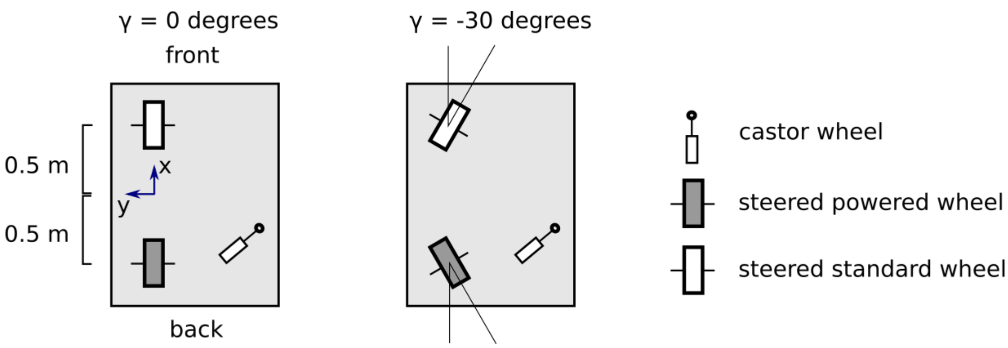
Robotics exercises for kinematics and control (solved)

March 11, 2025

> Answer the questions in the spaces provided on the question sheets. If you run out of room for an answer, continue on the back of the page or on the provided extra pages.
> Write your name in the header of all pages. The handed-in material must be filled in pen, not pencil.
> **Clarity is part of the evaluation criteria. If you have doubts about the details of a question, make an assumption that you believe is convenient and reasonable and state it clearly. The exam is evaluated by a human being, not a robot.**

Name, surname: ______________________________

1 Exercise (35/100 points)



Consider the 2D wheeled robot in the figure, where you control:

- the rotational speed of the back wheel $\dot{\varphi}_{\text{back}}$; the radius of the back wheel is 10 cm.
- the steering angle of both wheels at the same time. The wheel steering mechanisms are connected, so the two wheels steer in opposite directions, with a single controllable parameter γ representing the depicted angle. For $\gamma = 0$, both wheels are aligned with the x direction of the robot frame, and a positive value for $\dot{\varphi}_{\text{back}}$ will move the robot forwards. For $\gamma = +10\,\text{deg}$, the front wheel steers 10 degrees counterclockwise, and the back wheel steers 10 degrees clockwise.

1. (5 points) Assume we place the robot on an infinite floor without obstacles.

   If we limit $-30\,\text{deg} < \gamma < +30\,\text{deg}$: is the robot an holonomic robot? can the robot reach any pose (x, y, θ) on the plane?

   If we limit $+20\,\text{deg} < \gamma < +30\,\text{deg}$: is the robot an holonomic robot? can the robot reach any pose (x, y, θ) on the plane?

Solution: In both cases, the robot is not holonomic; for example, it can never access the velocity in C-space corresponding to a pure translation along y.

In both cases, the robot can access any pose on the plane. In the first case, it's easy to see even though some maneuvering would be necessary (just like a car does). In the second case, you can decompose any translation along the robot's y axis as a sequence of an even number of 180 degree turns with different curvature radii. Similarly, one can achieve any x translation as a sequence of several triplets of turns: 1) turning 90 degrees with a given curvature radius $gamma'$, 2) turning 180 degrees with a different curvature radius, 3) turning 90 degrees again with the first curvature radius $gamma'$. This is allows us to translate the robot to any point in the plane.

2. (5 points) Briefly comment on the kinematics of the robot for a steering angle $\gamma = 90\,\mathrm{deg}$

Solution: In that case, the robot's kinematics is underconstrained; like in a differential drive robot, the axes of the two wheels coincide, so the ICR could lie at any point along the line. However, because one wheel is not powered, we can not control where the ICR is, because we have no control on how the unpowered wheel spins.

3. (15 points) From this point on, assume $-90\,\mathrm{deg} < \gamma < +90\,\mathrm{deg}$.

