

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

|                |                        |               |
|----------------|------------------------|---------------|
| Rene Garcia    | First name2 Last name2 | Joel Aggefors |
| 20010124-5512  | person number          | 20030301-4575 |
| reneogt@kth.se | email                  | joelag@kth.se |

|               |                        |
|---------------|------------------------|
| Jenis Jain    | First name5 Last name5 |
| 20030424-T490 | person number          |
| jjjain@kth.se | email                  |

February 24, 2026

## Instructions and Help

**Please remove this part and the sample references before submitting your homework.**

Read the general homework instructions available on the course homepage before starting to write the report.

Here are some additional guidelines how to write a homework report.

- Fill in name and personal number of all group members.
- Each group uploads one file on Canvas within the deadline specified on the **Homework** page.
- This report should contain solutions to the first 19 tasks in Homework 3.
- This file needs to be one zip file with name `name1-name2-name3-name4-name5-HW3.zip`, where `name1, ..., name4, name5` are the last names of the authors. The zip file should contain 4 files: one pdf file named `name1-name2-name3-name4-name5-HW3.pdf` and three controller files `OwnVariables.c`, `Controller.c` and `RenewControllerState.c`.
- Do not copy the task descriptions and use the structure below.
- Do not include code unless the task explicitly states so.
- Motivate your answers well and how you derived them, but be concise.
- The number of points is not necessarily related to much you need to write for task.
- Put references in the end if any.
- Do not include plots from the Simulink scope (color on black background) but export the data to Matlab for plotting.

- Include graphics directly in the text and not in a Figure environment, as you normally would. That makes it easier to correct the report.
- There is plenty of material available how to use Latex. Use a search engine of your choice to learn more.

Here are some examples how to use Latex:

- An equation with a reference (1) to it

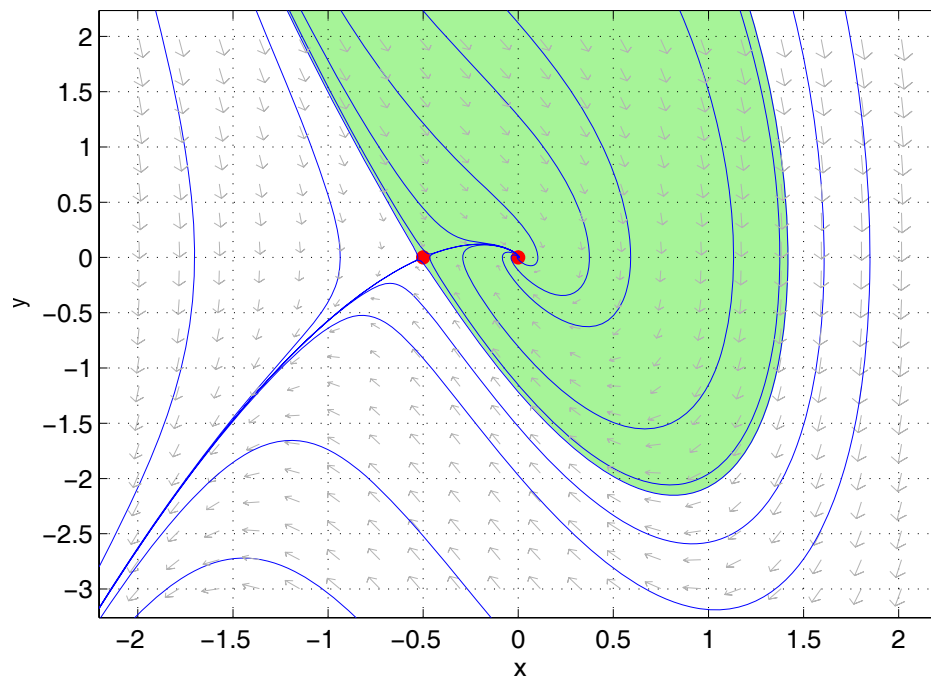
$$\dot{x} = \frac{3}{4}x. \tag{1}$$

- A multi-line equations with a reference to it

$$\begin{aligned}\hat{x} &= x - y \\ \alpha &= x + \gamma.\end{aligned}$$

- An equation in text:  $\Phi = \int_0^h e^{A\tau} d\tau$ .

