Hand-in Exercise on Kinematics and Dynamics

FRTN85 Applied Robotics
Faculty of Engineering LTH, Lund University

September 2025

- The deadline for the hand-in is **on October 24, 2025**.
- The hand-in exercise is performed individually.
- Upload your solutions in the assignments tab in Canvas.
- Submit your solution as one **.pdf** file. Scanned handwritten solutions are accepted, but please make sure that the solutions are readable in the **.pdf** file.
- The file has to contain your solutions with *clear motivations for the different steps.*
- Append any code you have used to the end of the **.pdf** file.

- Feel free to use any programming language (Matlab, Python, Julia, ...). Peter Corke's Robotics Toolbox is available both in Matlab and Python.
- You may find it useful with symbolic computations for some of the problems (Matlab, Mathematica, Maple, SymPy, ...).
- If you use any generative AI tools as part of solving the hand-in exercise, specify which and how you have used it.
- You are encouraged to ask questions about the exercises at the TA sessions. Also, feel free to discuss the problems on a general level with your colleagues, but you should present **your own** solutions and code in the submission in Canvas.

1 Coordinate Transformations

Consider two coordinate frames, $O_0 - x_0y_0z_0$ and $O_1 - x_1y_1z_1$. The two frames are related as defined in Figure 1, O_1 is obtained by rotating O_0 about the y-axis by an angle α. You may assume that $\alpha \in [0, \pi/2]$.

(a) Transform the three vectors from the frame O_0 given by

$$\mathbf{e_1} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{e_2} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \quad \mathbf{e_3} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

to the frame O_1.

(b) Find a 3×3 rotation matrix that transforms vectors from O_0 to vectors in O_1.

(c) Let $\alpha = \pi/4$. Apply the matrix from (b) to the vectors $\mathbf{e}_i$ from (a) to verify that you got the correct solution.

(d) Suppose that, in addition to rotation, we also wanted O_1 to be translated in space from O_0 by 1 units along the y-axis. Could this also be done by a 3×3 matrix? Find such a matrix, or explain why this cannot be done and suggest how such a transform can be represented instead.

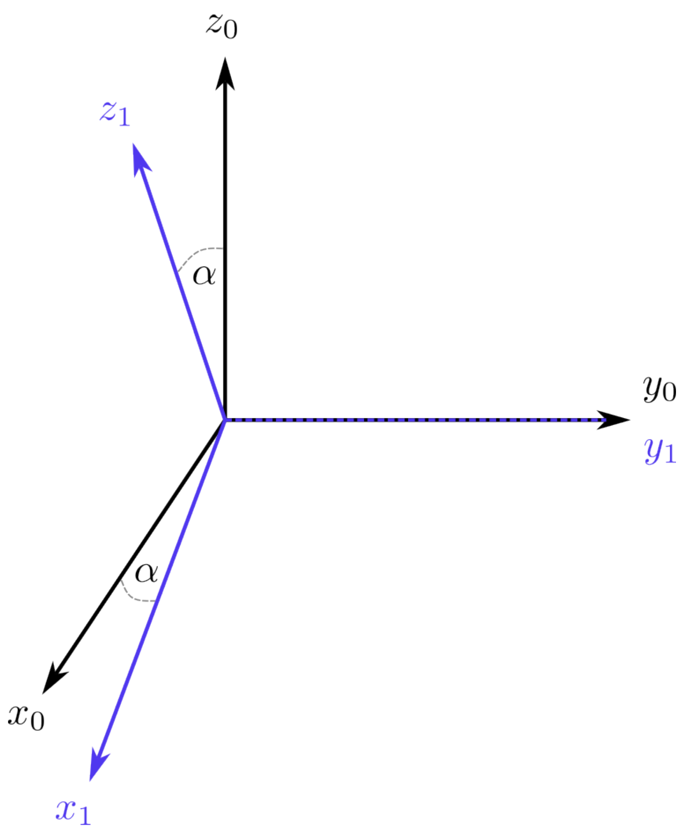

Figure 1: Rotating a coordinate frame about the y-axis with an angle α.

2 Denavit-Hartenberg Parameters

Consider the coordinate frames shown in Figure 2, corresponding to a two-link robot. The z-axis is always at the intersection of the x- and y-axes, with its orientation shown by the small circle next to the intersection.

(a) Determine the DH parameters for the coordinate frames.

(b) Determine the homogeneous transformation matrices $\mathbf{T}_1^0$, $\mathbf{T}_1^2$, and $\mathbf{T}_2^0$.

(c) Explain the difference between a homogeneous transformation matrix and a rotation matrix (like the one from Problem 1).

