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

Sujet Phodapol
19980626-9693
sujet@kth.se

Shengyang Zhuang
20000802-T172
szhuang@kth.se

Mimmi Lindgren
940214-2880
mimmili@kth.se

Beatriz Lourenço
20000607-T161
bebl@kth.se

February 28, 2022

Task 1

If v and ω are given, we can compute u_r and u_l from the following equations:

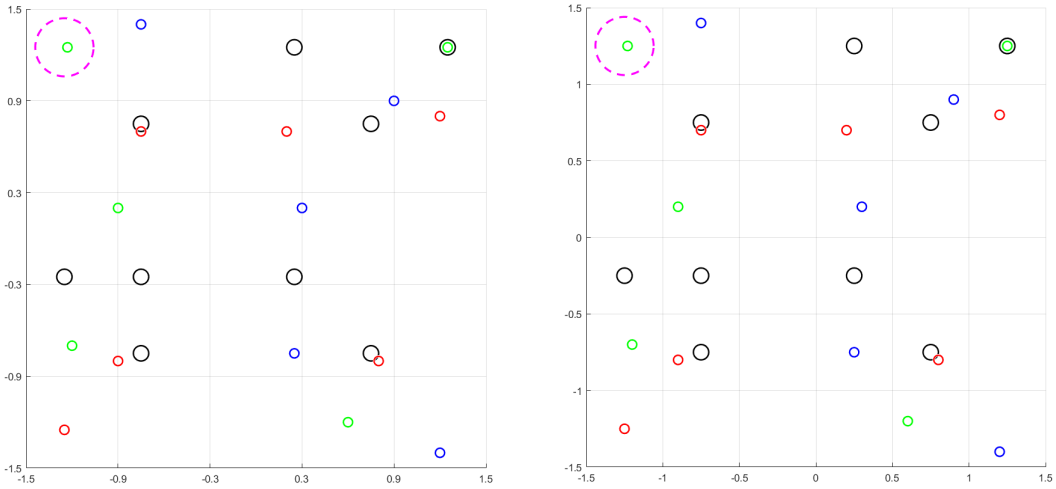
$$\begin{cases} v = \frac{u_r+u_l}{2} \\ \omega = u_r - u_l \end{cases} \implies \begin{cases} u_r = \frac{2v+\omega}{2} \\ u_l = \frac{2v-\omega}{2} \end{cases}$$

Task 2

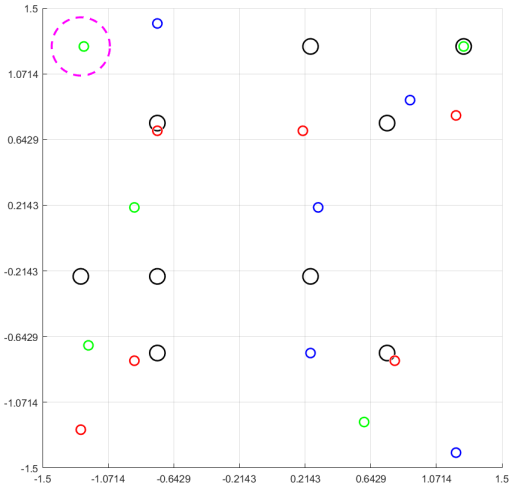
From the specification, we can propose a Transition System as $\mathcal{T} = (S, S_0, \Sigma, \rightarrow, AP, \mathcal{L})$.

- $S = \{R_1, \ldots, R_K\}$,where S is a set of states, K is the number of discretized rectangular regions.
  Here the determination of the value K should satisfy the desired robot behavior.
  ▶ **case 1:** If we discrete the workspace into 5×5 rectangular regions, as shown in (a) of Fig.1, the robot will inevitably go to a "obstacle" region when achieving its mission.
  ▶ **case 2:** If we discrete the workspace into 6×6 rectangular regions, as shown in (b) of Fig.1, it will be the optimal value of K. Meanwhile, the area of the region also satisfy the condition $0.5^2 > 0.38^2$.
  ▶ **case 3:** If we discrete the workspace into 7×7 rectangular regions, as shown in (c) of Fig.1, the robot will inevitably go to a "obstacle" region when achieving its mission.
- $S_0 \subseteq S$: initial state $S_0 = R_6$.
- $\Sigma = \{up, down, left, right, stop\}$: is a set of control inputs.
- $\rightarrow \subset S \times \Sigma \times S$: $(s, \sigma, s') \in \rightarrow$ represents the transitions $s \xrightarrow{\sigma} s'$ with the action $\sigma \in \Sigma$. s, s' are all the states of the robot in the workspace.

- $AP = \{blue, red, green, obstacle\}$, where AP(= Π) is a finite set of atomic propositions.
- $\mathcal{L} : S \to 2^{\Pi}$, labeling each region with its atomic propositions that are true in each state.



(a) 5×5 (b) 6×6



(c) 7×7

Figure 1: The workspace dicretized in three cases: 5x5, 6x6 and 7x7.

A graphical illustration of the discretized workspace along with the atomic propositions Π that are true in each state is shown in Fig.2.

