Università
di Genova

UNIVERSITÀ DEGLI STUDI DI GENOVA

DIBRIS

DEPARTMENT OF INFORMATICS,
BIOENGINEERING, ROBOTICS AND SYSTEM ENGINEERING

MODELLING AND CONTROL OF MANIPULATORS

Third Assignment

Jacobian Matrices and Inverse Kinematics

Author:

Surname Name

Student ID:

s0000000

Professors:

Enrico Simetti
Giorgio Cannata

Tutors:

Luca Tarasi
Simone Borelli

December 15, 2025

Contents

Mathematical expression	Definition	MATLAB expression
$< w >$	World Coordinate Frame	w
${}^{a}_{b}R$	Rotation matrix of frame $< b >$ with respect to frame $< a >$	aRb
${}^{a}_{b}T$	Transformation matrix of frame $< b >$ with respect to frame $< a >$	aTb
${}^{a}O_{b}$	Vector defining frame $< b >$ wit respect to frame $< a >$	aOb

Table 1: Nomenclature Table

1 Assignment description

The third assignment of Modelling and Control of Manipulators focuses on Inverse Kinematics (IK) control of a robotic manipulator.

The third assignment consists of three exercises. You are asked to:

- Download the .zip file called from the Aulaweb page of this course.
- Implement the code to solve the exercises on MATLAB by filling in the predefined files.
- Write a report motivating your answers, following the predefined format on this document.
- **Do NOT put your code in the report!**

1.1 Exercise 1

Given the geometric model of an industrial manipulator in Figure 1, you have to add a tool frame. The tool frame is rigidly attached to the robot end-effector according to the following specifications: $\Gamma_{t/e} = [\pi/10, 0, \pi/6], {}^{e}O_t = [0.3, 0.1, 0]^{\top}(m)$ where $\Gamma_{t/e}$ represents the YPR values from end effector frame to tool frame.

To complete this task you should modify the class *geometricModel* by adding a new method called *getToolTransformWrtBase*.

1.2 Exercise 2

Implement an inverse kinematic control loop to control the tool of the manipulator. You should be able to complete this exercise by using the MATLAB classes implemented for the previous assignment (*geometricModel*,*kinematicModel*), and also you need to implement a new class *cartesianControl* (see the template attached). The initial joint position is $q = [\pi/2, -\pi/4, 0, -\pi/4, 0, 0.15, \pi/4]^{\top}$, expressed in radians and meters. The procedure can be split into the following phases

Q2.1 Compute the cartesian error between the robot end-effector frame ${}^{b}_{t}T$ and the goal frame ${}^{b}_{g}T$.

The goal frame must be defined knowing that:

- The goal position with respect to the base frame is ${}^{b}O_g = [0.2, -0.7, 0.3]^{\top}$ (m)
- The goal frame is rotated by the YPR angles $\Gamma_g = [0, 1.57, 0]^{\top}$ (rad) with respect to the base frame.

Q2.2 Compute the desired angular and linear reference velocities of the tool with respect to the base: ${}^{b}\nu^{*}_{t/b} = \begin{bmatrix} \kappa_a & 0 \\ 0 & \kappa_l \end{bmatrix} \cdot {}^{b}e$, such that $\kappa_a = 0.8$,$\kappa_l = 0.8$ is the gain.

Q2.3 Compute the desired joint velocities $\dot{q}$

Q2.4 Simulate the robot motion by implementing the function: *"KinematicSimulation()"* for integrating the joint velocities in time.

Q2.5 Compute the end effector and tool linear and angular velocities with respect to the base frame projected on the base frame.

Remark 1: All the methods must be implemented for a generic serial manipulator. For instance, joint types, and the number of joints should be parameters.

