With both the controllers enabled, the robot is able to reach any arbitrary point irrespective of its initial state. The trajectory however, is not safe for this controller (not as safe as when heading and position are controlled separately, since it is now possible that the robot enters other regions unwantedly). The major difference is due to rotational controller which always turn the robot towards the goal as θ^R calculated in each loop.

What do you mean by it can reach any arbitrary point? That is not true

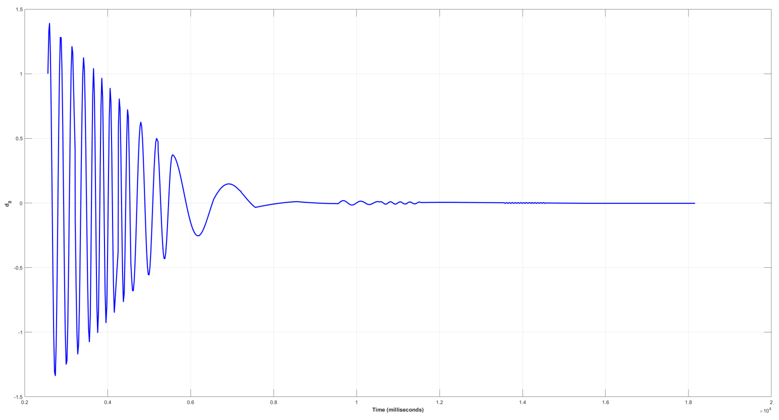


Figure 4: $d_0(m)$ **against** $time(ms)$

The plots are shown in Fig. 4 and Fig. 5

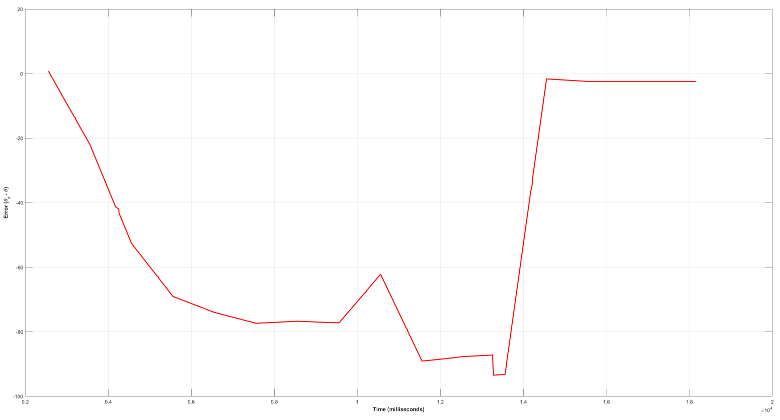


Figure 5: **Error** $\theta^R - \theta$ **with time**

Task 10 2/2

We take the same discretized system shown in (6) in Task 7 and plug in the control law $v[k] = K_\omega d_g[k]$ and the assumption $\theta[k] = \theta_g$:

$$x[k+1] = x[k] + hRK_\omega cos\theta_g d_g[k], \quad y[k+1] = y[k] + hRK_\omega sin\theta_g d_g[k] \tag{13}$$

Again we write the expression for $d_g[k]$ and $d_g[k+1]$:

$$d_g[k] = cos\theta_g x_0 - cos\theta_g x[k] + sin\theta_g y_0 - sin\theta_g y[k] \tag{14}$$

$$d_g[k+1] = cos\theta_g x_0 - cos\theta_g x[k+1] + sin\theta_g y_0 - sin\theta_g y[k+1] \tag{15}$$

Now replace $x[k+1]$ and $y[k+1]$ with their forms found in (13):

$$d_g[k+1] = cos\theta_g x_0 - cos\theta_g x[k] - cos^2\theta_g hRK_\omega d_g[k]$$

$$+sin\theta_g x_0 - sin\theta_g x[k] - sin^2\theta_g hRK_\omega d_g[k] \tag{16}$$

