UNIVERSITÀ DEGLI STUDI DI GENOVA

DIBRIS

DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY,
BIOENGINEERING, ROBOTICS AND SYSTEM ENGINEERING

MODELLING AND CONTROL OF MANIPULATORS

Second Assignment

Manipulator Geometry and Direct Kinematics

Author:

Condorelli Valentina
Meschini Marco
Piacenza Enrico

Student ID:

s4945679
s6273938
s4878469

Professors:

Enrico Simetti
Giorgio Cannata

Tutors:

Andrea Tiranti
Francesco Giovinazzo
George Kurshakov

November 30, 2023

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

Table 1: Nomenclature Table

1 Assignment description

The second assignment of Modelling and Control of Manipulators focuses on manipulators geometry and direct kinematics.

- Download the .zip file called MOCOM-LAB2 from the Aulaweb page of this course.
- Implement the code to solve the exercises on MATLAB by filling the predefined files called "*main.m*", "*BuildTree.m*", "*GetDirectGeometry.m*", "*DirectGeometry.m*", "*GetTransformationWrtBase.m*", "*GetBasicVectorWrtBase.m*" and "*GetFrameWrtFrame.m*".
- Write a report motivating your answers, following the predefind format on this document.

1.1 Exercise 1

Given the following CAD model of an industrial 7 dof manipulator:

Q1.1 Define all the model matrices, by filling the structures in the *BuildTree()* function. Be careful to define the z-axis coinciding with the joint rotation axis, and such that the positive rotation is the same as showed in the CAD model you received. Draw on the CAD model the reference frames for each link and insert it into the report.

Q1.2 Implement a function called *DirectGeometry()* which can calculate how the matrix attached to a joint will rotate if the joint rotates. Then, develop a function called GetDirectGeometry() which returns all the model matrices given the following joint configurations:

- **q** $= [0, 0, 0, 0, 0, 0, 0]$.
- **q** $= [0, 0, 0, 0, 0, \pi/2, 0]$.
- **q** $= [0, \pi/2, 0, -\pi/2, 0, 0, 0]$.
- **q** $= [\pi/4, \pi/2, -\pi/8, -\pi/2, \pi/4, 2/3\pi, 0]$.

Comment the results obtained.

Q1.3 Calculate all the transformation matrices between any two links, between a link and the base and the corresponding distance vectors, filling respectively: *GetFrameWrtFrame()*, *GetTransformationWrtBase()*, *GetBasicVectorWrtBase()*

Q1.4 Given the following starting and ending configuration:

- **q**$_i = [0, 0, 0, 0, 0, 0, 0]$ and **q**$_f = [\pi/4, \pi/2, -\pi/8, -\pi/2, \pi/4, 2/3\pi, 0]$
- **q**$_i = [0, \pi/2, 0, -\pi/2, 0, 0, 0]$ and **q**$_f = [0, 0, 0, 0, 0, 0, 0]$
- **q**$_i = [1.3, 0.1, 0.1, 1, 0.2, 0.3, 1.3]$ and **q**$_f = [2, 2, 2, 2, 2, 2, 2]$

Plot the intermediate link positions in between the two configurations (you can use the plot3() or line() functions) and comment the results obtained.

Q1.5 Test your algorithm by changing one joint position at the time and plot the results obtained for at least 3 configurations.