   Discuss how you could implement a function $f(\gamma, \dot{\varphi}_{\mathrm{back}}, \delta t)$ that returns a 3×3 homogeneous matrix representing the pose transformation that occurs when the robot moves for a time interval δt with a fixed rotational speed for the back wheel $\dot{\varphi}_{\mathrm{back}}$ and a fixed steering angle γ.

Solution: The tangential speed of the back wheel is (in meters per second):

$$v_{\text{back}} = \dot{\varphi}_{\text{back}} \cdot 0.1$$

If $\gamma \neq 0$, the ICR lies at a distance R along the robot's y axis:

$$R = 0.5/\tan(\gamma)$$

(negative if the ICR is to the right of the robot)
Then, the angular velocity of the robot is found as:

$$\omega = \frac{v_{\text{back}}}{\sqrt{R^2 + 0.5^2}}$$

or equivalently

$$\omega = \frac{v_{\text{back}} \sin\gamma}{0.5}$$

Then our function can be defined as follows.

$$f(\gamma, \dot{\varphi}_{\text{back}}, \delta t) = \begin{cases} \begin{bmatrix} 1 & 0 & v_{\text{back}}\delta t \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} & \text{if } \gamma = 0 \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & R \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\omega\delta t) & -\sin(\omega\delta t) & 0 \\ \sin(\omega\delta t) & \cos(\omega\delta t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -R \\ 0 & 0 & 1 \end{bmatrix} & \text{otherwise} \end{cases}$$

4. (5 points) From this point on, assume the function $f(\gamma, \dot{\varphi}_{\text{back}}, \delta t)$ is available and correct, so if you screwed up or didn't answer the previous question, that won't affect you.

   Consider the world reference frame O. The robot is initially placed at the origin, with the robot's x axis pointing towards the y axis of frame O. Relying on function $f(\cdots)$, write the expressions returning the pose of the robot ${}^{O}T_R$ in the world frame of reference O at each of the following times: $t = 0, 1, 5, 7$. Starting at time $t = 0$, the robot moves as follows:

   - for 2 seconds,
     $\dot{\varphi}_{\text{back}} = +10$ rad/s,
     $\gamma = +30\,\text{deg}$.
   - for 5 seconds,
     $\dot{\varphi}_{\text{back}} = -10$ rad/s,
     $\gamma = -45\,\text{deg}$.

Solution: The initial pose is given by $I = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

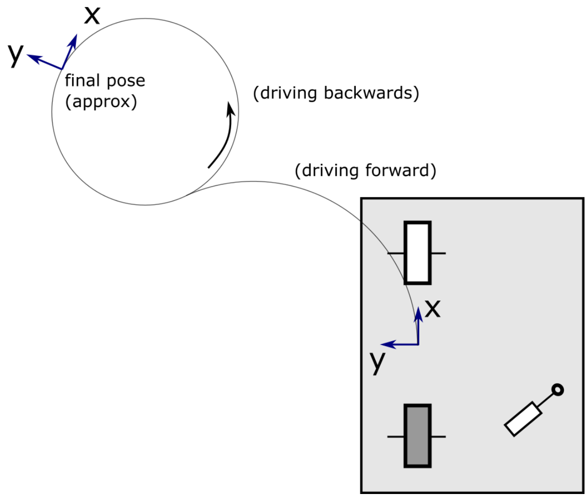
The pose at $t = 1$ second is $I \cdot f(+30\,\text{deg}, +10\text{rad/s}, 1\text{s})$.

The pose at $t = 5$ seconds is $I \cdot f(+30\,\text{deg}, +10\text{rad/s}, 2\text{s}) \cdot f(-45\,\text{deg}, -10\text{rad/s}, 3\text{s})$.

The pose at $t = 7$ seconds is $I \cdot f(+30\,\text{deg}, +10\text{rad/s}, 2\text{s}) \cdot f(-45\,\text{deg}, -10\text{rad/s}, 5\text{s})$.