We can see the expression for $d_g[k]$ which could be factorized:

$$d_g[k+1] = d_g[k] - (cos^2\theta_g + sin^2\theta_g)hRK_\omega d_g[k] = (1 - hRK_\omega)d_g[k] \tag{17}$$

Now for stability the pole must lie within the unit circle:

$$|1 - hRK_\omega| < 1 \quad \Longrightarrow \quad -1 < 1 - hRK_\omega < 1$$

$$0 < hRK_\omega < 2 \quad \Longrightarrow \quad 0 < K_\omega < \frac{2}{Rh} \tag{18}$$

The value of K_ω is tuned experimentally such that it's large enough so that convergence is not too slow, and at the same time not too large that would cause large oscillations about the steady-state value. Another fact to take into account while tuning K_ω is that it should not be too large to avoid requesting for physically impossible actions by the wheel motors.

Task 11 2/2

The robot arrives near $[x_g, y_g]$ but never exactly due to the numerical approximation in simulation. In real world however, performance might be even worse due to wheel slip and inaccuracy in motor responses. Fig. 6 shows the plot of xy with time, with heading constant.

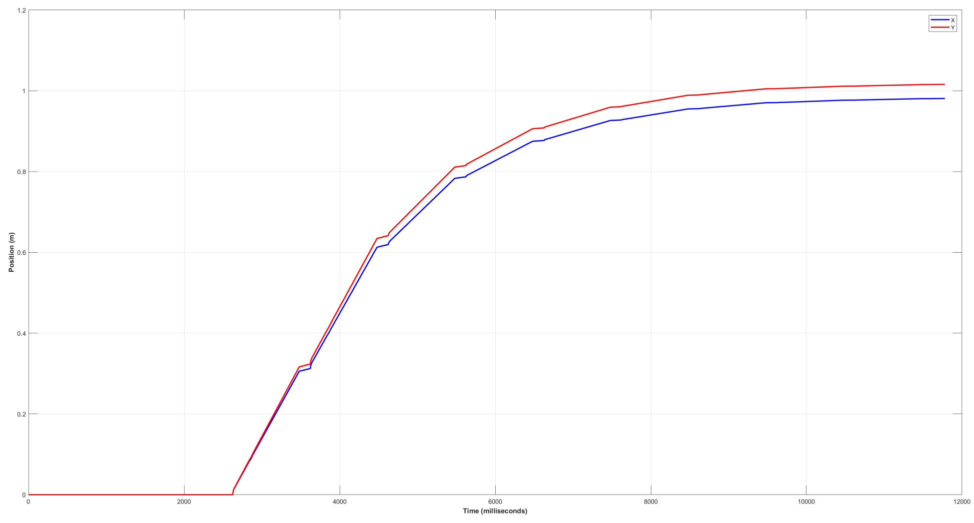


Figure 6: Position vs time(ms)

Task 12 1/2

Based on the assumption that $\theta[k]$ is close to θ_g we can write $d_p[k]$ and $d_p[k+1]$:

$$d_p[k] \approx p\theta_g - p\theta[k] \tag{19}$$

$$d_p[k+1] \approx p\theta_g - p\theta[k+1] \tag{20}$$

Based on the discretized system equation of the heading angle shown in (2) in Task 5, with the control law of $\omega[k] = K_\Psi d_p[k]$ we can re-write $d_p[k+1]$:

$$d_p[k+1] \approx p\theta_g - p\theta[k] - p\frac{RhK_\Psi}{L}d_p[k] \tag{21}$$

We can see the expression for $d_p[k]$ which could be factorized:

$$d_p[k+1] \approx d_p[k] - p\frac{RhK_\Psi}{L}d_p[k] = (1 - \frac{pRhK_\Psi}{L})d_p[k] \tag{22}$$

For stability the pole must lie within the unit circle:

$$|1 - \frac{pRhK_\Psi}{L}| < 1 \quad \Longrightarrow \quad -1 < 1 - \frac{pRhK_\Psi}{L} < 1$$