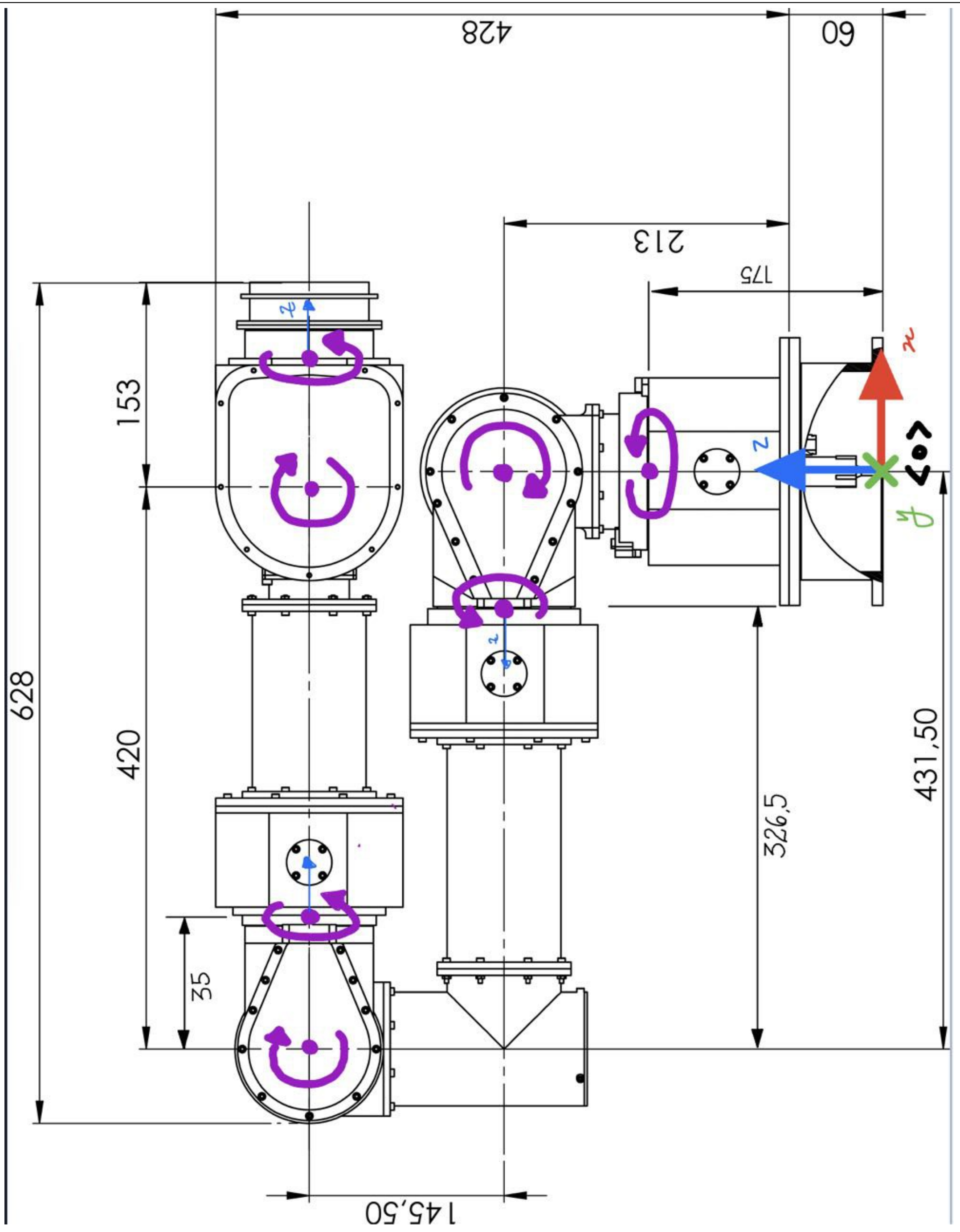


Figure 1: CAD model of the robot

2 Exercise 1

2.1 Introduction

The aim of this laboratory was to comprehend and analyse the geometry and the direct kinematics of a 7-DOF manipulator, shown in Fig.1. Firstly, we put the frames on each joint following the Denavit-Hartenberg convention:

1. **Z-Axis:** the axis Z_i is chosen to coincide with the axis of joint $i+1$, so that the resulting rotation is positive along this axis.
2. **X-Axis:** the axis X_i runs along the common normal between the axes Z_{i-1} and Z_i, with with positive orientation from joint *i* to joint *i+1*
3. **Y-Axis:** the Y axis is created to complete the right-handed frame

After that, we defined some functions to build the geometric model of the manipulator. In the end, we plotted on 3D environment different configurations of it, with the corresponding motions.