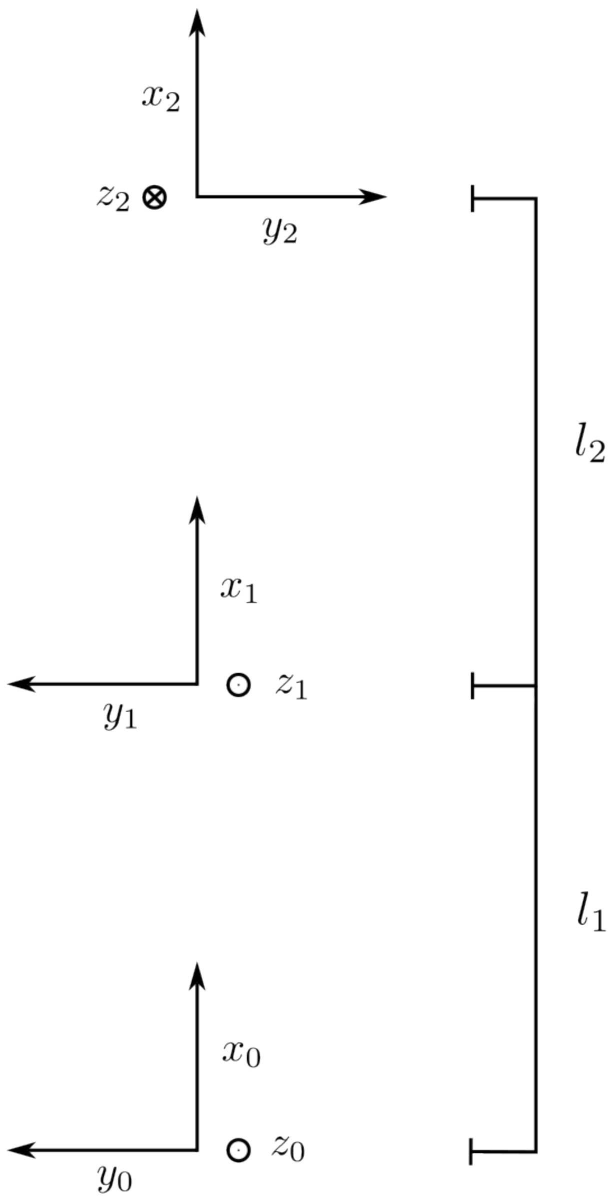

Figure 2: Setup with a two-link robot and associated coordinate frames.

3 Forward and Inverse Kinematics

In Figure 3 you can see a planar robot with three links. Each of the links of length l_i are attached to a motor which allows their base to rotate by an angle θ_i. In this problem you will define and discuss the kinematics of this robot with revolute joints. For simplicity, let $l_i = 2$ for all three links. Put the base coordinate frame down at the base of the robot (where joint 1 is located).

(a) Find the forward kinematics equations, i.e., a function which takes $(\theta_1, \theta_2, \theta_3)$ as input and computes the homogeneous transformation matrix of the end-effector / tool frame with respect to the base frame.

(b) To check your solution, compute the forward kinematics for the following joint configurations $(\theta_1,\theta_2,\theta_3)$: $(0, 0, 0), (0, \pi/4, \pi/4)$. Are the resulting homogeneous transformation matrices in agreement with what you would expect? Draw the robot configuration in the respective case and mark the end-effector / tool frame.

(c) How many solutions does the inverse kinematics problem have, given a specified coordinate (x, y) for the robot end-effector / tool? Can we find a closed-form function for the inverse kinematics (you do not need to find it)? Motivate your answers.