5. (5 points) Approximately draw the trajectory the robot followed and its final pose

Solution:



6. (Extra points) rewrite function $f(\cdots)$ in case that the robot reference frame was at the center of the robot chassis (the gray rectangle). Make the necessary reasonable assumptions.

Solution: Let K be the pose transform between the new reference frame (placed at the center of the robot chassis) and the reference frame we used in order to define $f(\cdots)$. For example, if the new frame is 20 cm to the right of the old frame,

$$K = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0.2 \\ 0 & 0 & 1 \end{bmatrix}$$

Now, $f'(\cdots) = K f(\cdots) K^{-1}$

2 Exercise (20/100 points)

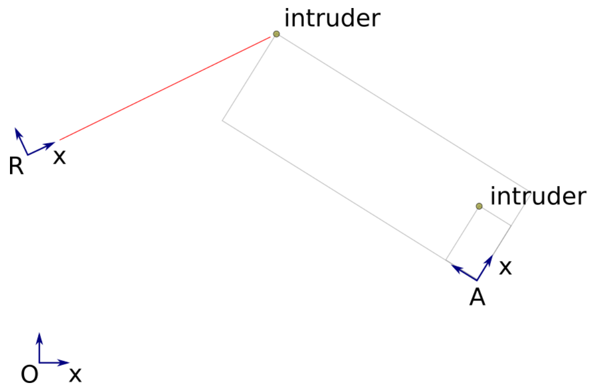
We consider a wheeled robot that explores a 2D world looking for intruders; we define a frame of reference R for such robot in such a way that x points forwards. The robot is equipped with a special laser sensor that points forward; with this sensor, the robot can detect whether there is an intruder in front of the robot, i.e. at any point lying on the positive half of the robot frame's x axis; however, the sensor can't measure the distance of the intruder, nor it can see any intruders that are even just a bit to the side.

Let ${}^{O}T_{R}$ be a 3×3 homogeneous transformation matrix representing the pose of the robot with respect to the world frame O.

The robot's sensor is detecting an intruder. At the same time, a fixed radar sensor has detected *two* potential intruders. Let (x_1, y_1) and (x_2, y_2) be their positions expressed in the radar frame A. Let ${}^{O}T_{A}$ be a 3×3 homogeneous transformation matrix representing the 2D pose of the radar frame with respect to the world frame.

1. (3 points) Draw the scenario and label the frames.

Solution:



2. (10 points) We want to determine which of the two potential intruders detected by the radar was detected by the robot. Describe how you would implement a function to do that.

Solution: Assume that ${}^{A}P$ is the position of an intruder expressed in the radar frame A. First, I want to compute ${}^{R}P$.

$$ {}^{R}P = {}^{R}T_{A}{}^{A}P = {}^{R}T_{O}{}^{O}T_{A}{}^{A}P = ({}^{O}T_{R})^{-1}{}^{O}T_{A}{}^{A}P $$

Let x, y be the coordinates of P in R's frame; P is seen by the sensor on R iff $x > 0$ and $y = 0$. We expect that this condition will be (almost) true for at least one of the two intruders. If it is true for both, we don't know who was actually seen. Depending on the sensor, it might be reasonable to think it is the closest (i.e. the one with a smaller x).

3. (7 points) Assume we determine that the intruder position is at the following coordinates of the world frame: $(x, y) = (30, 40)$ meters.

   Let ${}^{O}T_{K}$ be the pose (3×3 matrix in homogeneous coordinates) of another intruder-detecting robot K. Describe how we can determine by which angle the robot K must turn in place in order to detect the intruder.