How did you choose K_\Psi and why?

$$0 < \frac{pRhK_\Psi}{L} < 2 \quad \Longrightarrow \quad 0 < K_\Psi < \frac{2L}{pRh} \tag{23}$$

Task 13 0/2

Value of p presents a trade-off in the ability of the robot to follow the line. The larger the p, the farther horizon that the robot looks up to to correct its orientation. This means the attempt by the controlled to correct the orientation of the robot will be such that correct orientation will be achieved at a longer spatial (and also temporal) horizon (i.e. the convergence will be slower). On the other hand, although with a smaller value of p convergence will be faster, a too small p results in perpetual over-correction of the orientation and oscillations of the robot about the reference line. Therefore, the value of p must be tuned to find a balance in this trade-off.

Look at the influence of p in dp and see what p puts an emphasis on

Task 14 1/2

It's not possible in practice to main d_p exactly at 0, because for this to happen starting from a disturbed state, the robot needs to orient itself perfectly with the direction of the path towards goal, which requires a perfect correction gain that fully fills in the discretization step although this convergence happens asymptotically in theory, in practice is not likely to happen. However, performance is considered good enough in practice.

What do you mean by perfect correction gain?

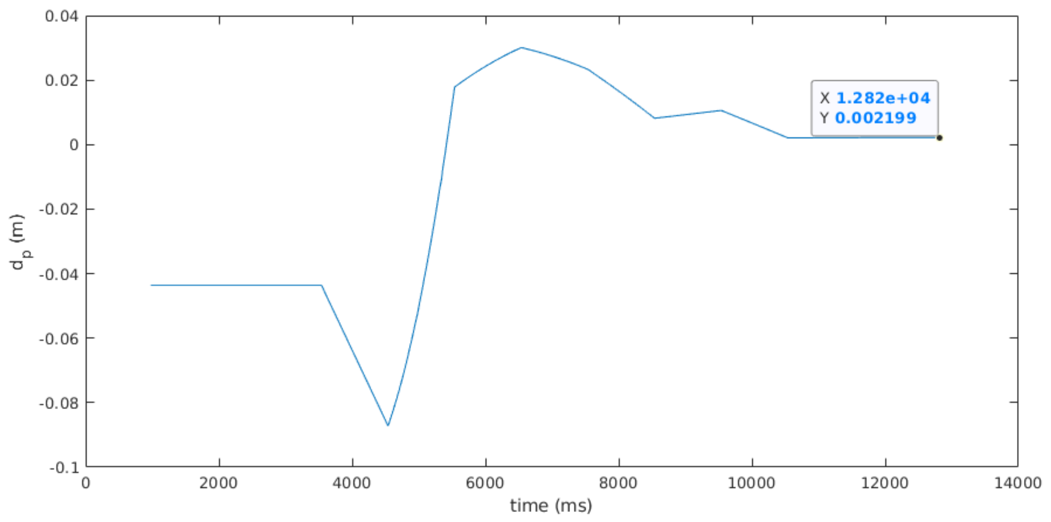


Figure 7: $d_p(m)$ against $time(ms)$ for correction of a 90° error with $K_\Psi = 0.3 * \frac{2L}{Rh}$

Task 15 2/2

Controller can fix the small deviations of robot from desired straight line and heads towards the goal. A difference that exists now between this case and the cases where only one controller is active is that here the action from one controller acts as a disturbance to the other controller. Therefore, there are small levels of noise observed in the responses (even after convergence, since none of the controllers can reduce the error exactly to zero).