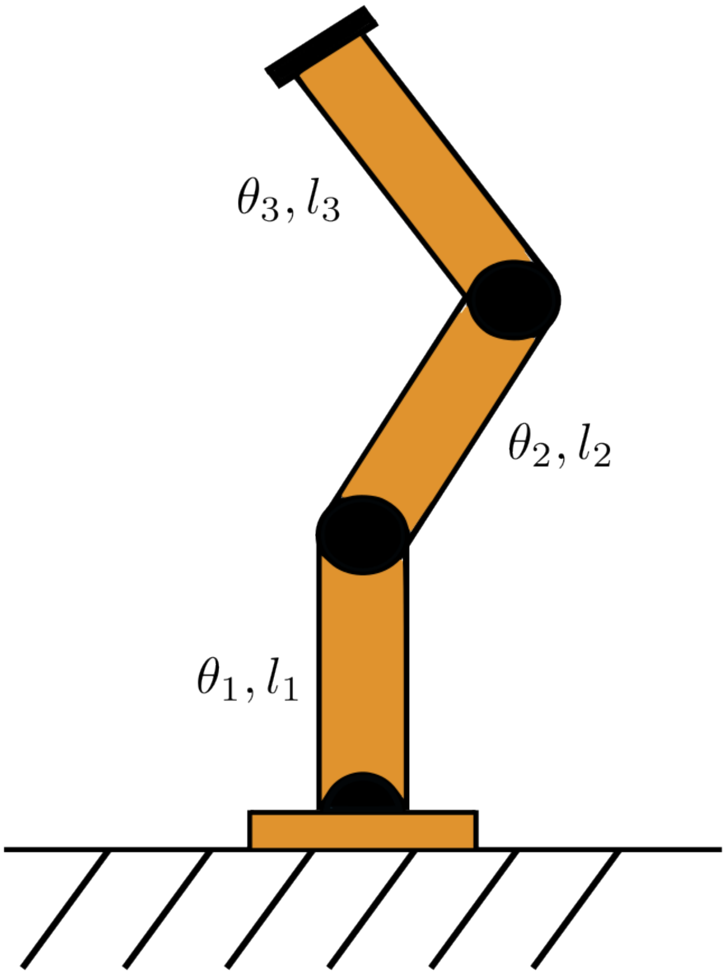

Figure 3: Three-link planar robot with joint angles θ_1, θ_2, θ_3 and link lengths l_1, l_2, l_3.

4 Motion Planning

You are examining a two-link planar robot, see Figure 4. The robot is represented by the three colorful circles and the two blue lines along the y-axis in the figure. The robot's base is at $(0, 0)$. The joint at the base is rotational. The first link goes from $(0, 0)$ to $(0, 2)$ where the robot has another rotational joint. Finally, the second link goes from $(0, 2)$ to $(0, 4)$ where the end-effector/tool is located. You can consider $(0, 4)$ as the starting point for all paths.

(a) Recall the equations for the forward and inverse kinematics for the two-link planar robot from Lectures 2–3.

(b) Use inverse kinematics to compute the corresponding joint angles for each of the starting point and points A, B, C, and D.

Plan a joint-interpolated trajectory (corresponding to `MoveJ` in the RAPID programming language) for each of the two joints that reaches the point A and then moves $A \to B \to C \to D$. Use a third-order polynomial for each of the four segments (see "Cubic Polynomial Trajectory" in Lecture 4).

You can assume that the robot starts and stops with zero joint velocity for each segment (corresponding to a "fine zone" in the RAPID programming language). Moreover, you can assume that each motion segment should take 2 seconds to complete.

(c) Plot the angular position and the corresponding angular velocity for each joint as function of time. Plot also the position of the robot end-effector/tool as function of time.

(d) Discuss potential problems with applying this kind of planning on a real robot. What would the advantages be of using higher-order polynomials, what could be the possible disadvantages? Which path would be preferable to use for a real robot and why? How can the trajectory generation be extended if the robot should not stop at each point, but rather have a certain specified velocity?

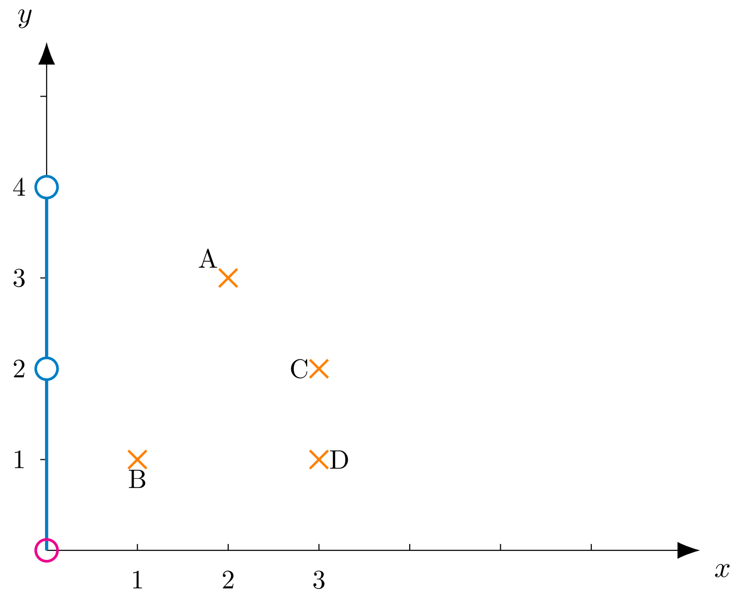

Figure 4: Two-link planar robot and the $ABCD$ path.

5 Peter Corke's Robotics Toolbox and Dynamics

(a) Use Peter Corke's Robotics Toolbox (recommended release 10.2 or later) to set up a model of the 3-DoF robot, which corresponds to the first three joints of IRB140, see Figure 5.

You should in addition to the kinematic parameters also supply enough parameters (choose 'realistic masses', the total robot weight is about 100 kg) to be able to get a dynamic model representation. You do not have to present an explicit model, but just define it to solve the sub-problems below.