2.2 Q1.1

The first step was to apply the convention explained above, obtaining the frames shown in Fig.2. Then, we defined the function *BuildTree()*, which computes, for each frame, the corresponding transformation matrix with respect to the previous one. Both the rotation matrices and the translation vectors were found by inspection of the CAD model. As a result, the following geometric model was obtained:

$$ {}^0_1T = \begin{bmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 1.0000 & 0.1750 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{1} $$

$$ {}^1_2T = \begin{bmatrix} -1.0000 & 0 & 0 & 0 \\ 0 & 0 & 1.0000 & 0 \\ 0 & 1.0000 & 0 & 0.0980 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{2} $$

$$ {}^2_3T = \begin{bmatrix} 0 & 0 & 1.0000 & 0.1050 \\ 1.0000 & 0 & 0 & 0 \\ 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{3} $$

$$ {}^3_4T = \begin{bmatrix} -1.0000 & 0 & 0 & 0.1455 \\ 0 & 0 & 1.0000 & 0 \\ 0 & 1.0000 & 0 & 0.3265 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{4} $$

$$ {}^4_5T = \begin{bmatrix} -1.0000 & 0 & 0 & 0 \\ 0 & 0 & -1.0000 & -0.0350 \\ 0 & -1.0000 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{5} $$

$$ {}^5_6T = \begin{bmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & 0 & -1.0000 & 0 \\ 0 & 1.0000 & 0 & 0.3850 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{6} $$

$$ {}^6_7T = \begin{bmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & 0 & 1.0000 & 0.1530 \\ 0 & -1.0000 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{7} $$

After verifying the correctness of the geometric model we initialized two useful variables in the main function:

- The number of links (*numberOfLinks*), initialized to 7.
- The joints type (*JointType*) which, in the configuration provided, was always rotational.

Finally, we preallocated the basic translational vectors matrix *bri*, containing the vectors of each frame with respect to the base, and the 3-dimensional matrix *bTi*, containing the transformation matrices of each frame with respect to the base.