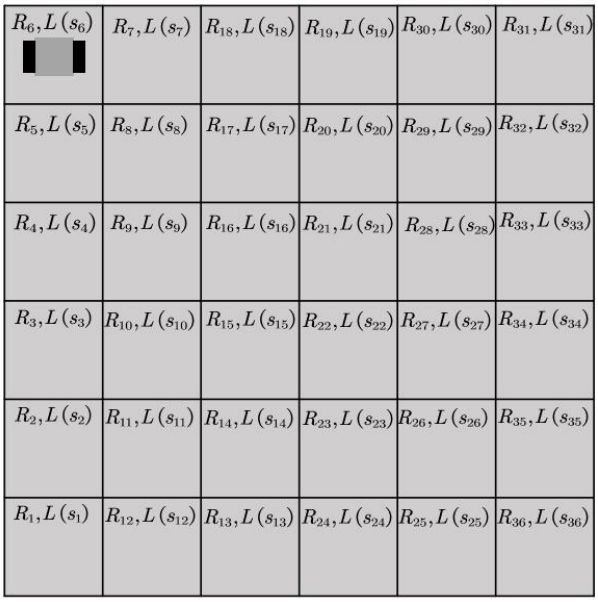


Figure 2: Workspace discretization in rectangular regions

Task 3

A path, as an infinite sequence of states of the Transition System, that satisfies the aforementioned behavior, is as follows, shown in Fig.3:

$$(R_6, right, R_7) \Rightarrow (R_7, right, R_{18}) \Rightarrow (R_{18}, down, R_{17})$$
$$\Rightarrow (R_{18}, down, R_{17}) \Rightarrow (R_{17}, right, R_{20}) \Leftrightarrow (R_{20}, down, R_{21})$$

i.e.

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

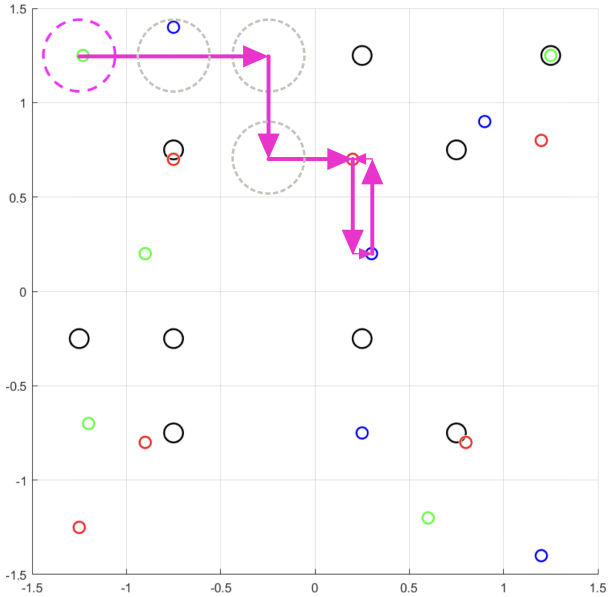


Figure 3: A path that satisfies the behavior

Task 4

For the safe property, we choose the grid size to be larger than the robot size so that we can guarantee that the robot will always stay within the region while moving. The problem if we enable both rotation and go-to-goal simultaneously is that the trajectory might be a curve line, therefore the robot might collide with the obstacle in unintended region.

Task 5

From the continuous time system, we can describe the changing of θ over time, considering ω as an input, by

$$\dot{\theta}(t) = \frac{R}{L}\omega(t)$$
$$A = 0, \quad B = \frac{R}{L}$$

Therefore, we can derive the system in to the discrete system by

$$\Phi = e^{Ah} = 1$$
$$\Gamma = \int_0^h e^{As} ds B = \frac{Rh}{L}\omega$$
$$\theta[kh+h] = \Phi\theta[kh] + \Gamma\omega[kh]$$
$$\theta[kh+h] = \theta[kh] + \frac{Rh}{L}\omega[kh]$$

Applying the control input $\omega[k] = K_\Psi(\theta^R - \theta[k])$, the closed-loop discrete system is described by

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

Eigenvalue of the system is $\lambda = 1 - K_\Psi\frac{Rh}{L}$. In order to make it asymptotically stable, eigenvalue must stay inside the unit circle. In other words,

$$-1 < 1 - K_\Psi\frac{Rh}{L} < 1$$
$$0 < K_\Psi < \frac{2L}{Rh}$$

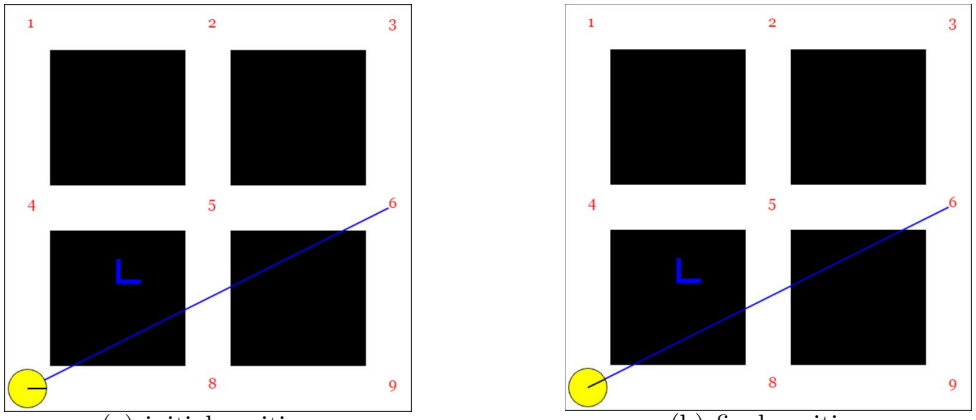
Therefore, the maximal value is $K_\Psi = \frac{2L}{Rh}$.

However, according to the classical control theory, the gain value K_Φ we choose should make sure that the system has the best possible dynamic performance, including fast rising time, short settling time and small overshoot. Hence, if we want the system to be non-oscillating, the maximal value is $K_\Psi = \frac{L}{Rh}$.

Task 6

After implementing the controller by filling the codes of `OwnVariables.c`, `Controller.c` and `RenewControllerState.c`, compiling them in the online simulator, the robot can act properly.