- An image



- A table

|       |        |       |        |
|-------|--------|-------|--------|
| -2.46 | 0      | -1.73 | 0      |
| 0     | -2.553 | 0     | 2.774  |
| 0     | 6.172  | -10   | 7.333  |
| 1.767 | -0.357 | 5.714 | -6.074 |

- A citation [2]
- Display something exactly as it is written: `\frac{1}{2}_`
- Basic formatting: **bold**, *italics*, typewriter

## Task 1: Compute $u_r$ and $u_l$ from $(v, \omega)$

The robot inputs are defined as :

$$v = \frac{u_r + u_l}{2}, \quad \omega = u_r - u_l. \quad (2)$$

From (2), multiply the first equation by 2:

$$2v = u_r + u_l. \quad (3)$$

Now add (3) and the second equation in (2):

$$2v + \omega = (u_r + u_l) + (u_r - u_l) = 2u_r \Rightarrow u_r = v + \frac{\omega}{2}. \quad (4)$$

Similarly, subtract the second equation in (2) from (3):

$$2v - \omega = (u_r + u_l) - (u_r - u_l) = 2u_l \Rightarrow u_l = v - \frac{\omega}{2}. \quad (5)$$

Therefore, the wheel speeds corresponding to  $(v, \omega)$  are:

$$\boxed{u_r = v + \frac{\omega}{2} \quad [1^\circ/s], \quad u_l = v - \frac{\omega}{2} \quad [1^\circ/s]} \quad (6)$$

## Task 2

|              |                 |               |               |                 |                   |
|--------------|-----------------|---------------|---------------|-----------------|-------------------|
| $R_6$<br>G,S | $R_7$<br>B      | $R_{18}$<br>- | $R_{19}$<br>O | $R_{30}$<br>-   | $R_{31}$<br>R,G,O |
| $R_5$<br>-   | $R_8$<br>R,O    | $R_{17}$<br>- | $R_{20}$<br>R | $R_{29}$<br>B,O | $R_{32}$<br>R     |
| $R_4$<br>-   | $R_9$<br>G      | $R_{16}$<br>- | $R_{21}$<br>B | $R_{28}$<br>-   | $R_{33}$<br>-     |
| $R_3$<br>O   | $R_{10}$<br>O   | $R_{15}$<br>- | $R_{22}$<br>O | $R_{27}$<br>-   | $R_{34}$<br>-     |
| $R_2$<br>G   | $R_{11}$<br>R,O | $R_{14}$<br>- | $R_{23}$<br>B | $R_{26}$<br>R,O | $R_{35}$<br>-     |
| $R_1$<br>R   | $R_{12}$<br>-   | $R_{13}$<br>- | $R_{24}$<br>- | $R_{25}$<br>G   | $R_{36}$<br>B     |

Figure 1: Workspace discretization with  $K = 36$  regions. Labels indicate where atomic propositions hold.

### (a) Discretization, numbering , and atomic propositions

**Workspace:**  $[-1.5, 1.5] \times [-1.5, 1.5]$  (meters). A discretization with  $K = 36$  gives a  $6 \times 6$  grid.

regions are numbered in a *column-wise snake* pattern: start at the bottom-left with  $R_1$ , go *up* in the first column, then *down* in the next column, and so on. Thus, the  $6 \times 6$  numbering is:

|       |          |          |          |          |          |
|-------|----------|----------|----------|----------|----------|
| $R_6$ | $R_7$    | $R_{18}$ | $R_{19}$ | $R_{30}$ | $R_{31}$ |
| $R_5$ | $R_8$    | $R_{17}$ | $R_{20}$ | $R_{29}$ | $R_{32}$ |
| $R_4$ | $R_9$    | $R_{16}$ | $R_{21}$ | $R_{28}$ | $R_{33}$ |
| $R_3$ | $R_{10}$ | $R_{15}$ | $R_{22}$ | $R_{27}$ | $R_{34}$ |
| $R_2$ | $R_{11}$ | $R_{14}$ | $R_{23}$ | $R_{26}$ | $R_{35}$ |
| $R_1$ | $R_{12}$ | $R_{13}$ | $R_{24}$ | $R_{25}$ | $R_{36}$ |

**Atomic propositions:** Let  $AP = \{\text{red, blue, green, obstacle}\}$ . The propositions are specified by the following positions (meters):

- **obstacle** centers (spheres of radius 0.05 m) at:

$$(-0.75, 0.75), (0.25, 1.25), (0.75, 0.75), (0.25, -0.25), (0.75, -0.75), (-0.75, -0.75), (-0.75, -0.25)$$

- **red** holds at:

$$(-0.75, 0.7), (0.2, 0.7), (1.25, 1.25), (1.2, 0.8), (0.8, -0.8), (-0.9, -0.8), (-1.25, -1.25).$$

- **blue** holds at:

$$(-0.75, 1.4), (0.9, 0.9), (0.3, 0.2), (0.25, -0.75), (1.2, -1.4).$$

- **green** holds at:

$$(-1.23, 1.25), (1.25, 1.25), (-0.9, 0.2), (-1.2, -0.7), (0.6, -1.2).$$

**Labeling rule:** a proposition is true in the region that contains its given position, i.e.

$$p \in L(R_i) \iff (x_p, y_p) \in R_i, \quad p \in AP.$$

The start position is  $(-1.25, 1.25)$ , hence  $S_0 = \{R_6\}$ .

