Group Project Report
Group D

Course Title: Group Project
Course Code: EMARO-MSA-2001

Group Members: Ahmed Yesuf Nurye
Taye Tsehaye Alamrew
Dilge Vurmaz
Mete Ogulcan Ozduygu

Instructor: Pawel Maciag

June 2023.

Contents

List of Figures

List of Tables

Project description

The main goal of this project is to:

- Model kinematics of serial 6R robot
- Develop a trajectory planning algorithm that solves a given task (pick and place)
- Test all computations in the simulation environment thoroughly
- Execute given task in hardware

1 Direct kinematics

1.1 Kinematic structure of the robot

The manipulator is a 6 DoF anthropomorphic arm with spherical wrist. The structure is shown in Figure 1.

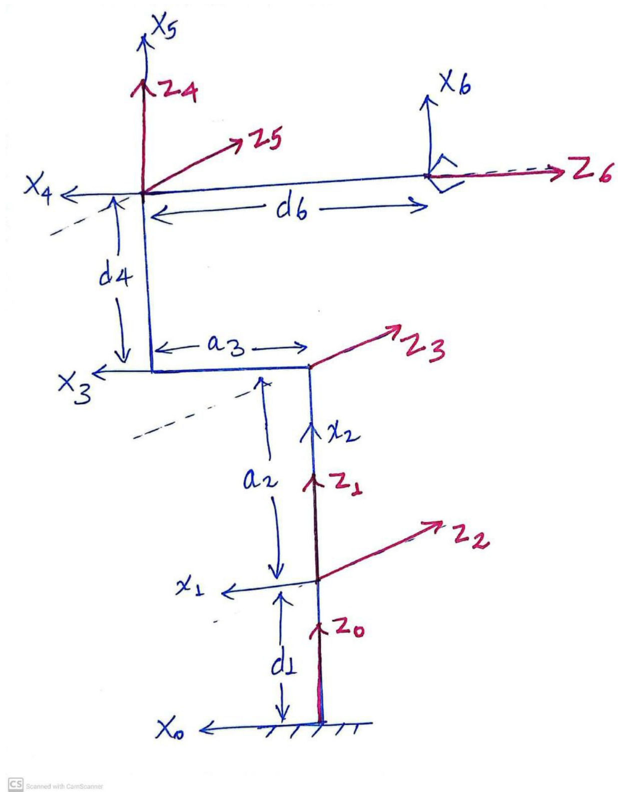

Figure 1: Kinematic structure of the robot

1.2 Denavit-Hartenberg parameters

We used the modified DH convention to perform the axis assignment and the result is shown in Figure 1. The summary of the axis assignment is given below.

1. Assign the z_i axis along the i^{th} axis of rotation.
    - $i = 1$: z_1 pointing up along axis 1 on the plane of the page.
    - $i = 2$: z_2 pointing into the page along axis 2.
    - $i = 3$: z_3 pointing into the page along axis 3.
    - $i = 4$: z_4 pointing up along axis 4 on the plane of the page.
    - $i = 5$: z_5 pointing into the page along axis 5.
    - $i = 6$: z_6 pointing to the right along axis 6 on the plane of the page.
2. Find the common perpendiculars between the z_{i-1} and z_i axes. if those axes intersect the common perpendicular is defined along $z_{i-1} \times z_i$
3. Define the x_i axes. The axis x_{i-1} coincides with the common perpendicular between axes z_{i-1} and z_i and points from z_{i-1} to z_i.

The DH parameters are defined as follows:

- a_{i-1}: The distance between axes z_{i-1} and z_i measured along x_{i-1}
- α_{i-1}: The angle between axes z_{i-1} and z_i measured around x_{i-1}
- d_i: The distance between axes x_{i-1} and x_i measured along z_i
- θ_i: The angle between axes x_{i-1} and x_i measured around z_i

Table 1: DH parameters

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	d_1	θ_1
2	$-\pi/2$	0	0	θ_2
3	0	a_2	0	θ_3
4	$-\pi/2$	a_3	d_4	θ_4
5	$\pi/2$	0	0	θ_5
6	$-\pi/2$	0	d_6	θ_6

DH parameters verification

The DH parameters have been verified using Peter Corke MATLAB robotics toolbox. The plot of the robot at home position is shown in Figure 2.