In this task, we set $\omega = K_\Psi \left(\theta^R - \theta\right), v = 0$. For instance, we set the initial position of the robot to be node 7 and its goal position to be node 6. After pressing the button "set preference" the robot will successfully change its orientation θ such that it is close to the desired angle θ^R, meanwhile it will maintain its position constant, shown in Fig.4. According to the log data and the coordinates of the nodes, the final relative orientation $\theta = 26°$, that is consistent with the desired angle $\theta^R = \arctan \frac{0.5912-(-0.8558)}{2.1884-(-0.7346)} = 26.337°$, which is yielded through the coordinates of node 7 and node 6. Considering the calculation error and the plotting error see in Fig.5, therefore, we can maintain $\theta[k]$ at θ^R exactly.



(a) initial position (b) final position

Figure 4: Simulation result of the rotation controller when setting $v = 0$

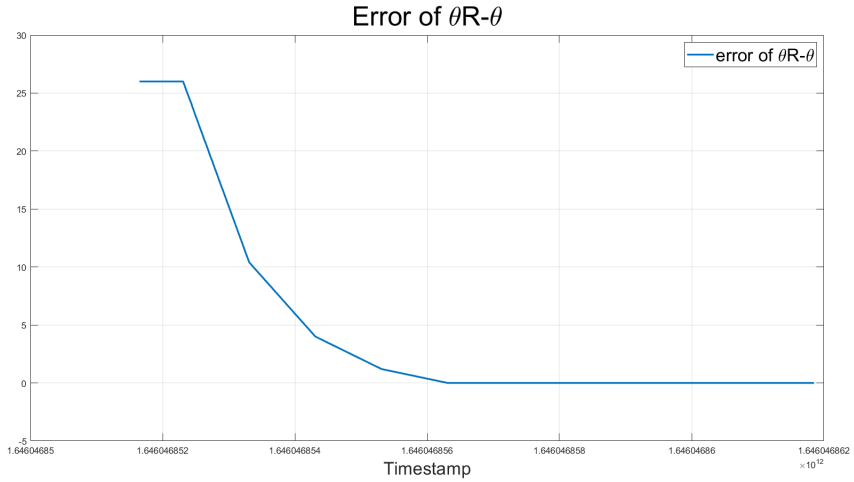


Figure 5: Error of $\theta^R - \theta$

Task 7

From the definition, we can describe vector d_0 in continuous time by

$$d_0 = \cos\theta(x_0 - x) + \sin\theta(y_o - y)$$

Therefore, we can get the derivation of d_0 by

$$\dot{d_0} = -\dot{x}\cos\theta - \dot{y}\sin\theta$$

Substituting $\dot{x} = Rv\cos\theta$ and $\dot{y} = Rv\sin\theta$ yields

$$\begin{aligned}\dot{d_0} &= -Rv\cos^2\theta - Rv\sin^2\theta \\ \dot{d_0} &= -Rv\end{aligned}$$

We can see that $A = 0$ and $B = -R$. Therefore, we can derive the system in to the discrete system by

$$\begin{aligned}\Phi &= e^{Ah} = 1 \\ \Gamma &= \int_0^h e^{As}dsB = -Rh \\ d_0[kh+h] &= \Phi d_0[kh] + \Gamma v[kh] \\ d_0[kh+h] &= d_0[kh] - Rhv[kh]\end{aligned}$$

Given the control input $V[k] = K_\omega d_0[k]$, the closed-loop discrete system is described by

$$\begin{aligned}d_0[kh+h] &= d_0[kh] - RhK_\omega d_0[k] \\ d_0[kh+h] &= (1 - K_\omega Rh)d_0[kh]\end{aligned}$$

In order to make it asymptotically stable, eigenvalue $\lambda = 1 - K\omega Rh$, must stay inside the unit circle. Hence, the boundary of the gain can be derived by

$$\begin{aligned}-1 < 1 - K_\omega Rh < 1 \\ 0 < K_\omega < \frac{2}{Rh}\end{aligned}$$

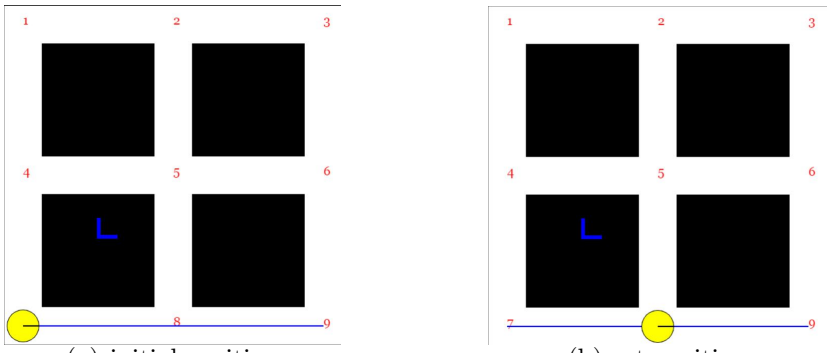
Thus the maximal value is given by $K\omega = \frac{2}{Rh}$.

Similarlly, if we want the system to be non-oscillating, the maximal value is $K\omega = \frac{1}{Rh}$.