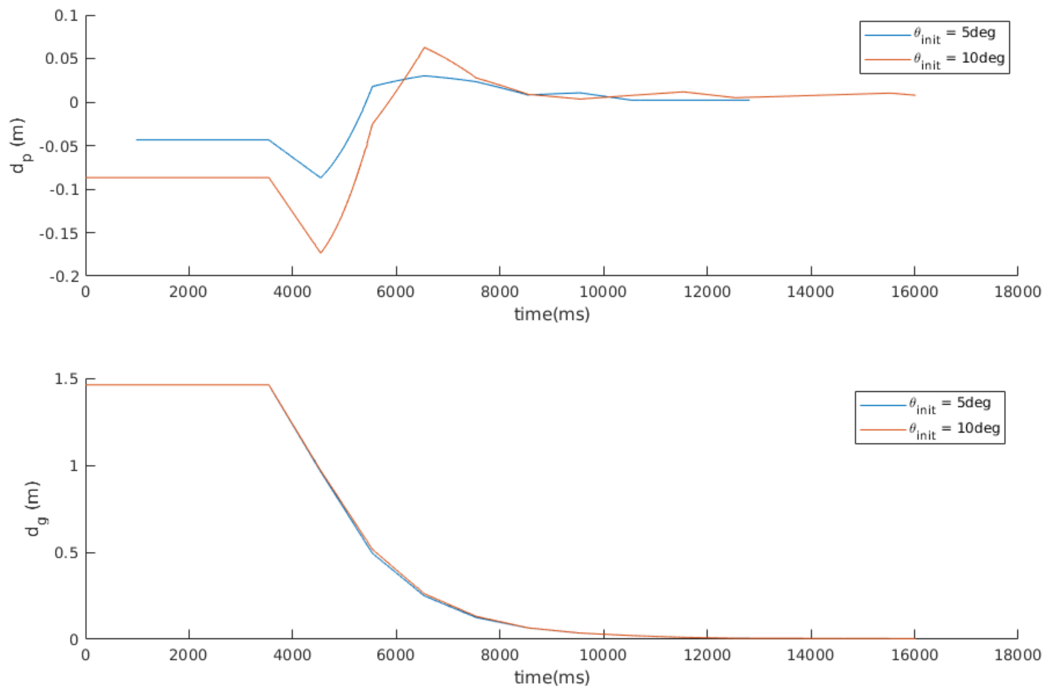


Figure 8: $d_p(m)$ and $d_g(m)$ against $time(ms)$ for $\theta_{init} = 5°$ and $10°$, with $K_\omega = 0.25 * \frac{2}{Rh}$

Task 16 1/2

The Hybrid controller can be described using the following hybrid automaton of 8-tuple

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

- $Q \equiv \{q_1, q_2, q_3\}$ denotes the set of discrete states. The state q_1 corresponds to pure rotation controller, the state q_2 corresponds to line-following controller, and the state q_3 is the stopped condition.
- $Init \equiv \{q_3\}$. The initial state is taken to be q_3, i.e stopped condition.
- $X \equiv \{(x, y, \theta) \in \mathbb{R}^3 : \theta \in (-180°, 180°]\}$ denotes the state space.
- Since, the controllers are discrete in nature, the vector fields f are given by.

$$f(q_1, X) = \begin{bmatrix} x[k+1] = x[k] \\ y[k+1] = y[k] \\ \theta[k+1] = \theta[k] + \frac{RhK_\Psi}{L}(\theta_g - \theta[k]) \end{bmatrix}$$

No discrete dynamics allowed.

where, θ_g is given by $atan(\frac{y_g - y_0}{x_g - x_0})$,

$$f(q_2, X) = \begin{bmatrix} x[k+1] = x[k] + hRK_\omega cos\theta_g d_g[k] \\ y[k+1] = y[k] + hRK_\omega sin\theta_g d_g[k] \\ \theta[k+1] = \theta[k] + \frac{RhK_\Psi}{L} d_p[k] \end{bmatrix}$$

where,

$$d_g[k] = cos\theta_g x_0 - cos\theta_g x[k] + sin\theta_g y_0 - sin\theta_g y[k]$$
$$d_p[k] = sin(\theta_g)(x[k] + p.cos(\theta[k]) - x_0) - cos(\theta_g)(y[k] + p.sin(\theta[k]) - y_0)$$

$$f(q_3, X) = \begin{bmatrix} x[k+1] = x[k] \\ y[k+1] = y[k] \\ \theta[k+1] = \theta[k] \end{bmatrix}$$

- D shows what conditions need to be satisfied in order for the automaton to stay in a state.