## (b) Cell size $dx, dy$

Since the side length is 3 m and there are 6 cells per side,

$$dx = \frac{3}{6} = 0.5 \text{ m}, \quad dy = \frac{3}{6} = 0.5 \text{ m}.$$

## (c) Comment on the choice $K = 36$

A  $6 \times 6$  grid provides a finer abstraction than coarse grids (better separation of obstacles/colored areas), but increases the number of states and transitions compared to smaller  $K$ .

(d) **Transition system**  $T = (S, S_0, \Sigma, \rightarrow, AP, L)$

$S = \{R_1, \dots, R_{36}\}$ ,  $S_0 = \{R_6\}$ ,  $\Sigma = \{\text{Up, Down, Left, Right}\}$ ,  $AP = \{\text{red, blue, green, obstacle}\}$ .

The transition relation  $\rightarrow$  is the 4-neighborhood relation on the grid:

$$R_i \xrightarrow{\sigma} R_j \iff R_j \text{ is the adjacent region to } R_i \text{ in direction } \sigma \in \Sigma.$$

The labeling function  $L : S \rightarrow 2^{AP}$  is defined using the point-in-region rule above.

### Q3: Find an infinite path satisfying the specification

**Specification:** (i) visit **red** infinitely often, (ii) whenever the robot is in a **red** region, the *next* region is **blue**, (iii) never enter a region labeled **obstacle**.

**Chosen red–blue pair:** From the labeling in Q2,  $R_{20} \in \text{red}$  and  $R_{21} \in \text{blue}$ , and they are adjacent in the  $6 \times 6$  grid ( $R_{21}$  is directly below  $R_{20}$ ). Moreover,  $R_{20}, R_{21} \notin \text{obstacle}$ .

**A valid infinite path:** Starting from  $S_0 = \{R_6\}$ , one feasible prefix to reach  $R_{20}$  without entering obstacles is

$$R_6 \rightarrow R_7 \rightarrow R_{18} \rightarrow R_{17} \rightarrow R_{16} \rightarrow R_{21} \rightarrow R_{20}.$$

Then, repeat the 2-cycle  $(R_{20}, R_{21})$  forever:

$$\pi = \underbrace{(R_6, R_7, R_{18}, R_{17}, R_{16}, R_{21}, R_{20})}_{\text{prefix}} \cdot \underbrace{(R_{21}, R_{20})^\omega}_{\text{suffix repeated forever}}.$$

**Why  $\pi$  satisfies the specification:**

- $\pi$  visits  $R_{20}$  infinitely often, and  $R_{20} \in \text{red}$ , hence **red** is visited infinitely often.
- Every time  $\pi$  is in  $R_{20}$  (a **red** region), the next state is  $R_{21}$  and  $R_{21} \in \text{blue}$ , so the “after **red** next is **blue**” condition holds.
- All regions used in  $\pi$  are chosen outside the set of **obstacle** regions, hence obstacles are never entered.

### Q4

The hybrid strategy prevents entering unintended regions by separating the motion into two simple phases. First, the robot uses a rotation mode with (approximately) zero forward speed, i.e.,  $v \approx 0$ , so it turns in place to align its heading with the straight line connecting the center of the current region to the center of the target (neighbor) region. Because the robot does not translate during this phase, it stays close to the current region center and does not drift into adjacent regions. Second, the robot switches to a line-following mode and drives forward while tracking that same center-to-center line. For neighboring regions, this line segment lies within the union of the two adjacent cells. Therefore, if the tracking error is kept small, the robot remains inside only the current and target regions during the transition, and it avoids passing through any other region that could contain an obstacle.