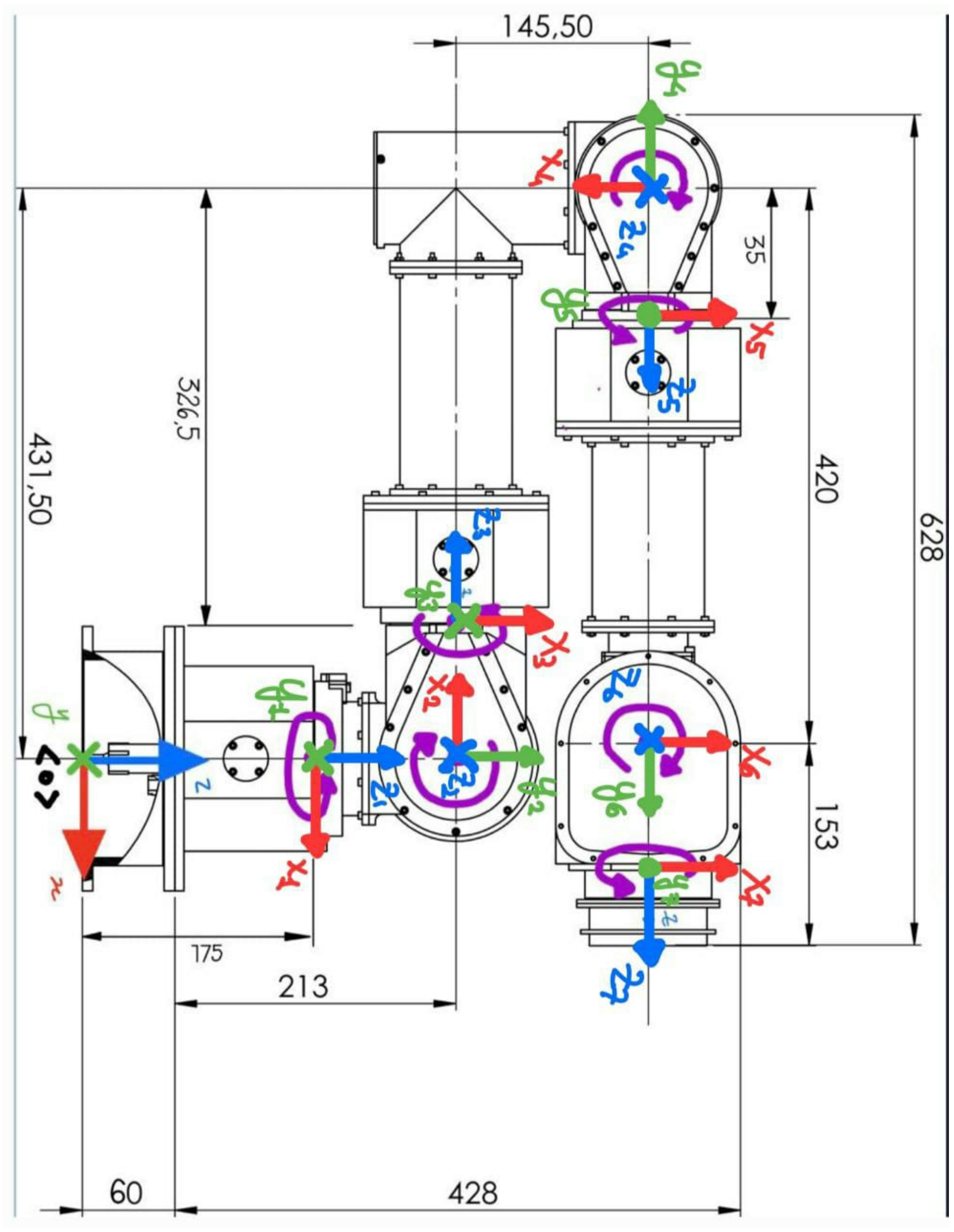


Figure 2: Reference frames on the model

2.3 Q1.2

In this section, we defined the function *DirectGeometry()*, which takes as inputs:

- The current joints position (*qi*).
- The transformation matrix between the i-th and the j-th frames in the initial configuration (*iTj*).
- The i-th joint type (*JointType_i*).

The output is the transformation matrix between the i-th and j-th frames for the given configuration.
For this computation, the function implements the following formulas:

	R	P
$\underline{r}_{\frac{i+1}{i}}$	$\underline{r}_i$	$\underline{r}_i + \underline{k}_{i+1} q_{i+1}$
${}^{i}_{i+1}R$	$R_i \cdot R_z(q_{i+1})$	R_i

With $R_z(\alpha) = \begin{bmatrix} cos(\alpha) & -sin(\alpha) & 0 \\ sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix}$.
The next step was to implement the function *GetDirectGeometry()*, which executes the *DirectGeometry()* function for 7 (*numberOfLinks*) times.
Finally, the *GetDirectGeometry()* function was tested on four different joint configuration:

- **q** $= [0, 0, 0, 0, 0, 0, 0]$
- **q1** $= [0, 0, 0, 0, 0, \pi/2, 0]$
- **q2** $= [0, \pi/2, 0, -\pi/2, 0, 0, 0]$
- **q3** $= [\pi/4, \pi/2, -\pi/8, -\pi/2, \pi/4, 2/3\pi, 0]$

As a result, four different 3-dimensional matrices were obtained: iTj_q, iTj_q1, iTj_q2 and iTj_q3. For the sake of simplicity, only iTj_q3 is reported below:

$$ {}^{0}_{1}T_{q_3} = \begin{bmatrix} 0.7071 & -0.7071 & 0 & 0 \\ 0.7071 & 0.7071 & 0 & 0 \\ 0 & 0 & 1.0000 & 0.1750 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{8} $$

$$ {}^{1}_{2}T_{q_3} = \begin{bmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 1.0000 & 0 \\ 1.0000 & 0.0000 & 0 & 0.0980 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{9} $$

$$ {}^{2}_{3}T_{q_3} = \begin{bmatrix} 0 & 0 & 1.0000 & 0.1050 \\ 0.9239 & 0.3827 & 0 & 0 \\ -0.3827 & 0.9239 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{10} $$

$$ {}^{3}_{4}T_{q_3} = \begin{bmatrix} 0 & -1.0000 & 0 & 0.1455 \\ 0 & 0 & 1.0000 & 0 \\ -1.0000 & 0 & 0 & 0.3265 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{11} $$

$$ {}^{4}_{5}T_{q_3} = \begin{bmatrix} -0.7071 & 0.7071 & 0 & 0 \\ 0 & 0 & -1.0000 & -0.0350 \\ -0.7071 & -0.7071 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{12} $$

$$ {}^{5}_{6}T_{q_3} = \begin{bmatrix} -0.5000 & -0.8660 & 0 & 0 \\ 0 & 0 & -1.0000 & 0 \\ 0.8660 & -0.5000 & 0 & 0.3850 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{13} $$

$$ {}^{6}_{7}T_{q_3} = \begin{bmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & 0 & 1.0000 & 0.1530 \\ 0 & -1.0000 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{14} $$