  $D(q_1) = \{(x, y, \theta) : x, y \in \mathbb{R}^2, |\theta_g - \theta| > \delta\}$

  $D(q_2) = \{(x, y, \theta) : x, y \in \mathbb{R}^2, \sqrt{(x_g - x)^2 + (y_g - y)^2} > \xi\}$

  $D(q_3) = \{x, y \in \mathbb{R}^2, x, y \in \mathbb{R}^2, \sqrt{(x_g - x)^2 + (y_g - y)^2} \sim 0, \}$

- E: The edges show which transitions are possible.

  $E_1 = \{q_1, q_2\}$

  $E_2 = \{q_2, q_3\}$

  $E_3 = \{q_3, q_1\}$

  $E = E_1 \cup E_2 \cup E_3$

- G: The guards show under what conditions the system can transition from one to another state.

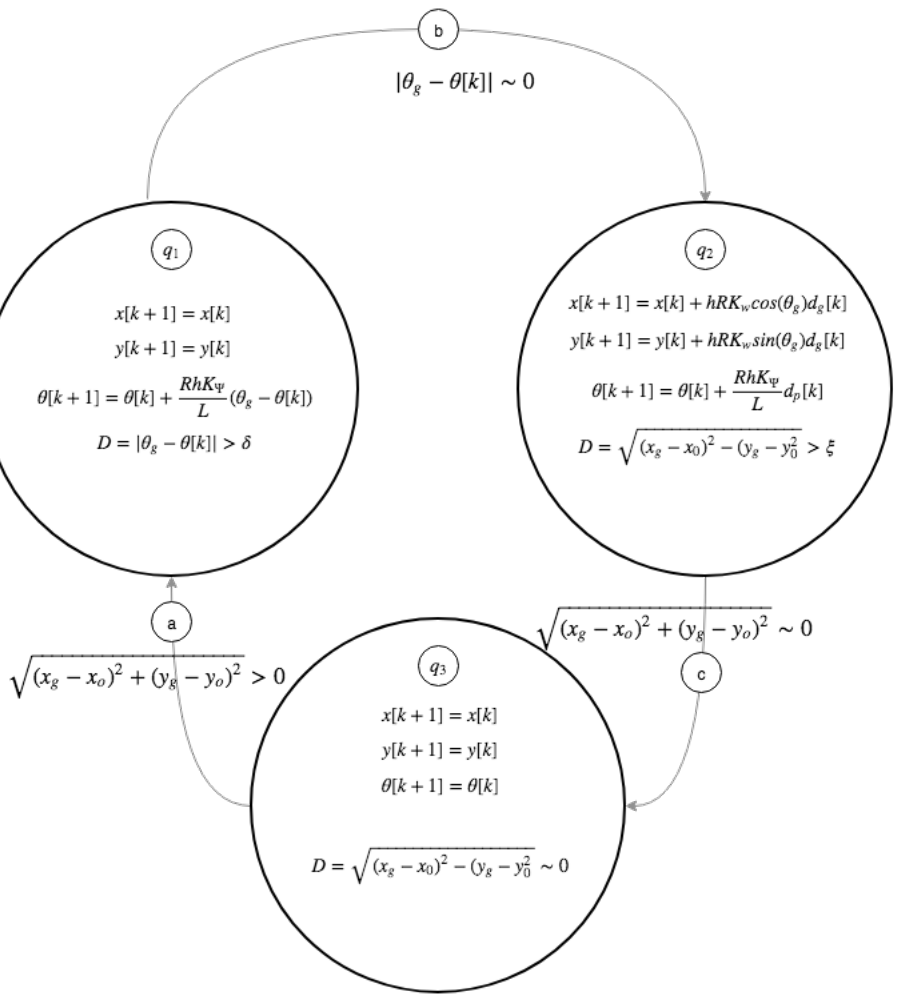
  $G(\{q_1, q_2\}) = \{|\theta_g - \theta| \leq \delta\}$