Remark 2: The class *plotManipulators* is provided for plot generation. You must not modify its implementation, or its calls in *main.m.*

2 Exercise 1

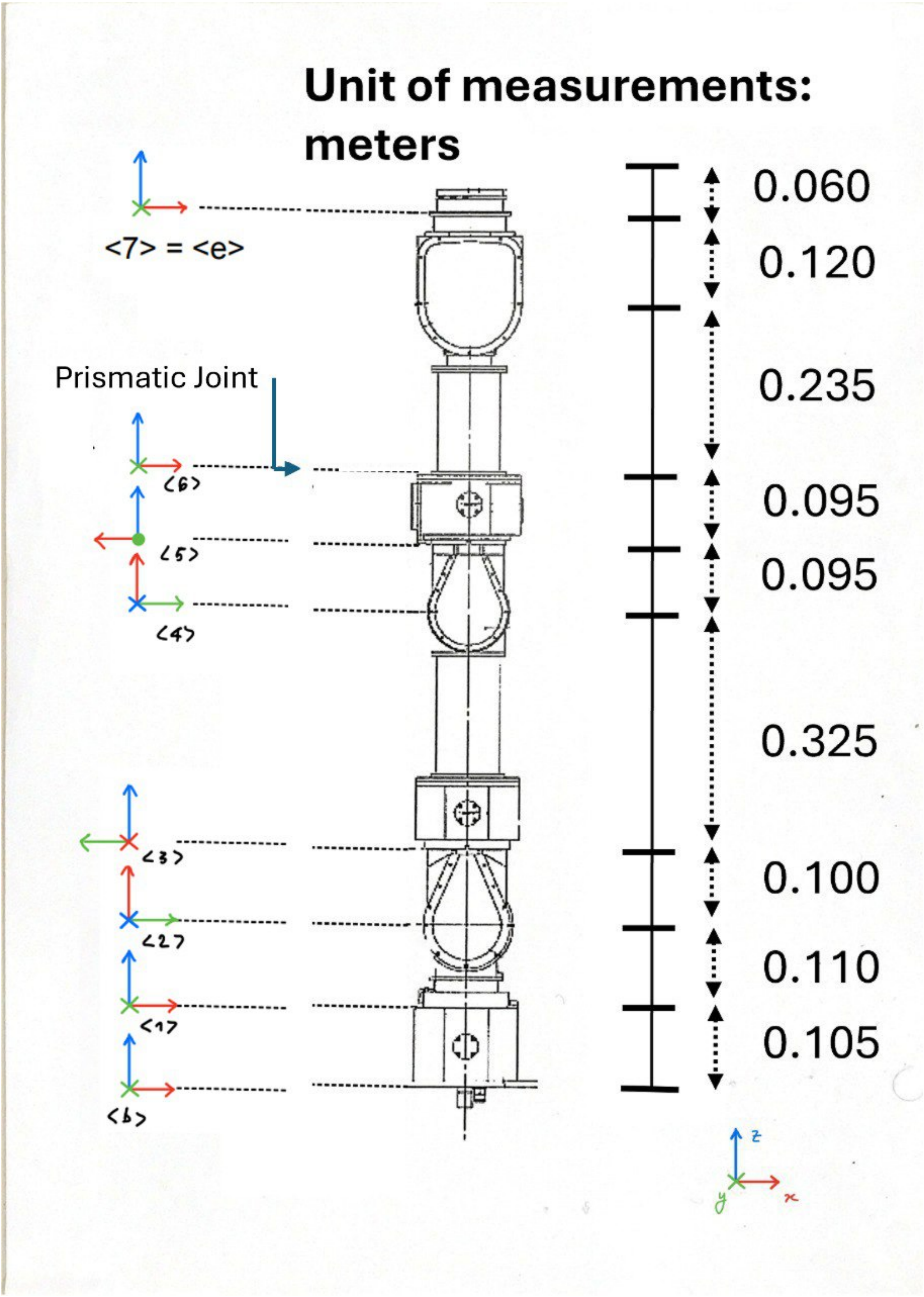

Figure 1: Manipulator CAD