Industrial Robotics Assignment 2

Lecturer: Francesco Giorgio-Serchi

November 19, 2025

Exercise 1

1. Explain with words, schematics and mathematically what forward kinematics and inverse kinematics are and how they are used in industrial robotics by giving practical examples.

2. Then give a clear mathematical explanation of how to solve an inverse kinematics problem numerically for a general spatial manipulator with multiple degrees of freedom, providing a detailed schematic explanation of the mathematical steps involved in the iterative solution.

3. Then take the manipulator depicted in Fig. 4 and write an inverse kinematic solver in Matlab for this robot (without using any of the functions pre-made in the matlab library, such as *IKinSpace*) and prove that your solver works correctly by testing for various configurations of your choice.

4. Finally, can you find configurations where your inverse kinematics solver cannot find a solution for $\vec{\theta}$? explain why.

Exercise 2

With reference to the schematics shown in Figure 1: the body shown at the top right of the figure rotates about the point with coordinates (L, L) with an angular velocity $\dot{\theta} = 1\ rad/s$. Solve the following exercises by hand, explaining each passage and calculation in detail:

1. express the position of point P identified by the vector $\vec{p}_P$ on the moving body relative to the fixed reference frame $\{s\}$ in terms of θ, and then express the velocity $\dot{\vec{p}}_P$ of point P in terms of the fixed frame $\{s\}$,

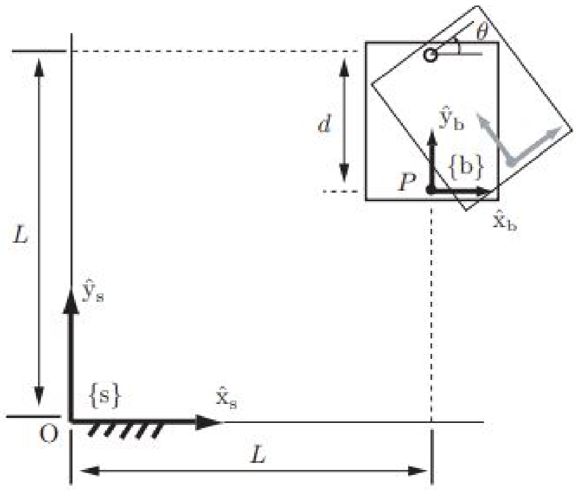

Figure 1: (From exercise 2) Schematic of 3R planar manipulator.

2. express the Homogenous Transformation Matrix T_{sb}, that represents the configuration of frame $\{b\}$, as seen from the fixed frame $\{s\}$,
3. write the expression for the twist of T_{sb}, called $\mathcal{V}_b$, in body coordinates attached to the body $\{b\}$; then write the expression for the twist of T_{sb} in space coordinates, called $\mathcal{V}_s$, with respect to the fixed frame $\{s\}$,
4. express the relationship that relates the twists $\mathcal{V}_s$ and $\mathcal{V}_b$ from exercise 2.4 and 2.5,
5. write a Matlab script that solves exercises from 2.1 to 2.6 when $L = 1.0\ m$, $d = 0.4\ m$ and $\theta = 30°$.

Exercise 3

The company Unitree is a chinese manufacturer of quadruped robots, also known as legged robots. One example of such robots is the one shown in Fig. 2(a). These robots employ a locomotion analogous to that of dogs, horses, etc. In order to enable smooth locomotion on all types of terrains, the robot controller needs to constantly solve an inverse kinematic problem to ensure that each leg follows a desired trajectory according to obstacles along the path: for example while climbing a staircase, Fig. 2(b).