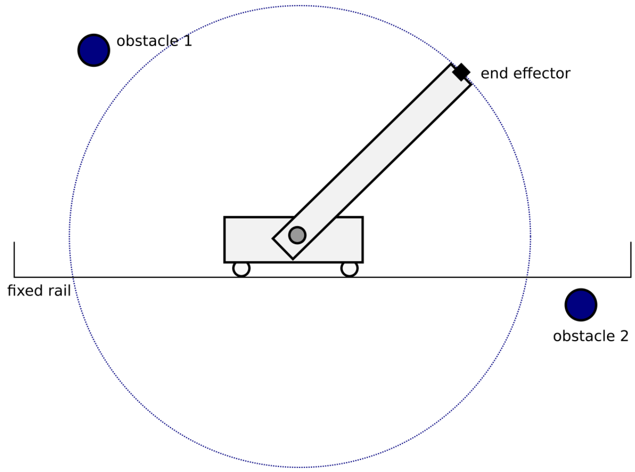
Solution: We now know that ${}^{O}P = (30, 40)$. As above, we first compute ${}^{K}P$.

$$ {}^{K}P = {}^{K}T_{O}{}^{O}P = ({}^{O}T_{K})^{-1}{}^{O}P $$

Let x, y be the coordinates of P in K's frame; we want to rotate the robot by $\arctan(y/x)$. In code, one would use the `arctan2(y,x)` function, which returns an angle in the range $[-180\,\text{deg}, +180\,\text{deg}]$, so would turn the robot either clockwise or counterclockwise, in such a way to cover the smallest absolute angle.

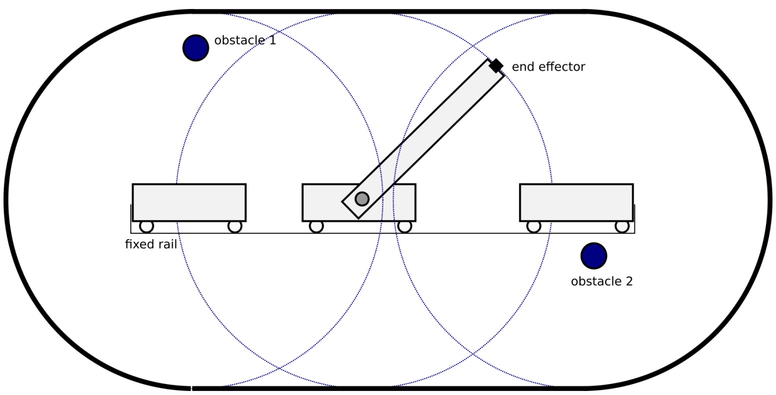
3 Exercise (23/100 points)

Consider the robot arm in the figure:



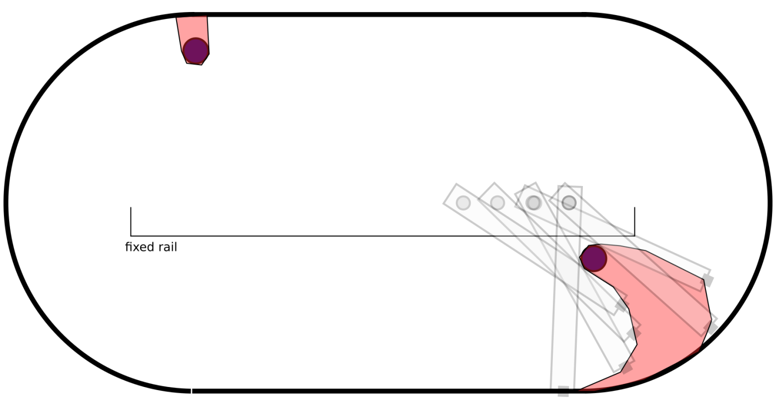
1. (2 points) Draw the workspace, ignoring the obstacles.

Solution:



2. (4 points) Mark the parts of the workspace not reachable because of the obstacles (no part of the robot can hit any of the obstacles).

Solution:



Note: figuring out the precise shape is very difficult without a computer; in an exam, a rough approximation would be sufficient. Still, make sure you understand *why* the shape looks like that.

3. (2 points) Consider the whole robot as a manipulator with one prismatic joint and one revolute joint. Define two reasonable variables that can describe the configuration of the manipulator.

> **Solution:** We can define for example
>
> - x as the horizontal position of the cart between 0 (full left) and L (full right); this corresponds to a prismatic joint; and
> - θ as the angle of the revolute joint, with 0 deg being horizontal to the right, between -180 deg (clockwise half turn) and $+180$ deg (counterclockwise half turn).