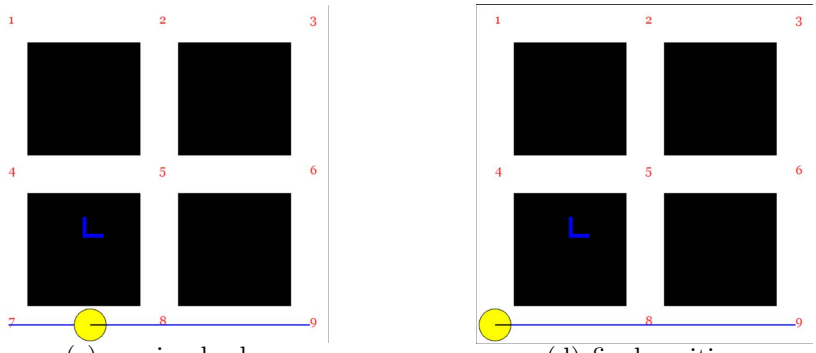
Task 8

In this task, we set $v = K_\omega(\cos(\theta)(x_0 - x) + \sin(\theta)(y_0 - y)), \omega = 0$. For instance, we set the initial position of the robot to be node 7 and its goal position to be node 9. After pressing the button "set preference" and meanwhile set the robot position to node 8, the robot would successfully move to back to initial node 7, shown in Fig.6.
Since the coordinates of the initial point is node 7, which is $(x_0 = -0.7346, y_0 = -0.8558)$, the error of d_0 is defined by the following formula

$$d_0 = \cos(\theta)(x_0 - x) + \sin(\theta)(y_0 - y)$$



(a) initial position (b) set position



(c) moving back (d) final position

Figure 6: Simulation result of the rotation controller when setting $\omega = 0$

Plot the error through MATLAB, as shown in Fig.7, the error of d_0 converges to 0.

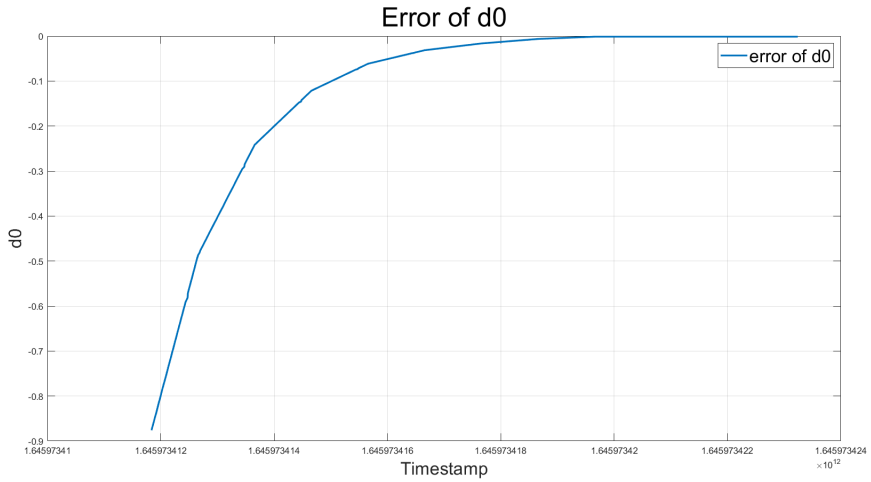


Figure 7: Error of d_0

Therefore, it is possible to maintain $[x[k], y[k]]$ at $[x_0, y_0]$ exactly.

Task 9

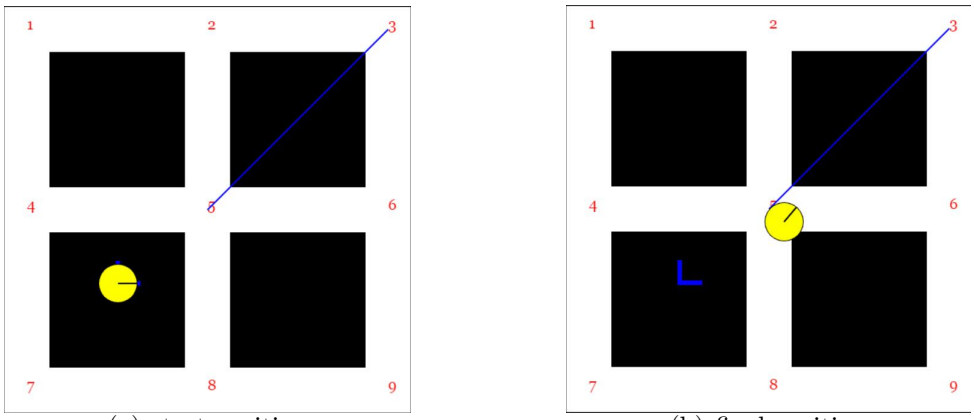
Now we enable both controllers together, i.e.

$$\omega = K_\Psi \left(\theta^R - \theta\right)$$
$$v = K_\omega(\cos(\theta)(x_0 - x) + \sin(\theta)(y_0 - y))$$

Then we tested the control performance. For instance, we set node 5 to node 3 as preference. To evaluate,

- Whether the robot could change its orientation θ such that it is close to the desired angle θ^R.
- Whether the robot could maintain its location constant while rotating.

we set the robot's initial position to the origin point $(0, 0)$ and check the two criterion below when the controller tried to order the robot move back to the start position, as shown in Fig.8.



(a) start position (b) final position

Figure 8: Simulation result of the rotation controller when two controllers are enabled

Plot the error through MATLAB, as shown in Fig.9 and Fig.10.