2.4 Q1.3

In this section, three different functions were developed:

- *GetTransformationWrtBase()*: computes the transformation matrix between the i-th joint and the manipulator base (*bTi*) as:
$$bTi = {}^{0}_{i}T = {}^{0}_{1}T \cdot {}^{1}_{2}T \ \ldots \cdot {}^{i-1}_{i}T \tag{15}$$
- *GetFrameWrtFrame()*: computes the transformation matrix between the i-th joint and the j-th joint (*iTj*) as:
  - ${}^{i}_{j}T = {}^{i}_{i+1}T \cdot {}^{i+1}_{i+2}T \ \cdot \ldots \cdot {}^{j-1}_{j}T$ if $i < j$;
  - ${}^{i}_{j}T = {}^{i}_{i-1}T \cdot {}^{i-1}_{i-2}T \ \cdot \ldots \cdot {}^{j-1}_{j}T$ if $i > j$
- *GetBasicVectorWrtBase()*: computes the translation vector between the i-th frame and the base, so 0 frame, by computing the transformation matrix between the i-th frame and the base (*bTi*). Then, the translation vector was obtained by extracting the first three rows of the last column.

Finally, all the functions were tested on the first (default) configuration **q** by giving as input the previously found transformation matrix iTj_q.

2.5 Q1.4

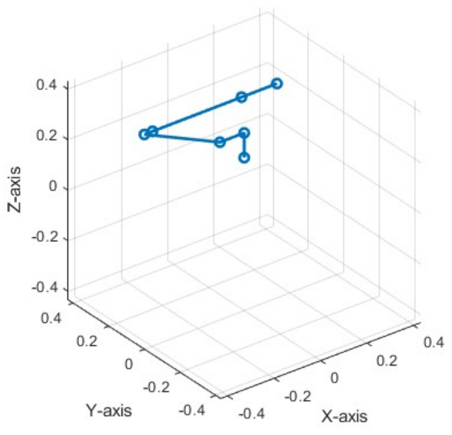
For this section, we used the Matlab built-in function *plot3()* to graphically represent the manipulator in a three-dimensional vector space, moving from a starting configuration ($\mathbf{q}_i$) to an ending one ($\mathbf{q}_f$).
For this purpose, two variables were defined:

- *numberOfSteps*: the number of iterations to reach the final configuration from the starting one. In order to have an animated plot which was both fluid and fast, this value was set to 30
- *delta*: the variation of each joint variable for each step of the animation

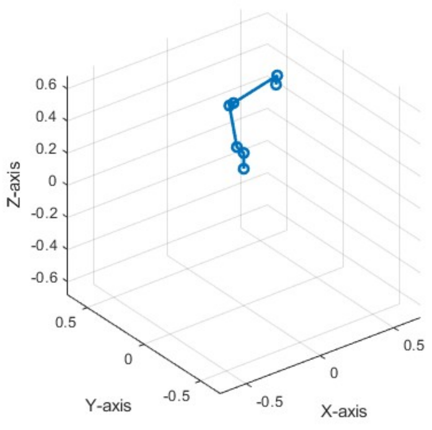
Firstly, we used the function *GetDirectGeometry()* to obtain the transformation matrices of each frame with respect to the previous one in the starting configuration.
Then, inside a for loop, we used the function *GetBasicVectorWrtBase()* to get the updated position of each joint with respect to the base. This step was repeated for the *numberOfSteps*.
Finally, this procedure was tested on three different sets of initial and final configurations:

- $\mathbf{q}_i = [0, 0, 0, 0, 0, 0, 0]$ and $\mathbf{q}_f = [\pi/4, \pi/2, -\pi/8, -\pi/2, \pi/4, 2/3\pi, 0]$
- $\mathbf{q}_i = [0, \pi/2, 0, -\pi/2, 0, 0, 0]$ and $\mathbf{q}_f = [0, 0, 0, 0, 0, 0, 0]$
- $\mathbf{q}_i = [1.3, 0.1, 0.1, 1, 0.2, 0.3, 1.3]$ and $\mathbf{q}_f = [2, 2, 2, 2, 2, 2, 2]$

Since the plot was animated, only the initial and final graphical results are shown for each set, respectively in Figure 3, Figure 4 and Figure 5.