## Q5

During the rotation mode, the controller is

$$\omega[k] = K_{\Psi,1}(\theta_R - \theta[k]). \quad (7)$$

The robot yaw dynamics satisfy

$$\dot{\theta}(t) = \frac{R}{L} \omega(t) \quad [1^\circ/s]. \quad (8)$$

Using forward Euler discretization with sampling time  $h$ ,

$$\theta[k+1] = \theta[k] + h \frac{R}{L} \omega[k]. \quad (9)$$

Substituting (7) into (9) and defining the error  $e[k] = \theta_R - \theta[k]$ , we obtain

$$e[k+1] = \left(1 - \frac{hR}{L} K_{\Psi,1}\right) e[k]. \quad (10)$$

For asymptotic stability of the discrete-time error dynamics, we require

$$\left|1 - \frac{hR}{L} K_{\Psi,1}\right| < 1 \iff 0 < \frac{hR}{L} K_{\Psi,1} < 2 \iff \boxed{0 < K_{\Psi,1} < \frac{2L}{hR}}. \quad (11)$$

A practical choice is to pick  $K_{\Psi,1}$  well inside this interval (e.g.,  $K_{\Psi,1} = \alpha \frac{L}{hR}$  with  $\alpha \in (0, 1)$ ) to avoid oscillations and actuator saturation.

## Task 6

First we define the proportional constant value. Given eq. (11), we could aim for a deadbeat controller by setting  $K_{\Psi,1} = \frac{2L}{hR}$ , nonetheless, this would force the controller to correct the error in one sampling step, completely ignoring physical limits and inertia. A safer value to chose is the middle of the stability range:

$$K_{\Psi,1} = \frac{1}{2} \frac{2L}{hR} = \frac{L}{hR} = \frac{0.16}{0.033 * 1} = 4.85 \quad [1/s]. \quad (12)$$

## Task 7

Assume  $\theta[k+1] = \theta[k] = \theta$  so  $v_c = [\cos \theta \quad \sin \theta]^\top$  is constant and  $v_c^\top v_c = 1$ . Using  $\dot{p} = Rvv_c$  with  $p = [x \quad y]^\top$  and forward Euler,

$$p[k+1] = p[k] + hRv[k]v_c \Rightarrow \Delta_0[k+1] = \Delta_0[k] - hRv[k]v_c.$$

Premultiplying by  $v_c^\top$  yields

$$d_0[k+1] = v_c^\top \Delta_0[k+1] = d_0[k] - hRv[k].$$

With  $v[k] = K_{\omega,1}d_0[k]$ ,

$$d_0[k+1] = (1 - hRK_{\omega,1})d_0[k].$$

Asymptotic stability requires  $|1 - hRK_{\omega,1}| < 1$ , hence

$$\boxed{0 < K_{\omega,1} < \frac{2}{hR}}.$$

## Task 8

Solution to the task

## Task 9

Solution to the task

## Task 10

Assume  $\theta[k] = \theta_g$ , hence  $v_g = [\cos \theta_g \quad \sin \theta_g]^\top$  is constant and  $v_g^\top v_g = 1$ . With  $\dot{p} = R v v_g$  and forward Euler,

$$p[k+1] = p[k] + h R v[k] v_g \Rightarrow \Delta_g[k+1] = \Delta_g[k] - h R v[k] v_g.$$

Premultiplying by  $v_g^\top$  yields

$$d_g[k+1] = v_g^\top \Delta_g[k+1] = d_g[k] - h R v[k].$$

Using  $v[k] = K_{\omega,2} d_g[k]$ ,

$$d_g[k+1] = (1 - h R K_{\omega,2}) d_g[k].$$

Asymptotic stability requires  $|1 - h R K_{\omega,2}| < 1$ , hence

$$\boxed{0 < K_{\omega,2} < \frac{2}{h R}}.$$

A practical choice is to select  $K_{\omega,2}$  well inside this interval (e.g.,  $K_{\omega,2} = \alpha \frac{1}{h R}$  with  $\alpha \in (0, 1)$ ) to ensure monotone convergence and robustness to discretization/actuator limits.