Figure 9: Error of $\theta^R - \theta$

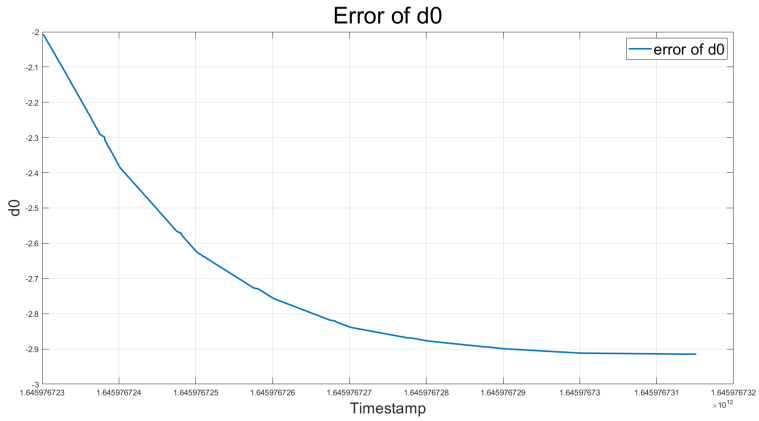


Figure 10: Error of d_0

From Fig.8 we can see that the robot moved from the initial position (original point) towards preference start position (node 5). In the duration, the robot would rotate to the desired orientation and at the same time move towards the start position to reduce the distance error d_0. However, from (b) of Fig.8 and Fig.10 we can see that d_0 asymptotically became stable at the value of 1 but due to the error in its position it did not become 0. And that situation is similar to θ.

Task 10

The assumption is that the robot is correctly headed, $\theta[k] = \theta_g$. Designing a proportional controller $v[k] = K_\omega d_g[k]$ where,

$$d_g[k] = (v_g^T)(\Delta g[k]), \quad v_g = [\cos(\theta g), \sin(\theta g)]^T, \qquad \theta_g = \angle([x_0, y_0]^T, [x_g, y_g]^T)$$

$d_g[k]$ can be written as

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

Derivation and substitution:

$$\begin{aligned} \dot{d}_g &= -\dot{x}\cos(\theta) - \dot{y}\sin(\theta) \\ \dot{x} &= Rv\cos\theta, \qquad \dot{y} = Rv\sin\theta \\ \dot{d}_g &= -Rv\cos^2(\theta) - Rv\sin^2(\theta) = -Rv \end{aligned}$$

Periodic sampling with $h > 0$ gives a linear time-invariant system:

$$d_g[kh+h] = \Phi d_g[kh] + \Gamma v[kh]$$

$$\Gamma = \int_0^h e^{As} ds B = \{_{A=0,B=-R}\} = [s]_0^h(-R) = -Rh, \quad \Phi = e^{Ah} = \{_{A=0}\} = e^0 = 1$$

$$\begin{aligned} d_g[kh+h] &= d_g[kh] - Rhv[kh] = \{v[k] = K_\omega d_g[k] \to v[kh] = K_\omega d_g[kh]\} = \\ &= d_g[kh] - RhK_\omega d_g[kh] = (1 - RhK_\omega) d_g[kh] \end{aligned}$$

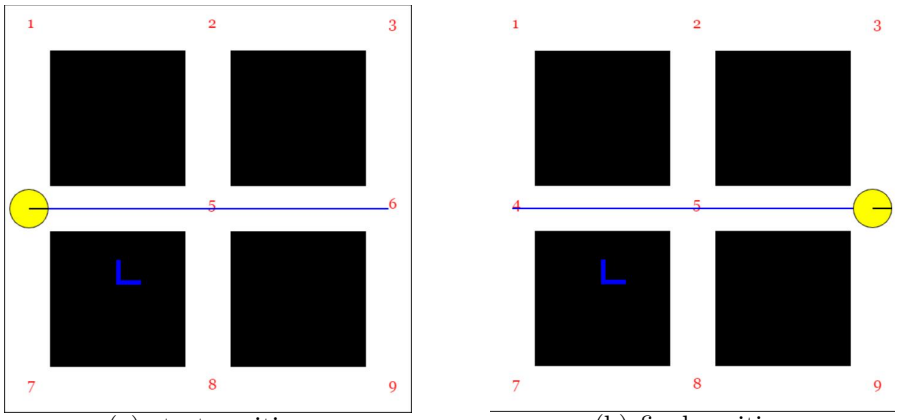
To achieve stability in the system, the value has to be inside the unit circle, like:

$$|1 - RhK_\omega| < 1 \to 0 < K_\omega < \tfrac{2}{Rh}$$

The maximum value for $K_\omega = \frac{2}{Rh}$, when $d_g[k]$ converges to zero.

Task 11

Implemented controller from task 10 and tested it on online simulator. In this task, we set the start position to be node 4 and the goal position to be node 6. We can see from Fig.11 that the robot could exactly arrived at the desired point. Note though that the controller could only work when the robot is in the heading towards the goal. location



(a) start position (b) final position

Figure 11: Simulation of task 11. $\omega = 0$

To further illustrate the conclusion, we draw the error graph through MATLAB, shown in Fig.12. It can be seen that the controller could allow the robot to arrive at $[x_g, y_g]$ exactly.

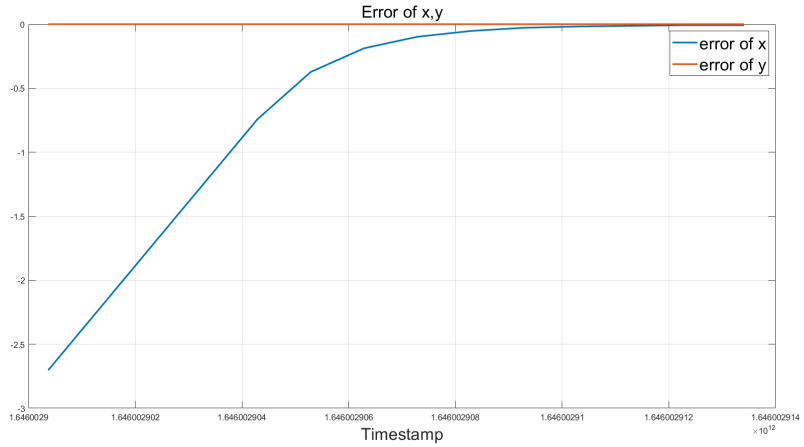


Figure 12: Error of $(x, y), (x_g, y_g)$

Task 12

Design a proportional controller $\omega[k] = K_\Psi d_p[k]$.