  $G(\{q_2, q_3\}) = \{\sqrt{(x_g - x)^2 + (y_g - y)^2} \leq \xi\}$

  $G(\{q_3, q_1\}) = \{\sqrt{(x_g - x)^2 + (y_g - y)^2} > \xi\}$

- R: No reset is utilised in the hybrid controller

  $R = \{x, y, \theta\}$



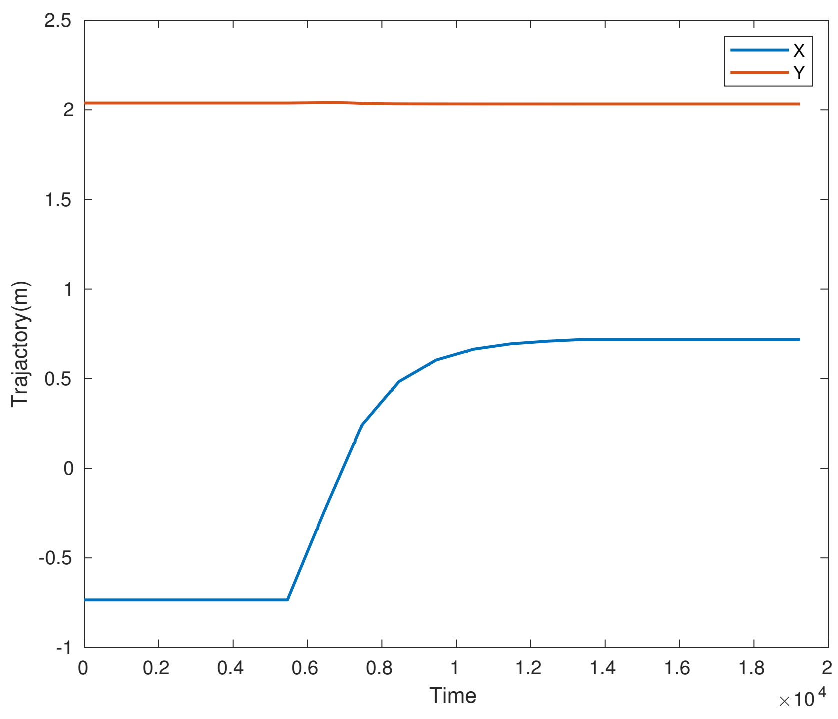
b
$|\theta_g - \theta[k]| \sim 0$
q_1
$x[k+1] = x[k]$
$y[k+1] = y[k]$
$\theta[k+1] = \theta[k] + \frac{RhK_\Psi}{L}(\theta_g - \theta[k])$
$D = |\theta_g - \theta[k]| > \delta$
q_2
$x[k+1] = x[k] + hRK_w cos(\theta_g) d_g[k]$
$y[k+1] = y[k] + hRK_w sin(\theta_g) d_g[k]$
$\theta[k+1] = \theta[k] + \frac{RhK_\Psi}{L} d_p[k]$
$D = \sqrt{(x_g - x_0)^2 - (y_g - y_0^2} > \xi$
a
$\sqrt{(x_g - x_o)^2 + (y_g - y_o)^2} > 0$
$\sqrt{(x_g - x_o)^2 + (y_g - y_o)^2} \sim 0$
c
q_3
$x[k+1] = x[k]$
$y[k+1] = y[k]$
$\theta[k+1] = \theta[k]$
$D = \sqrt{(x_g - x_0)^2 - (y_g - y_0^2} \sim 0$