In this exercise you are required to focus only on the kinematics of the front leg, as schematically shown in the Fig. 2(c). The fixed reference frame is assumed to be located at the Centre of Mass (CoM) of the robot, distant $L_0 = 0.4$m from the centre of *joint-1*. The front leg consists of two revolute joint, respectively located at the shoulder joint and at the knee joint and indicated in the schematics

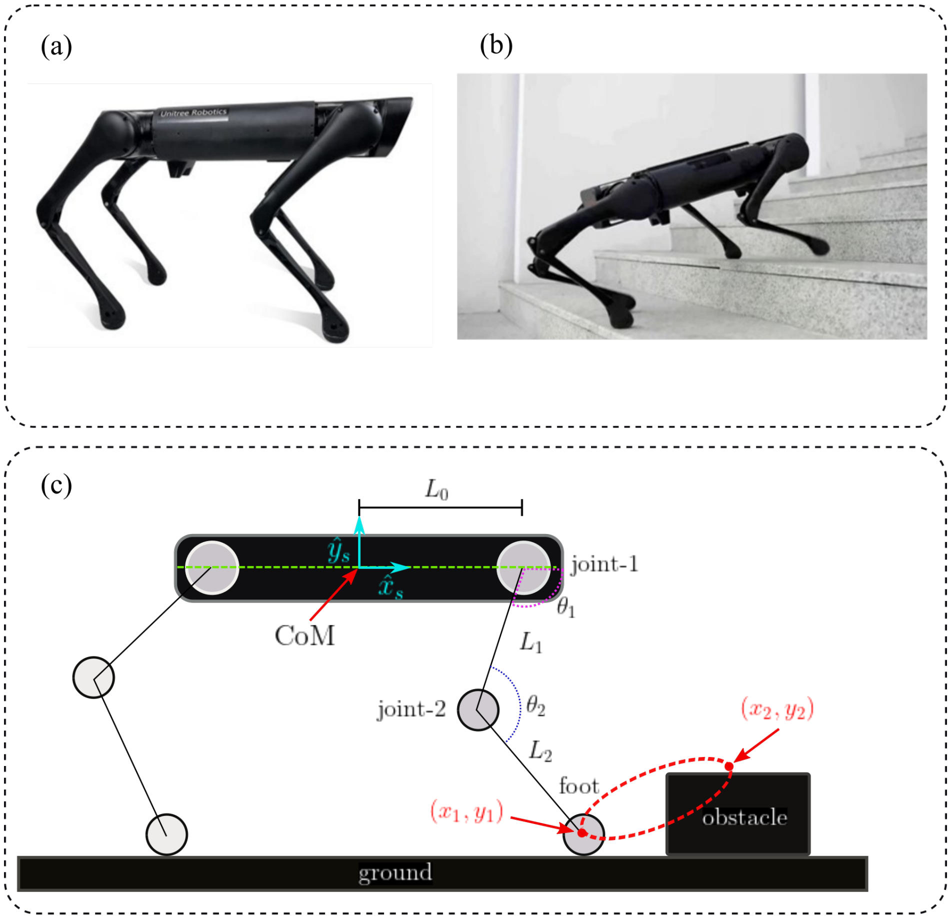

Figure 2: (From exercise 3) Schematic of the Unitree legged robot approaching an obstacle.

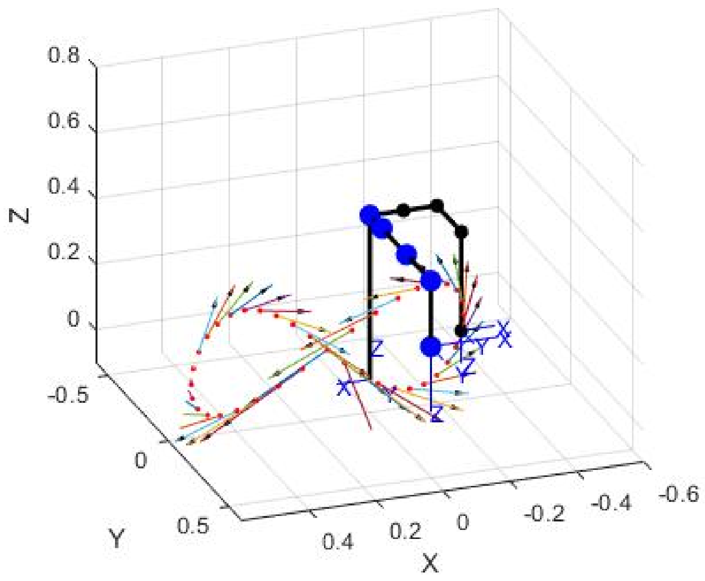

Figure 3: (From exercise 3) style of vector plotting required for foot velocity during trajectory.

with *joint-1*, θ_1, and *joint-2*, θ_2. The link lengths, which define the upper and lower limbs of the leg are respectively indicated with $L_1 = 0.3$m and $L_2 = 0.4$m. With reference to the situation depicted in Fig. 2(c), where the front leg needs to climb the obstacle, solve the following problems.