4. (3 points) Ignoring the obstacles, draw the configuration space and label the axes and their ranges appropriately. Discuss the topology of the configuration space.

> **Solution:** The topology is a cylinder, assuming that the revolute joint can turn continuously, because $\theta = +180$ deg is the same configuration as $\theta = -180$ deg.

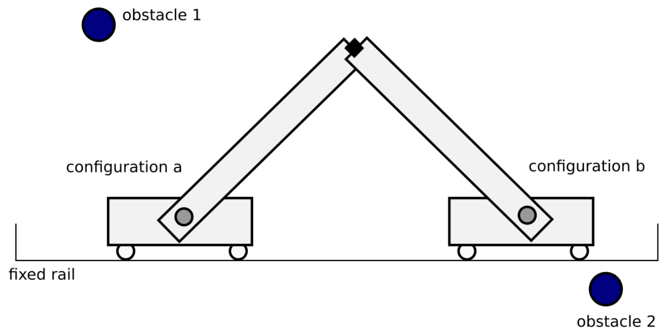
5. (7 points) Approximately draw the two obstacles in the configuration space.

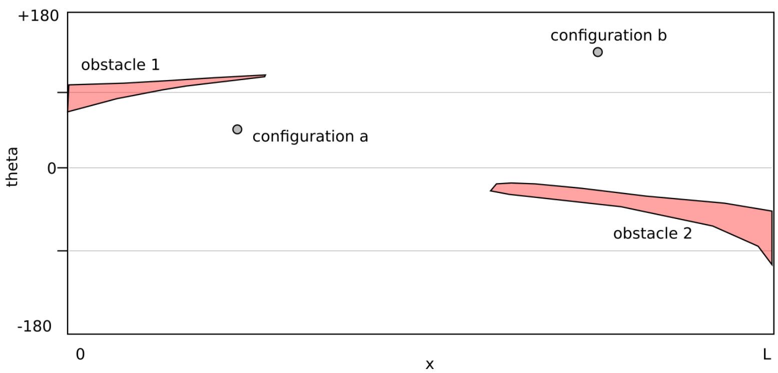
> **Solution:** see below

6. (3 points) Draw two different points on the configuration space for which the end-effector position is the same.

7. (2 points) Mark one of these points and draw the robot in that configuration.

> **Solution:**



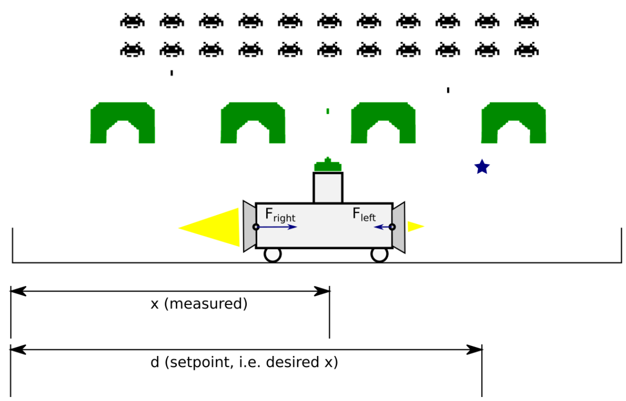


Note: the obstacles in the configuration space are probably wider (along x) than you expect (and than we drew in class). To understand why, consider the illustration in the very first figure of the exercise. How much can you move the cart left or right before the arm can't make a full turn anymore because of obstacles? Not much! This means that the obstacles are pretty wide in the C-space, and that the range of x for which the C-space is free of obstacles along for any θ is not very wide.

Also, note that from configuration a you can't do a full turn of the revolute joint without hitting obstacle 1.

4 Exercise (22/100 points)

The space-invaders tank can freely move on an horizontal rail with very little friction. Few people know that it is in fact actuated by two powerful thrusters, one pushing towards the right with a force $F_{\text{right}} \geq 0$, and one pushing towards the left with a force $F_{\text{left}} \geq 0$.