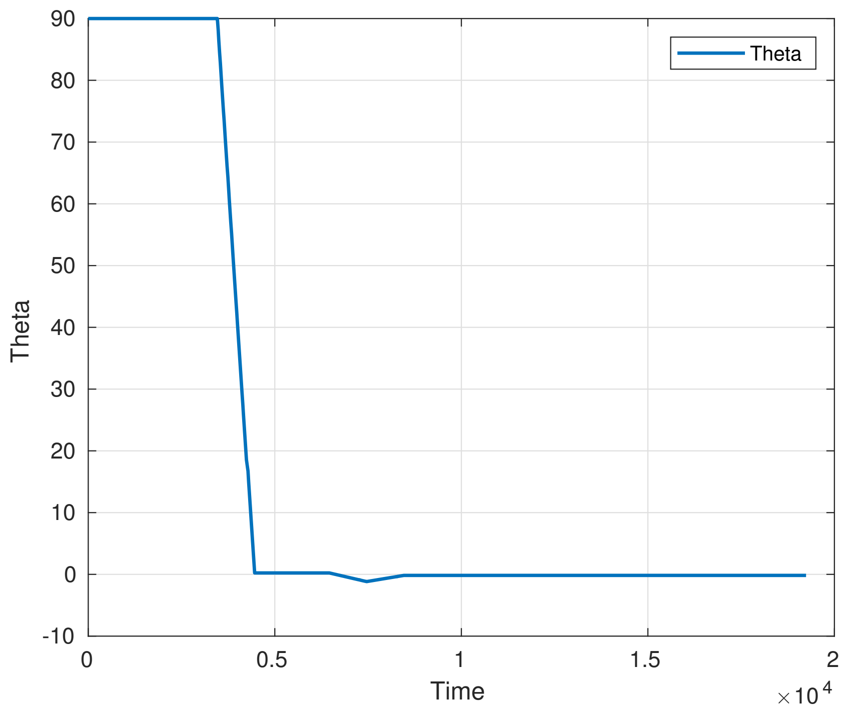
The dynamics in a discrete state of the hybrid controller need to be continuous

Figure 9: Hybrid Controller

Task 17 2/2

As Fig 10 shows, the behavior of robot is as expected. The robot first orient itself towards the goal follwed by a transition to go-to-goal state. Here robot navigates towards the goal while correcting for possible deviations from derired line.