1. Assume that the trajectory of the foot $\vec{x}_e(t)$ which enables effective climbing of the obstacle can be represented with an ellipse described by the following parametrization: $(x_1, y_1) = (0, 0)$ represents the initial position of the foot on the ground (in the foot reference frame), $(x_2, y_2) = (0.25, 0.2)$ represents the desired contact point of the foot on the obstacle (expressed with respect to the initial foot position on the ground), $e = 0.9$ represents the eccentricity of the ellipse (how elongated the ellipse trajectory is along the major semi-axis), then having defined

$$\begin{aligned} a &= \frac{1}{2}\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \\ b &= a\sqrt{1 - e^2} \\ w &= \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right) \end{aligned}$$

   for $0° \leq \beta \leq 360°$ and $X = a\cos\beta$ and $Y = b\sin\beta$, we can define the trajectory of the foot as follows:

$$x_e(\beta) = \frac{x_1 + x_2}{2} + X\cos w - Y\sin w \quad (1)$$

$$y_e(\beta) = \frac{y_1 + y_2}{2} + X\sin w + Y\cos w \quad (2)$$

   Given the above parameters which define the trajectory, write a Matlab code that plots the above trajectory according to eq. 1 and eq. 2.

2. Assuming the centre of joint-1 to be located at (0, 0.5)m with respect to the foot frame, solve analytically the inverse kinematic of the leg in order to compute the joint angles $\vec{\theta}$ for the two configurations of the foot at (x_1, y_1) and (x_2, y_2) and show your hand-calculations (place your reference frames where you find it more convenient for you).

3. then re-express the parameter β as a function of time $\beta(t) = \dot{\beta}t$ with $\dot{\beta} = 1rad/s$ and re-write the position of the foot along the trajectory $\vec{x}_e(t) = (x_e(t), y_e(t))$ as a function of time. Then write a Matlab code that solves numerically the inverse kinematic for $\vec{\theta}$ along this trajectory in time and present the results by plotting the foot position in time (with respect to the CoM reference frame) and the joint angles in time.

4. assuming that the maximum joint velocities are $\dot{\theta}_1, \dot{\theta}_2 = 5rad/s$, test for various $\dot{\beta}$ until you can confirm that the foot can follow the trajectory respecting the maximum joint velocity. Explain carefully, and with a lot of details, your reasoning and your code to achieve this result.

5. Finally compute the Jacobian of the leg and use the joint velocities from exercise 3.4 to compute the foot velocity at each instant in time, plotting the leg velocity, the joint velocities and the leg with velocity vectors at each instant in time similarly to what is shown in Fig. 3.

Exercise 4

Hyper-redundant manipulators are manipulators with more than 6 Degrees of Freedom which are suitable for undertaking complex tasks; an example of this type of manipulators is given in Fig. 4. Write a series of Matlab scripts and provide clear written explanation in the text about all the mathematical steps to solve the following problems:

1. prescribe a linear motion of the end-effector with speed $\dot{x} = -0.1$m/s, $\dot{y} = 0.0$m/s, $\dot{z} = 0.05$m/s (expressed with respect to the base frame), and solve the Inverse Kinematics (without using the function *IKinSpace* from the Matlab library) to enable the end-effector to follow this motion for $t = 2$s starting from the configuration

$$T = \begin{bmatrix} 0 & 0 & 1 & 0.5 \\ 0 & 1 & 0 & 0.1 \\ -1 & 0 & 0 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and plot the motion of the whole robot during this trajectory and the joint velocities.