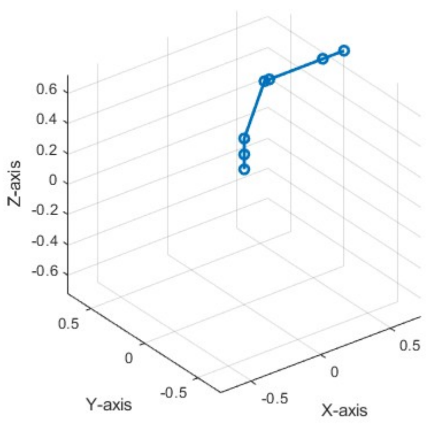


(a)

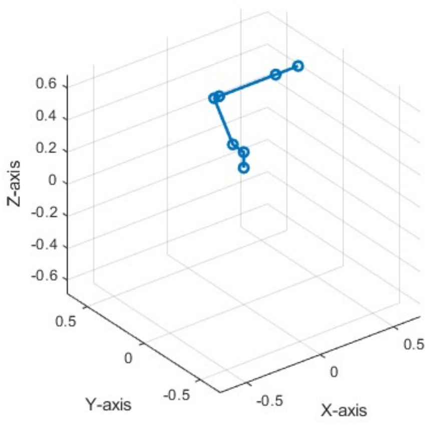


(b)

Figure 3: Initial and final graphical results for the first set

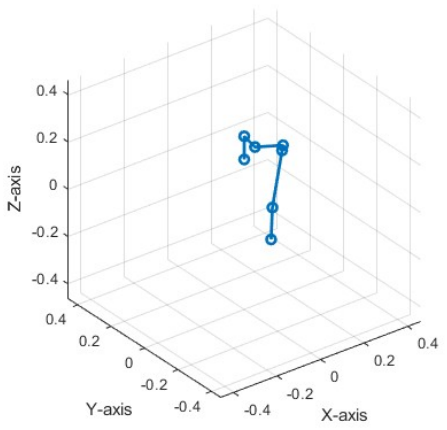


(a)

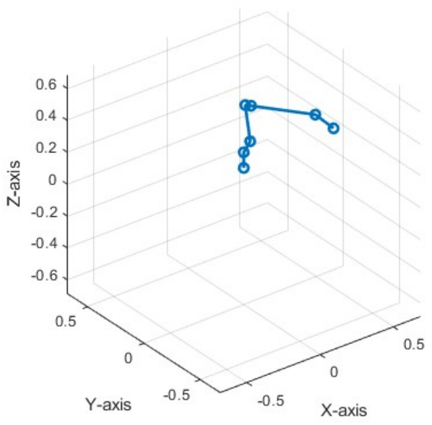


(b)

Figure 4: Initial and final graphical results for the second set



(a)



(b)

Figure 5: Initial and final graphical results for the third set

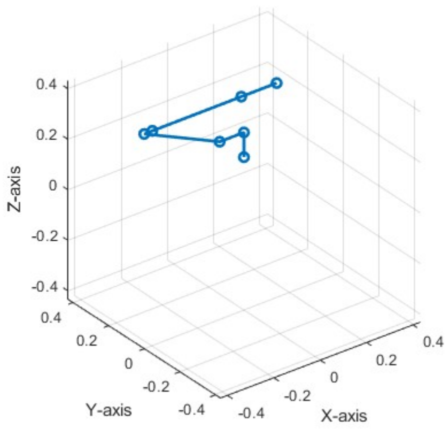
2.6 Q1.5

In this section, we implemented an algorithm to change one joint position at a time. To do so, we used the script implemented in the previous section and slightly modified it by adding another external cycle iterating on the number of links and increasing one joint variable at a time.