Make sure you choose your direction of the rotational axes (i.e., the z-axes for the DH-parameters) to get the same angle convention (direction and zero-position) as in RobotStudio (compare and verify).

(If you need to change the zero-position/offset for some link, you can use the Link-parameter offset, property of the Link-object in Peter Corke's toolbox).

(b) Find out how much (stationary) torque is needed on the motor side to overcome the influence of gravity for the configuration ($\theta_1 = 0°$, $\theta_2 = 20°$, $\theta_3 = 30°$). You can approximate the gear ratios to 1:100 for all three motors. Mention some benefits why you in industrial robotics want to have a gear-box with a gear ratio providing significantly 'higher speed on the motor-side than on the arm-side' through the gearbox. Also comment on some drawback(s) with gear-boxes.

(c) Make a path-generation/simulation for joint-wise or linear motion (corresponding to `MoveJ` and `MoveL`, respectively, in the RAPID programming language) from

$$\begin{pmatrix}\theta_1 & \theta_2 & \theta_3\end{pmatrix} = \begin{pmatrix}20^\circ & 20^\circ & 30^\circ\end{pmatrix} \text{ to } \begin{pmatrix}-10^\circ & 40^\circ & 0^\circ\end{pmatrix}$$

The robot is supposed to start and stop (i.e., zero velocity) at these points.

(d) Make a simulation of a "free-falling" robot from its home-position (joint torques = 0). No considerations need to be taken to joint limitations and colliding links.

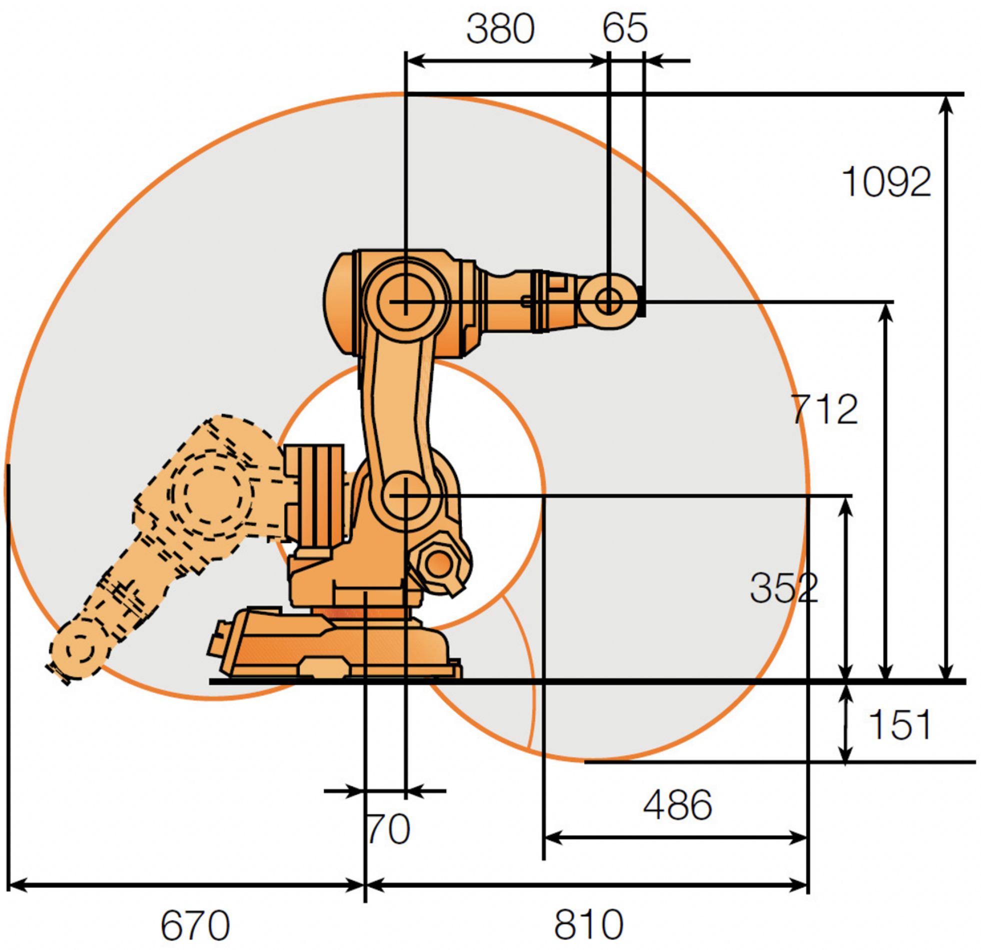

Figure 5: The IRB140 robot (the robot used in the RobotStudio Lab exercises). ABB Robotics (2004): "Operating manual — IRC5 with FlexPendant", Document ID: 3HAC16590-1, Revision: P.