$$d_p[k] = (v_{g,\perp}^T)(v_p[k]), \qquad v_{g,\perp} = [\sin\theta_g, -\cos\theta_g]^T$$
$$v_p[k] = [x[k] + p \cdot \cos(\theta[k]) - x_0, y[k] + p \cdot \sin(\theta[k]) - y_0]^T$$

Where $p > 0$ is a design parameter.

$$\begin{aligned} d_p[k] &= [\sin\theta_g, -\cos\theta_g] \begin{bmatrix} x[k] + p \cdot \cos(\theta[k]) - x_0 \\ y[k] + p \cdot \sin(\theta[k]) - y_0 \end{bmatrix} \\ &= \sin\theta_g(x[k] - x_0) + \sin\theta_g(p \cdot \cos(\theta[k]) - \cos\theta_g(y[k] - y_0) - \cos\theta_g(p \cdot sin(\theta[k]) \\ &= \sin\theta_g(x[k] - x_0) - \cos\theta_g(y[k] - y_0) + p(sin\theta_g \cos\theta[k] - \cos\theta_g \sin\theta[k]) \\ &= \sin\theta_g(x[k] - x_0) - \cos\theta_g(y[k] - y_0) + p \cdot \sin(\theta_g - \theta[k]) \end{aligned}$$

We can approximate $d_p[k] \approx p \cdot (\theta_g - \theta[k])$ if we assume that $\theta[k]$ is close to θ_g.

$$\begin{aligned} \dot{d_p} &= -p \cdot \dot{\theta} = \{\dot{\theta} = \tfrac{R}{L}\omega\} = -p \cdot \tfrac{R}{L}\omega \\ d_p[kh + h] &= \Phi d_p[kh] + \Gamma\omega[kh] \\ \Phi &= e^{Ah} = 1, \qquad \Gamma = \int_0^h e^{As} ds B = -p\frac{R}{L}h \\ d_p[kh + h] &= d_p[kh] - p\frac{R}{L}\omega[kh] \\ &= \{\omega[k] = K_\Psi d_p[k] \rightarrow \omega[kh] = K_\Psi d_p[kh]\} \\ &= d_p[kh] - \frac{pRh}{L}K_\psi d_p[kh] = (1 - \frac{pRh}{L}K_\Psi)d_p[kh] \end{aligned}$$

The unit circle gives the boundary $|1 - \frac{pRh}{L}K_\psi| < 1$. The boundary for K_ψ is then:

$$0 < K_\psi < \tfrac{2L}{pRh}$$

Maximum value of $K_\Psi = \frac{2L}{pRh}$, when $d_p[k]$ converges to zero.

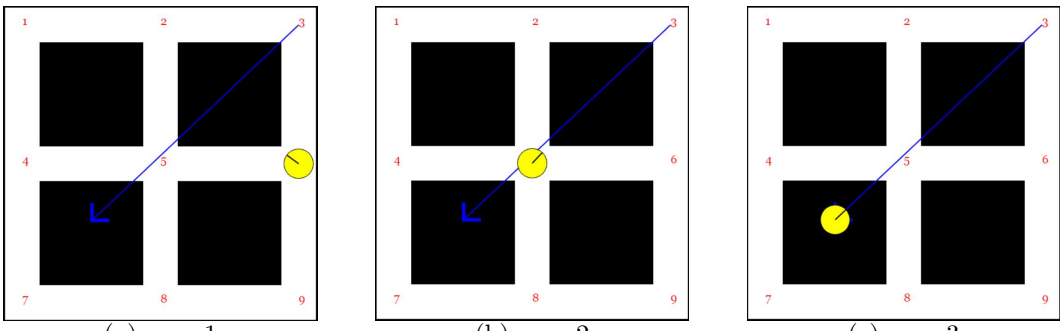
Task 13

Consider the maximum value of $K_\psi = \frac{2L}{pRh}$,

- If we take the larger value of p that means K_ψ will have ample range of values. Therefore, the robot's ability to follow the line will be better i.e. system is stable in this scenario.
- If we take the smaller value of p that means it will be harder for the robot to follow the line.

Task 14

In this task, we set $v = 0$ and the value of ω is consistent with task 13. On simulating the controller, we see that when the reference line was given, if the robot was near the reference line, it would aligns itself to it(See (b)(c) of Fig.13). On the contrary, it would rotate into a wrong direction. See (a) of Fig.13.



(a) case 1 (b) case 2 (c) case 3

Figure 13: Simulation result of the controller when setting $v = 0$.

Task 15

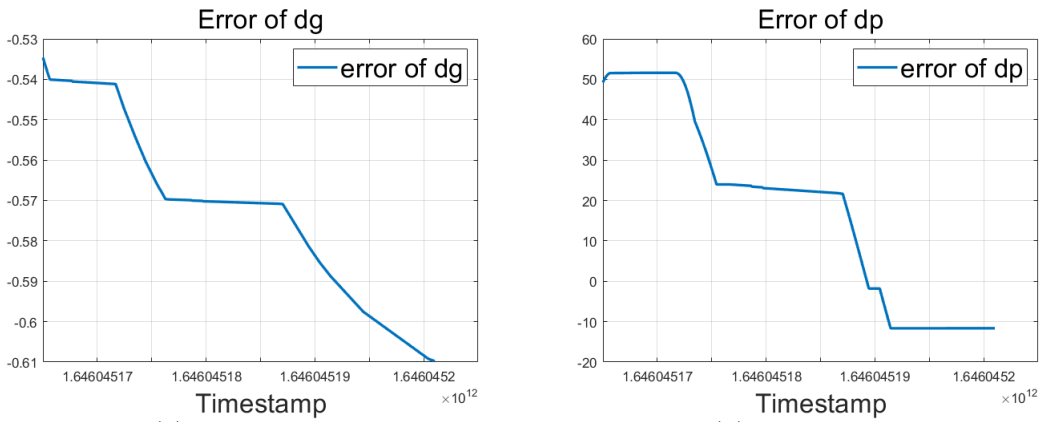
Now we enable both controllers together, i.e.