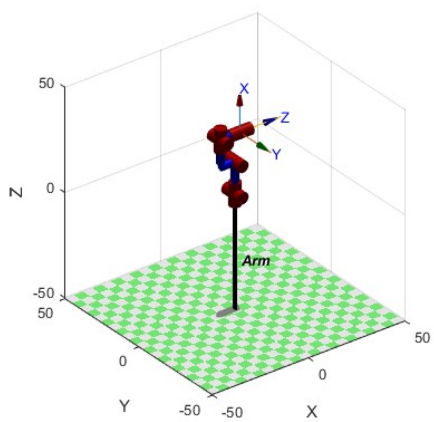

Figure 2: DH parameters verification

1.3 Homogeneous transformation matrices

For the sake of simplicity, we define the following notations:

$$\begin{aligned} c_i &= cos(\theta_i) \\ s_i &= sin(\theta_i) \\ c_{ij} &= cos(\theta_i + \theta_j) \\ s_{ij} &= sin(\theta_i + \theta_j) \end{aligned} \tag{1.1}$$

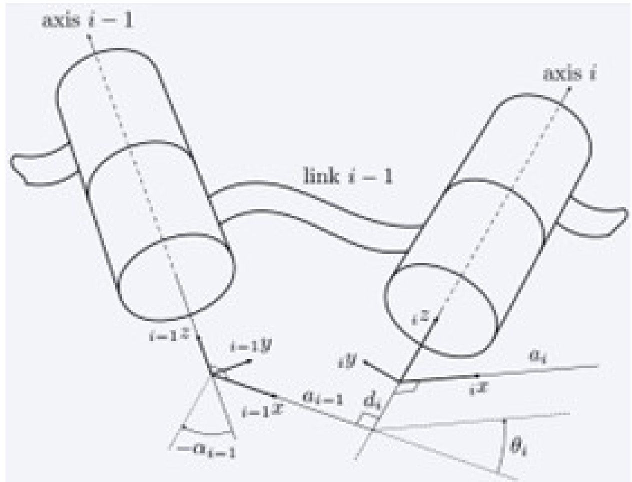

Figure 3: DH frame assignment

From the frame assignment shown in Figure 3, the homogenous transformation matrix between frame i and fram $i-1$ is given by:

$$ {}^{i-1}_{i}T = Rot(x_{i-1}, \alpha_{i-1}) Trans(x_{i-1}, a_{i-1}) Rot(z_i, \theta_i) Trans(z_i, d_i) \tag{1.2} $$

$$ {}^{i-1}_{i}T = \begin{bmatrix} c_i & -s_i & 0 & a_{i-1} \\ s_i c_{\alpha_{i-1}} & c_i c_{\alpha_{i-1}} & -s_{\alpha_{i-1}} & -d_i s_{\alpha_{i-1}} \\ s_i s_{\alpha_{i-1}} & c_i s_{\alpha_{i-1}} & c_{\alpha_{i-1}} & d_i c_{\alpha_{i-1}} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.3} $$

Now, by substituting the DH parameters from Table 1 into Equation 1.3, let's solve the direct kinematics problem.

$$ {}^{0}_{6}T = \prod_{i=1}^{6} {}^{i-1}_{i}T \tag{1.4} $$

$$ {}^{0}_{1}T = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.5} $$

$$ {}^{1}_{2}T = \begin{bmatrix} c_2 & -s_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s_2 & -c_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.6} $$

$$ {}^{0}_{2}T = \begin{bmatrix} c_1 c_2 & -c_1 s_2 & -s_1 & 0 \\ c_2 s_1 & -s_1 s_2 & c_1 & 0 \\ -s_2 & -c_2 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.7} $$

$$ {}^{2}_{3}T = \begin{bmatrix} c_3 & -s_3 & 0 & a_2 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.8} $$

$$
{}^{0}_{3}T = \begin{bmatrix} c_{23}c_1 & -s_{23}c_1 & -s_1 & a_2c_1c_2 \\ c_{23}s_1 & -s_{23}s_1 & c_1 & a_2c_2s_1 \\ -s_{23} & -c_{23} & 0 & d_1 - a_2s_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.9}
$$

$$
{}^{3}_{4}T = \begin{bmatrix} c_4 & -s_4 & 0 & a_3 \\ 0 & 0 & 1 & d_4 \\ -s_4 & -c_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.10}
$$

$$
{}^{0}_{4}T = \begin{bmatrix} s_1s_4 + c_{23}c_1c_4 & c_4s_1 - c_{23}c_1s_4 & -s_{23}c_1 & c_1\left(a_3c_{23} - d_4s_{23} + a_2c_2\right) \\ c_{23}c_4s_1 - c_1s_4 & -c_1c_4 - c_{23}s_1s_4 & -s_{23}s_1 & s_1\left(a_3c_{23} - d_4s_{23} + a_2c_2\right) \\ -s_{23}c_4 & s_{23}s_4 & -c_{23} & d_1 - d_4c_{23} - a_3s_{23} - a_2s_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.11}
$$

$$
{}^{4}_{5}T = \begin{bmatrix} c_5 & -s_5 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ s_5 & c_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.12}
$$