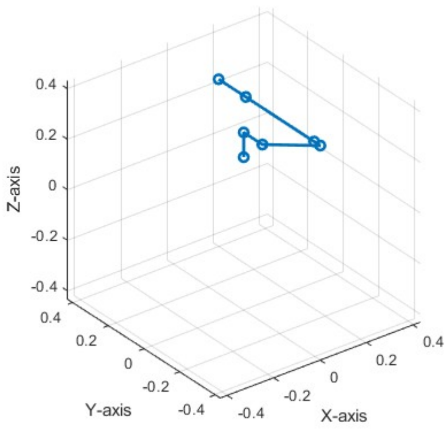
The new script was tested on three sets of initial and final configurations. While the second and third sets were the same as the ones used in the previous section, the first one was defined as:

$$\mathbf{q}_i = [0, 0, 0, 0, 0, 0, 0] \text{ and } \mathbf{q}_f = [\pi/2, \pi/2, \pi/2, \pi/2, \pi/2, \pi/2]$$

This set was chosen to have a clear and easy-understandable graphical representation of each joint position change. For this reason and for the sake of simplicity, only the results obtained by using the first set are shown, from Figure 6 to Figure 9, representing each joint position change.

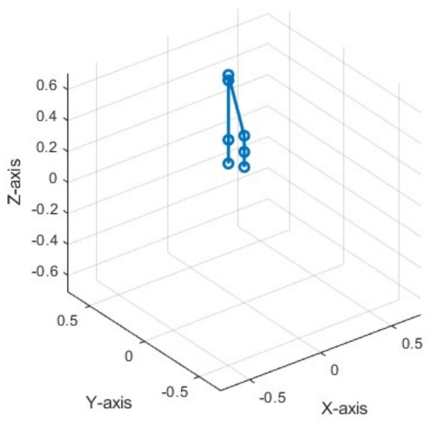


(a)

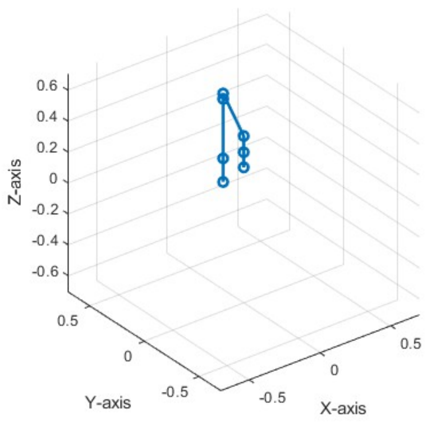


(b)

Figure 6: First and second joints positions change

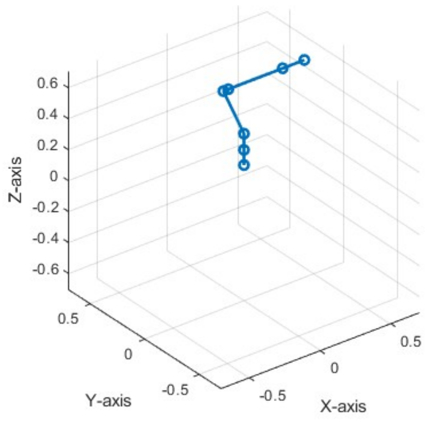


(a)

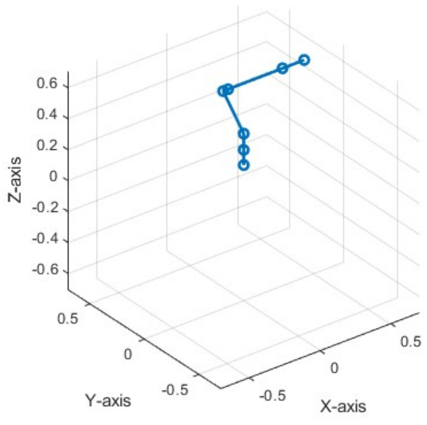


(b)

Figure 7: Third and fourth joints positions change



(a)



(b)

Figure 8: Fifth and sixth joints positions change

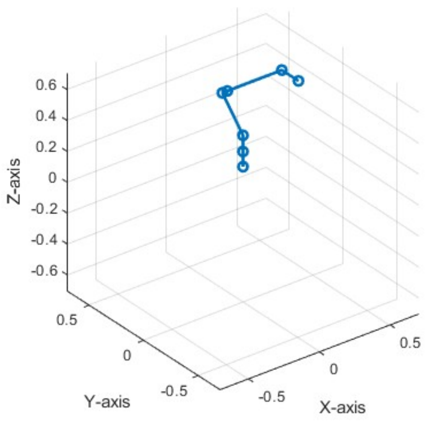


Figure 9: Seventh joint position change