You have been asked to implement a software function to control this cart in a closed-loop way. The cart position along the rail is known as x. The task is to reach and stay at a position d, which is defined by the Great Commander. Your function is run 10 times per second and must produce a value for F_{left} and a value for F_{right}, which are instantaneously implemented by the thrusters.

1. (3 points) Your function has two inputs (x and d) and two outputs (F_{right} and F_{left}). Inside your function, you want to use one PID controller, which needs a single input (the position error) and produces a single output (the total force acting on the cart). Describe: how you compute the error from the function inputs, and; how you compute the function outputs from the PID output. Note that fuel is expensive and scarce in times of war, so it would be really stupid to fire both rockets at the same time.

Solution: The PID controller can be fed with single input (the position error $e = d - x$) and will produce a single output, i.e. the total resulting force F we want to enact towards the right (if positive) or left (if negative).

We can compute F_{left} and F_{right} from F as follows:

```
if F > 0:
  Fright = F
  Fleft = 0
else:
  Fright = 0
  Fleft = -F
```

2. (3 points) According to your definition of the error and output, elaborate on the sign of the proportional gain (K_p) of the controller.

Solution: According to the definitions above, we expect K_p to be positive. In fact, if $e = d - x$ is positive, it means the cart should move right, i.e. we want a force to the right (a positive output from our PID).

Note that when solving in class we defined a different input to the PID $(x - d)$, then K_p was negative.

3. (3 points) Do you expect that using a derivative gain $K_d \neq 0$ will be useful? Briefly motivate.

4. (3 points) Do you expect that using an integral gain $K_i \neq 0$ will be useful? Briefly motivate.

Solution: K_d is necessary. Otherwise, starting from the initial position depicted in the figure, the cart will continue accelerating to the right until it is at the setpoint; only then (and it will be moving quite fast) it will start braking; without friction, this will yield infinite oscillations. Using $K_d > 0$ will make sure that when e is decreasing fast, i.e. the robot is moving fast to the right, the F will be reduced and possibly become negative, i.e. the robot will start braking before it reaches the setpoint.

K_i is likely unnecessary; consider what happens when $K_i = 0$, and the robot is static at the setpoint; then e and its derivative are 0. The output force will be 0. This will keep the robot at the setpoint because there is no external force acting on it.

If the track was tilted (with gravity pushing the cart left or right), then K_i would be needed to exactly reach the setpoint.

5. (5 points) A colleague insists that there is a setting which would work very well using $K_p = K_i = 0$, and some magic value for K_d. Write a single sentence to convince them that they are wrong.

Solution: Dear colleague, if the robot is not moving, then F will be 0 whatever the value of K_d.

6. (5 points) Briefly explain why a bang-bang controller would not be a good idea in place of the PID controller.

Solution: Same argument as for why K_p alone is not enough. We need a derivative term to avoid oscillations.

7. (Extra points) Sketch a hierarchical control architecture with one PID controller to set a desired velocity and one PID controller to set the force.

Solution: One possible solution is to use two PID controllers. One (high-level) is fed with e and decides what velocity the cart should have; the other (low-level) tries to achieve such velocity by controlling the force.

The high-level controller takes e as input and outputs a desired velocity v_{des}. Optionally, one could then clamp v_{des} to have a maximum absolute value, if we don't want the cart to ever go too fast.

The low-level controller takes $v - v_{\text{des}}$ as input and outputs F. v can be computed as the derivative of x.

This hierarchical architecture is probably a bit easier to tune and easier to understand than the single PID. You could also show that if both controllers are pure proportional (i.e. $K_i = K_d = 0$ and v_{des} is not clamped, then this architecture is equivalent to a single PID.

8. (Extra points) It turns out we don't have enough memory to implement two PID controllers. We have to substitute one with a Bang-Bang controller. Which one do you choose and why?

Solution: We expect that the high-level controller can't be substituted with bang-bang controller without causing a lot of oscillations; instead, substituting the low-level PID with a bang-bang controller could work (but would waste a lot of fuel)

Extra page 1

Extra page 2