X
Y
Trajactory(m)
Time
$\times 10^4$


Theta
Theta
Time
$\times 10^4$

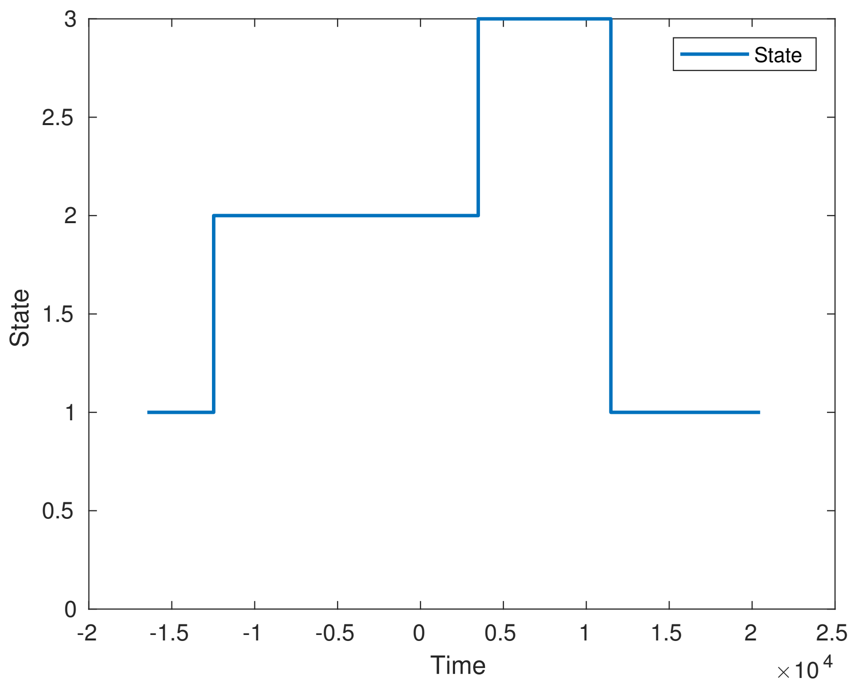


Figure 10: Continuous and Discrete state trajectory

Task 18 2/2

Fig. 11 shows the trajectory of the robot in xy plane when it was provided the waypoints derived in *Task 3*. Fig. 12 shows the evolution of θ. The waypoints were taken as -

[[-1.25, 1.25;] [-0.75, 1.25;] [-0.75, 1.25;] [-0.25, 1.25;] [-0.25, 1.25;] [-0.25, 0.75;] [-0.25, 0.75;] [0.25, 0.75;] [0.25, 0.75;] [0.25, 0.25;]]

The robot follows the desire trajectory without going into other "bad" regions. The performance of the Hybrid controller is much better than the combined controller implemented previously.

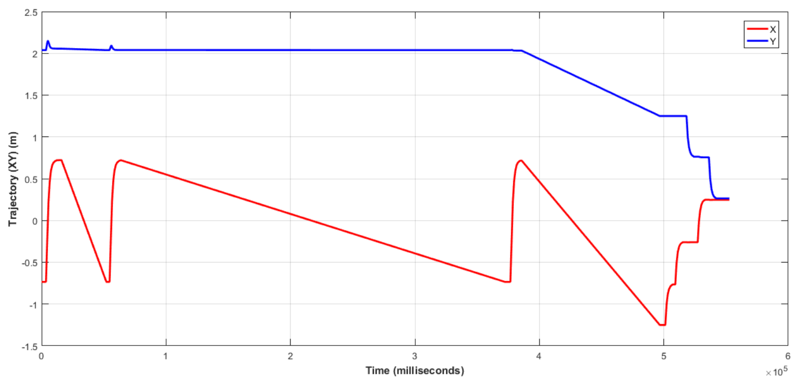


Figure 11: **Trajectory of xy with time for waypoint control**

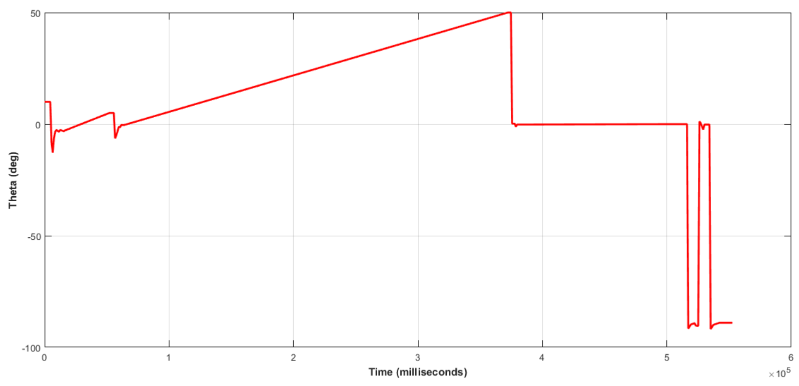


Figure 12: **Trajectory of θ with time**

Task 19 1/1

The Hybrid controller ensures the robot to be in regions of transition i.e the one it starts from and the one it wants to go. Unlike the continuous controller implemented in *Task 9*, where the robot may go it undetermined states. In the hybird controller, when the orientation gets corrected in the initial state itself and the go-to-goal controller ensures the robot goes in straight line while correcting the slight deviation in path. This results the safe behaviour of the robot.