$$\omega = K_\Psi(\sin(\theta_g)(x + p\cos(\theta) - x_0) - \cos(\theta_g)(y + p\sin(\theta) - y_0))$$
$$v = K_\omega(\cos(\theta_g)(x_g - x) + \sin(\theta_g)(y_g - y))$$

Set $K_\Psi = 1, K_\omega = 1, p = 50$. Then we tested the control performance. For instance, we set node 7 to node 3 as preference. Here we have $\theta_g = \arctan\frac{0.5912+0.8558}{2.1884+0.7346} = 26.337°$. Therefore, the error for d_g and d_p satisfy

$$d_g = \cos(\theta_g)(x_g - x) + \sin(\theta_g)(y_g - y)$$
$$d_p = \sin(\theta_g)(x + p\cos(\theta) - x_0) - \cos(\theta_g)(y + p\sin(\theta) - y_0)$$

Plot the error graph through MATLAB, see Fig.14



(a) error of d_g (b) error of d_p

Figure 14: Simulation result of the line following controller

The difference between one controller and two controllers:

- If we enable both controllers together, there are small levels of noise and disturbance observed in the responses.

Task 16

The constructed Hybrid Controller that navigates from the initial state $[x_0, y_0, \theta_0]$ to goal positions $[x_g, y_g]$, $H = (Q, X, Init, f, D, E, G, R)$: ($\delta$ = a small number.)

- $Q = \{q_1, q_2, q_3\}$ Discrete state spaces. The three different of the controller: Translational(q_2), rotational(q_1) and the end/goal position(q_3).
- $X = \{[x, y, \theta] \in \Re^3; \theta \in (-180°, 180°]\}$ Continuous state space. Rotation of the robot θ.
- $Init = \{q_1, x_0, y_0, \theta_0\}$ All initial states. Starting in rotational state, and $[x_0, y_0, \theta_0]$ initial positioning.
- $f = \begin{cases} f(q_1, X) = \{\dot{x}, \dot{y}, \dot{\theta}\} = \{Rv\cos\theta, Rv\sin\theta, \frac{R}{L}\omega\} \\ f(q_2, X) = \{\dot{x}, \dot{y}, \dot{\theta}\} = \{Rv\cos\theta, Rv\sin\theta, \frac{R}{L}\omega\} \\ f(q_3, X) = \{\dot{x}, \dot{y}, \dot{\theta}\} = \{0, 0, 0\} \end{cases}$ Vector fields
- $D: \begin{cases} D(q_1) = \{|\theta - \theta_0| \geq \delta\} \\ D(q_2) = \{|(x_g, y_g) - (x, y)| \geq \delta\} \\ D(q_3) = \{|(x_g, y_g) - (x, y)| < \delta, |\theta - \theta_0| < \delta\} \end{cases}$ Domains, to stay in this state.
- $E = \{(q_1, q_2), (q_2, q_3), (q_3, q_1)\}$ Edges, possible transitions between states.
- $G = \begin{cases} G(q_1, q_2) = \{|\theta - \theta_0| < \delta\} \\ G(q_2, q_3) = \{|(x_g, y_g) - (x, y)| < \delta\} \\ G(q_3, q_1) = \{|(x_g, y_g) - (x, y)| \geq \delta; |\theta - \theta_0| \geq \delta\} \end{cases}$
  Guards, switching are allowed.
- $R = \begin{cases} R(q_1, q_2, X) = \{x, y, \theta\} \\ R(q_2, q_3, X) = \{x, y, \theta\} \\ R(q_3, q_1, X) = \{x, y, \theta\} \end{cases}$ Reset state.

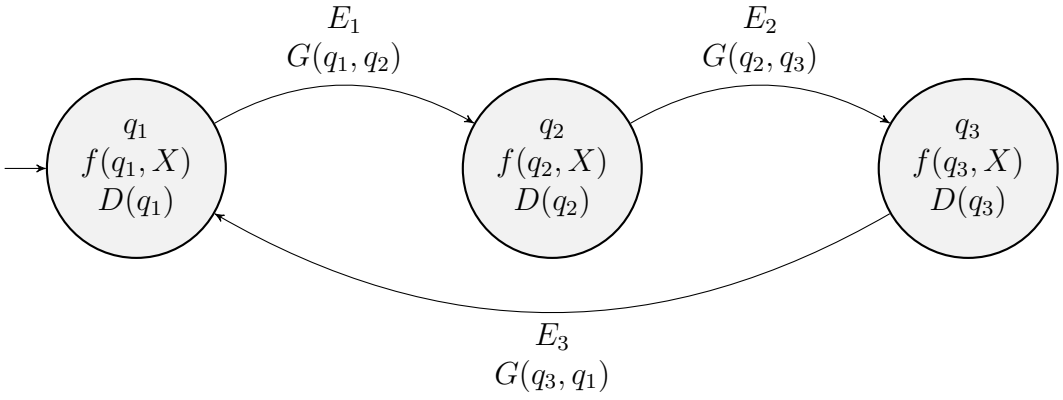


Figure 15: Translational(q_2), rotational(q_1) and the end position(q_3). Starts in q_1 and ends in q_3. Goes back to q_1 if end position changes.