2. prescribe a point-to-point trajectory of the end-effector from:

$$T = \begin{bmatrix} 0 & 0 & 1 & 0.5 \\ 0 & 1 & 0 & 0.1 \\ -1 & 0 & 0 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ at } t = 0s$$

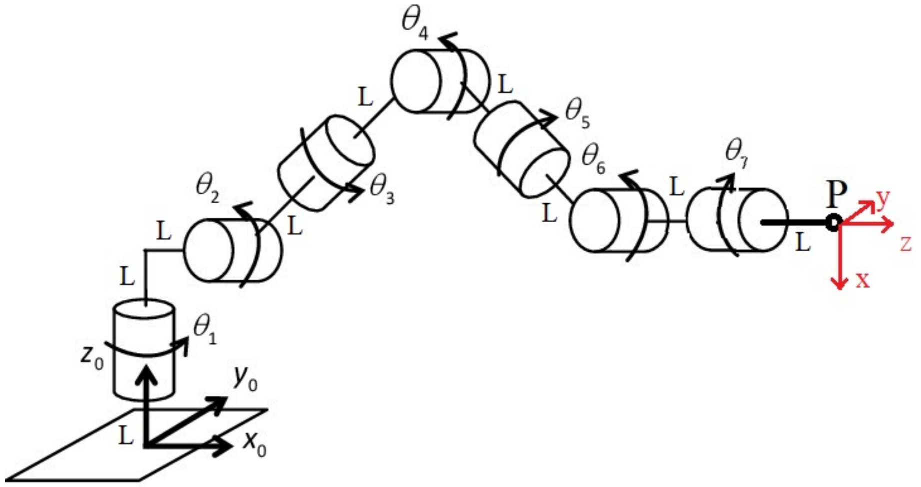

Figure 4: (From exercise 4) Schematic of a redundant manipulator.

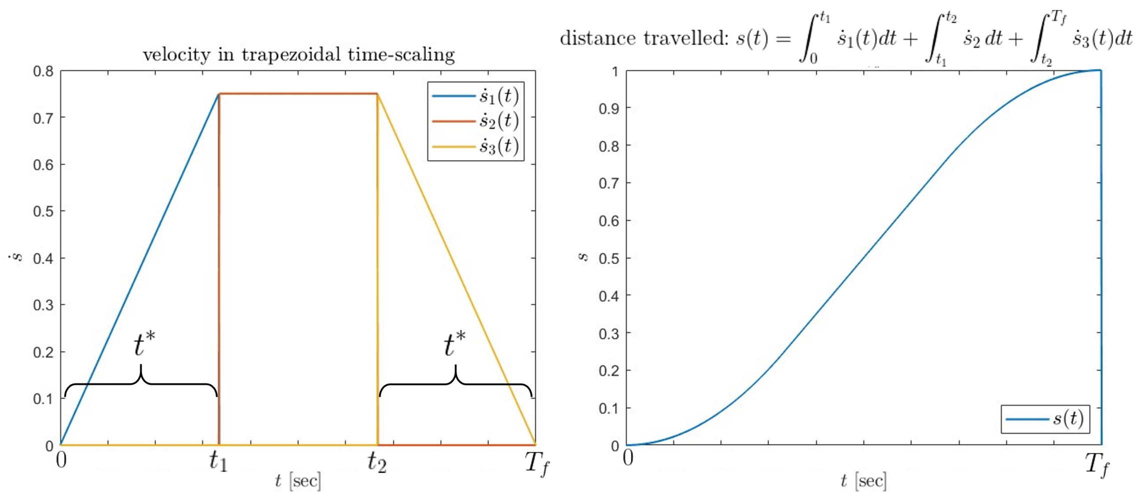

Figure 5: (From exercise 4) Example of trapezoidal velocity profile and the resultant displacement profile. Notice the parameter t^* which defines the rise time of the velocity.

to

$$T = \begin{bmatrix} 1 & 0 & 0 & 0.1 \\ 0 & 1 & 0 & 0.1 \\ 0 & 0 & 1 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ at } t = 4s$$

using the function *CartesianTrajectory* to generate the trajectory and its timescaling and solve the Inverse Kinematics to drive the end-effector. Plot the whole motion in the best way you can.