## Task 11

Solution to the task

## Task 12

The controller for line-following (part II) is

$$\omega[k] = K_{\Psi,2} d_p[k].$$

Assume the robot is on the line from  $(x_0, y_0)$  to  $(x_g, y_g)$  and  $\theta$  is close to  $\theta_g$  so that

$$d_p[k] \approx p(\theta_g - \theta[k]), \quad p > 0.$$

Let  $e[k] = \theta_g - \theta[k]$ , hence  $d_p[k] \approx p e[k]$ . From the yaw dynamics  $\dot{\theta} = \frac{R}{L} \omega$  and forward Euler with sampling time  $h$ ,

$$\theta[k+1] = \theta[k] + h \frac{R}{L} \omega[k].$$

Therefore,

$$e[k+1] = \theta_g - \theta[k+1] = e[k] - h \frac{R}{L} \omega[k] = e[k] - h \frac{R}{L} K_{\Psi,2} d_p[k] \approx \left(1 - h \frac{R}{L} K_{\Psi,2} p\right) e[k].$$

Multiplying by  $p$  gives the discrete-time dynamics of  $d_p$ :

$$d_p[k+1] \approx \left(1 - h \frac{R}{L} K_{\Psi,2} p\right) d_p[k].$$

Thus  $d_p[k]$  is asymptotically stabilized in 0 iff

$$\left|1 - h \frac{R}{L} K_{\Psi,2} p\right| < 1 \iff \boxed{0 < K_{\Psi,2} < \frac{2L}{hRp}}.$$

**Choice of  $K_{\Psi,2}$ .** Select  $K_{\Psi,2}$  strictly inside the stability interval, e.g.

$$K_{\Psi,2} = \alpha \frac{L}{hRp}, \quad \alpha \in (0, 1),$$

which yields the contraction factor  $1 - \alpha$  (monotone convergence) and keeps the closed-loop behavior consistent when the sampling time  $h$  changes.

## Task 13

Solution to the task

## Task 14

Solution to the task

## Task 15

Solution to the task

## Task 16 : Hybrid automaton

Define the hybrid automaton  $H = (Q, X, Init, f, D, E, G, R)$ .

**Discrete states.**

$$Q = \{q_{\text{rot}}, q_{\text{line}}\},$$

where  $q_{\text{rot}}$  aligns the robot with the goal direction and  $q_{\text{line}}$  performs line-following / go-to-goal.

**Continuous state and initialization.**

$$X = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}, \quad Init = \left\{ (q_{\text{rot}}, X) \mid X = \begin{bmatrix} x_s \\ y_s \\ \theta_s \end{bmatrix} \right\}.$$



**Continuous dynamics (closed-loop).** Robot kinematics:

$$\dot{x} = R v \cos \theta, \quad \dot{y} = R v \sin \theta, \quad \dot{\theta} = \frac{R}{L} \omega.$$

Let  $\theta_R = \text{atan2}(y_g - y, x_g - x)$ .

Mode  $q_{\text{rot}}$ :

$$\omega = K_{\Psi,1}(\theta_R - \theta), \quad v = K_{\omega,1}d_0.$$

Mode  $q_{\text{line}}$ :

$$v = K_{\omega,2}d_g, \quad \omega = K_{\Psi,2}d_p.$$

**Domains.**

$$D(q_{\text{rot}}) = D(q_{\text{line}}) = \mathbb{R}^2 \times (-180^\circ, 180^\circ].$$

**Edges and guards.**

$$E = \{(q_{\text{rot}}, q_{\text{line}}), (q_{\text{line}}, q_{\text{rot}})\}.$$

Using thresholds  $\varepsilon_\theta > 0$  and  $\varepsilon_g > 0$ :

$$G(q_{\text{rot}}, q_{\text{line}}) = \{X : |\theta_R - \theta| \leq \varepsilon_\theta\}, \quad G(q_{\text{line}}, q_{\text{rot}}) = \{X : \|(x_g - x, y_g - y)\| \leq \varepsilon_g\}.$$

**Resets.** No state jump at switching (identity reset):

$$R(q_{\text{rot}}, q_{\text{line}}) : X^+ = X, \quad R(q_{\text{line}}, q_{\text{rot}}) : X^+ = X.$$

(If multiple waypoints are used, the next goal  $(x_g, y_g)$  is updated externally when  $\varepsilon_g$  is reached.)

## Task 17

Solution to the task

## Task 18

Solution to the task

## Task 19

Solution to the task

## References

- [1] Hassan K Khalil. *Nonlinear systems*. Prentice Hall, Upper Saddle river, 3. edition, 2002. ISBN 0-13-067389-7.
- [2] Tobias Oetiker, Hubert Partl, Irene Hyna, and Elisabeth Schlegl. *The Not So Short Introduction to  $\LaTeX 2_\varepsilon$* . Oetiker, OETIKER+PARTNER AG, Aarweg 15, 4600 Olten, Switzerland, 2008. <http://www.ctan.org/info/lshort/>.
- [3] Shankar Sastry. *Nonlinear systems: analysis, stability, and control*, volume 10. Springer, New York, N.Y., 1999. ISBN 0-387-98513-1.