Task 17

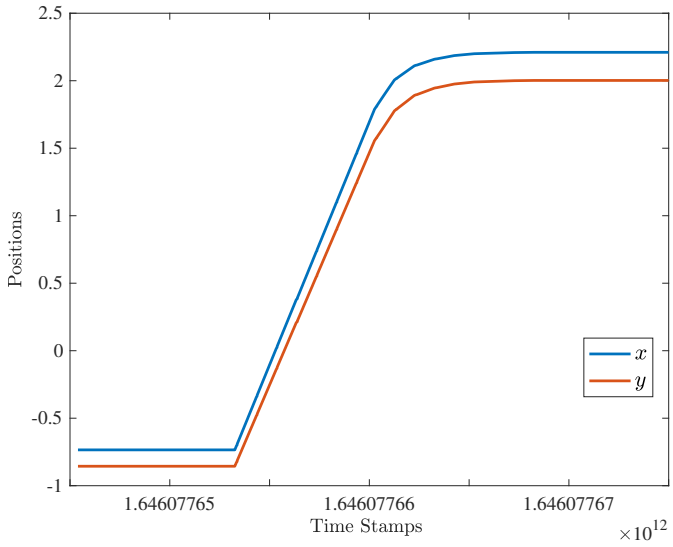
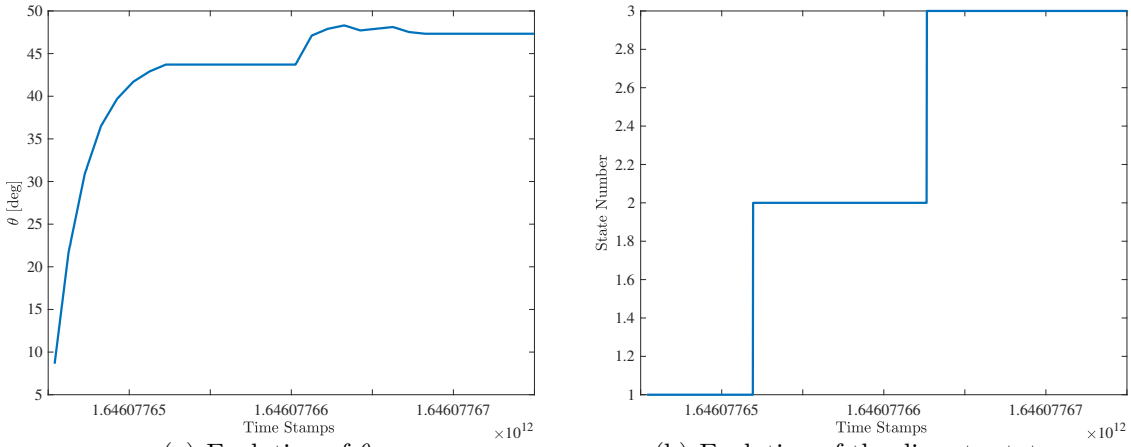


Figure 16: Evolution of the position from node 7 to node 3.



(a) Evolution of θ.

(b) Evolution of the discrete states.

Figure 17: Evolution from node 7 to 3.

Task 18

The path proposed in Task 3 is dictated by the following waypoints

$$[-1.25; 1.25; -0.75; 1.25]$$
$$[-0.75; 1.25; -0.25; 1.25]$$
$$[-0.25; 1.25; -0.25; 0.75]$$
$$[-0.25; 0.75; 0.25; 0.75]$$
$$[0.25; 0.75; 0.25; 0.25]$$

The robot follows the desired trajectory as shown in Fig.18. The controller can perform as designed because the robot can pass through all the way-points.

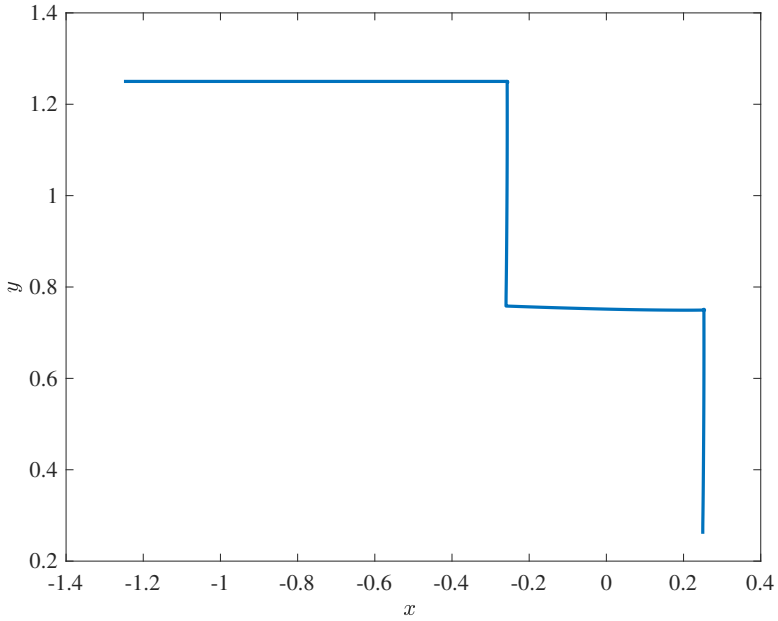


Figure 18: Trajectory of the robot in the xy-plane.

Task 19

The hybrid controller ensures the robot remains between the regions. Firstly, the controller corrects the orientation and later it heads straight to the desired point. This performance ensures the transition of the robot between two regions is well performed.