3. The function *CartesianTrajectory* builds a Point-to-Point trajectory using the cubic time-scaling. In class we have also seen a simpler time-scaling, where the velocity $\dot{s}(t)$ has a triangular routine. Implement this type of triangular time-scaling in Matlab and modify the existing *CartesianTrajectory* function so that it can work with this new type of time-scaling. Demonstrate that your newly-coded function works by plotting an example of a Point-to-Point trajectory with triangular velocity routine.

4. Derive another type of time-scaling based on a trapezoidal velocity profile, as demonstrated in Fig. 5. In this type of velocity routine, you impose the total duration of the trajectory T_f and the rise-time t^*, which represents the time interval during which the velocity increases and decreases, respectively between $0 \le t \le t_1$ and $t_2 \le t \le T_f$. Similarly to the triangular velocity routine shown in class, this velocity routine is composed of three segments: $\dot{s}_1$, $\dot{s}_2$ and $\dot{s}_3$. Integrating these three velocities over their respective time-interval and imposing the constraints $s(T_f) = 1$, yields a function for the time-scaling $s(t)$ which depends on T_f and t^*. Write down the math and implement the calculation in a Matlab code that produces a plot similar to the one shown in Fig. 5. Show how the shape of $s(t)$ changes when you change the duration of t^*?

Exercise 5

Humanoid robots are becoming very common in industrial robotics. These robots use manipulators which have the same kinematic structure of a human arm. We call this type of robotics arm *antropomorphic* manipulators. An example of such manipulator and the robot on which they are mounted is shown in Fig. 6. The robot shown in Fig. 6 is the SE01 from Shenzhen EngineAI Robotics. This manipulator has 6 revolute joints, where the last three θ_4, θ_5, θ_6 constitute a *wrist* joint, this means that these three joints are geometrically coincident (i.e. there are no links between them). The other joints are connected by links $L_1 = 0.15\ m$, $L_2 = 0.1\ m$, $L_3 = 0.25\ m$, $L_4 = 0.2\ m$. You are required to do the following:

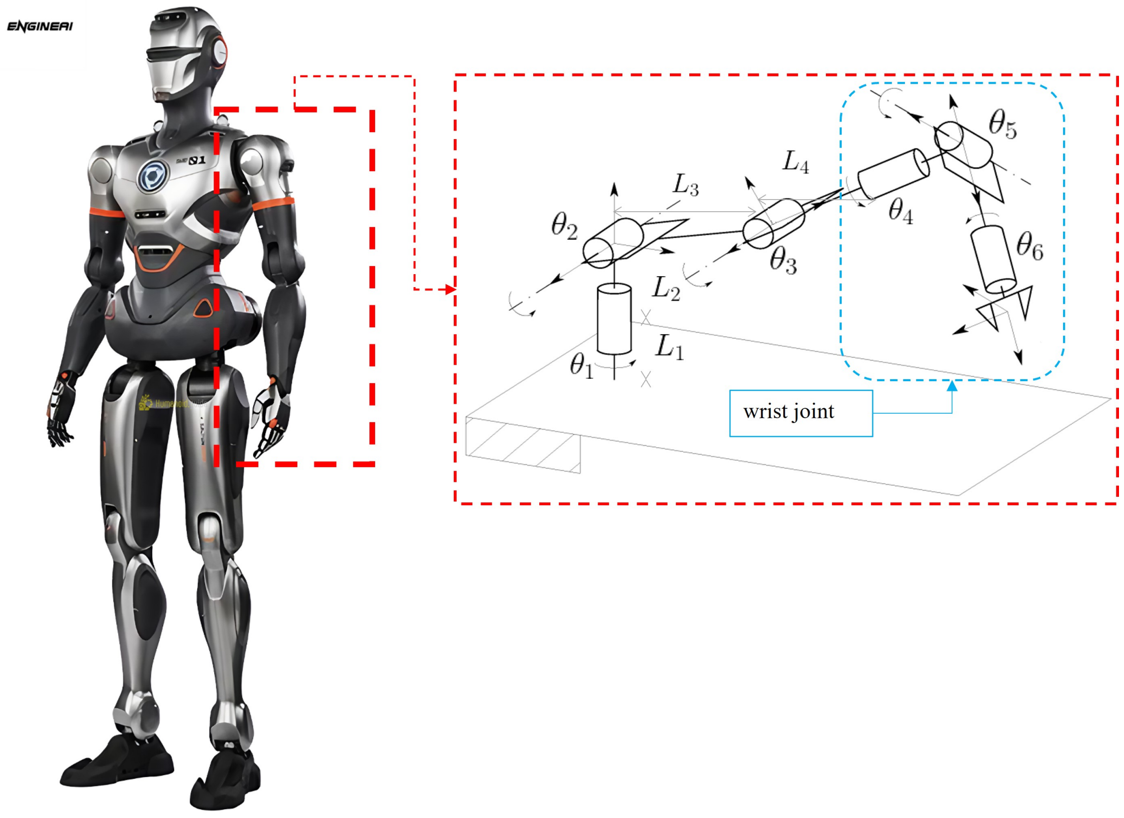

Figure 6: (From exercise 5) Schematic of a humanoid $6R$ antropomorphic manipulator with a wrist joint analogous to the one used in the humanoid robot ENGINEAI-SE01 from the Shenzhen company EngineAI Robotics.

1. write the Screw Axes and HTMs of all joints at the home configuration (you decide the home configuration) and formulate mathematically the Forward Kinematic solution, then write a Matlab script to solve the forward kinematic and draw the workspace of this manipulator in the case where joints 1, 2 and 3 have the following mechanical constraints: $-90° < \theta_1 < 90°$, $-70° < \theta_2 < 90°$, $-90° < \theta_3 < 90°$;

2. write the Jacobian matrix of this manipulator and create a Matlab script that computes the Jacobian (without using the function *JacobianSpace*) when $\vec{\theta} = (50°,\ 10°,\ -15°,\ 30°,\ 30°,\ -15°)$ reporting the results and plotting the vectors which represent each column of the Jacobian in this configuration, then compute and plot the end-effector velocity in the case where the joint velocities are instantaneously $\dot{\vec{\theta}} = (10,\ 80,\ -50,\ 15,\ 20,\ -50)\,\deg/s$;

3. write a Matlab script (without using the function *IKinSpace.m*) that solves the inverse kinematics of this manipulator when the desired end-effector pose is $\vec{x}_e = (0.2\,m,\ 0.1\,m,\ 0.3\,m,\ 10°,\ 30°,\ 20°)$ and when the desired end-effector pose is given by:

$$T_{se} = \begin{bmatrix} 0 & -1 & 0 & 0.2 \\ 1 & 0 & 0 & 0.3 \\ 0 & 0 & 1 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

   make sure to report the results in a professional manner.

4. write a Matlab script that performs a point-to-point trajectory from $\vec{x}_e$ to T_{se} from exercise 5.3. The motion from one pose to the other is executed in a time interval $T_f = 0.5\ s$ and follows a cubic time scaling. Plot the solution of the inverse kinematic and the manipulator

configuration at 20 timesteps during the trajectory. Finally plot the three linear velocities and three rotation velocities of the end-effector in time at each timestep and plot the joints' angles and joints' angular velocities in time at each timestep (you can use the Matlab library functions *IKinSpace.m* and *CartesianTrajectory.m*). Once again, report all your results as professionally as possible. If the maximum joint velocity of any joint is $\dot{\theta} = 30deg/s$ how long should T_f be in order to respect joint velocity constraint?

5. build a multi-waypoint trajectory that starts from the *home configuration*, then travels to the pose $\vec{x}_e$, then travels to T_{se} and then comes back to the *home configuration* in an interval $T_f = 25.0s$ with 100 timesteps. Then solve the inverse kinematic for this entire trajectory and plot the values of the end-effector position, euler angles and joint values in time at each timestep. Explain schematically how the code works and report the results in a professional manner.