$$
{}^{0}_{5}T = \begin{bmatrix} \begin{matrix} c_5\left(s_1s_4 + c_{23}c_1c_4\right) \\ -\, s_{23}c_1s_5 \end{matrix} & \begin{matrix} -\, s_5\left(s_1s_4 + c_{23}c_1c_4\right) \\ -\, s_{23}c_1c_5 \end{matrix} & c_{23}c_1s_4 - c_4s_1 & c_1\left(a_3c_{23} - d_4s_{23} + a_2c_2\right) \\ \begin{matrix} -\, c_5\left(c_1s_4 - c_{23}c_4s_1\right) \\ -\, s_{23}s_1s_5 \end{matrix} & \begin{matrix} s_5\left(c_1s_4 - c_{23}c_4s_1\right) \\ -\, s_{23}c_5s_1 \end{matrix} & c_1c_4 + c_{23}s_1s_4 & s_1(a_3c_{23} - d_4s_{23} + a_2c_2) \\ -c_{23}s_5 - s_{23}c_4c_5 & s_{23}c_4s_5 - c_{23}c_5 & -s_{23}s_4 & d_1 - d_4c_{23} - a_3s_{23} - a_2s_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.13}
$$

$$
{}^{5}_{6}T = \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ -s_6 & -c_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.14}
$$

$$
{}^{0}_{6}T = \begin{bmatrix} \begin{matrix} c_5c_6\left(s_1s_4 + c_{23}c_1c_4\right) \\ -\, s_{23}c_1c_6s_5 \\ +\, s_6\left(c_4s_1 - c_{23}c_1s_4\right) \end{matrix} & \begin{matrix} -\, c_5s_6\left(s_1s_4 + c_{23}c_1c_4\right) \\ +\, s_{23}c_1s_5s_6 \\ +\, c_6\left(c_4s_1 - c_{23}c_1s_4\right) \end{matrix} & \begin{matrix} -\, s_1s_4s_5 \\ -\, c_{23}c_1c_4s_5 \\ -\, s_{23}c_1c_5 \end{matrix} & \begin{matrix} c_1\left(a_3c_{23} - d_4s_{23} + a_2c_2\right) \\ -\, d_6s_5\left(s_1s_4 + c_{23}c_1c_4\right) \\ + -d_6s_{23}c_1c_5 \end{matrix} \\ \begin{matrix} -\, c_5c_6\left(c_1s_4 - c_{23}c_4s_1\right) \\ -\, s_{23}c_6s_1s_5 \\ -\, s_6\left(c_1c_4 + c_{23}s_1s_4\right) \end{matrix} & \begin{matrix} s_6c_5\left(c_1s_4 - c_{23}c_4s_1\right) \\ +\, s_{23}s_1s_5s_6 \\ -\, c_6\left(c_1c_4 + c_{23}s_1s_4\right) \end{matrix} & \begin{matrix} c_1s_4s_5 \\ -\, c_{23}c_4s_1s_5 \\ -\, s_{23}c_5s_1 \end{matrix} & \begin{matrix} s_1\left(a_3c_{23} - d_4s_{23} + a_2c_2\right) \\ +\, d_6s_5\left(c_1s_4 - c_{23}c_4s_1\right) \\ -\, d_6s_{23}c_5s_1 \end{matrix} \\ \begin{matrix} -\, c_6c_{23}s_5 \\ -\, s_{23}c_4c_5c_6 \\ +\, s_{23}s_4s_6 \end{matrix} & \begin{matrix} c_{23}s_5s_6 \\ +\, s_{23}c_4c_5s_6 \\ +\, s_{23}c_6s_4 \end{matrix} & \begin{matrix} s_{23}c_4s_5 \\ -\, c_{23}c_5 \end{matrix} & \begin{matrix} d_1 - d_4 * c_{23} - a_3 * s_{23} \\ -\, a_2s_2 - d_6c_{23}c_5 \\ +\, d_6s_{23}c_4s_5 \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.15}
$$

2 Inverse Kinematics

Before solving the inverse kinematics problem, we need to find the singular confugurations of the manipulator using the Jacobian matrix. The singular configurations are the configurations where the determinant of the Jacobian matrix is 0

2.1 Singular configurations

Geometric Jacobian

All data necessary to compute the geometric Jacobian can be extracted from the matrices derived while solving the Direct Kinematics Problem.

$$\mathcal{J} = \begin{bmatrix} \mathcal{J}_v \\ \mathcal{J}_\omega \end{bmatrix} \tag{2.1}$$

$${}^0_iT = \begin{bmatrix} {}^0_ir_{11} & {}^0_ir_{12} & {}^0_ir_{13} & {}^0_ir_{i3} & {}^0_iP_x \\ {}^0_ir_{21} & {}^0_ir_{22} & {}^0_ir_{23} & {}^0_ir_{i3} & {}^0_iP_y \\ {}^0_ir_{31} & {}^0_ir_{32} & {}^0_ir_{33} & {}^0_ir_{i3} & {}^0_iP_z \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{2.2}$$

$$\implies {}^0_iP = \begin{bmatrix} {}^0_ip_x \\ {}^0_ip_y \\ {}^0_ip_z \end{bmatrix} \ and \quad {}^0_iz = \begin{bmatrix} {}^0_ir_{13} \\ {}^0_ir_{23} \\ {}^0_ir_{33} \end{bmatrix} \tag{2.3}$$

All joints of the manipulator are revolute. Threfore,

$$\begin{aligned} \mathcal{J}_\omega &= \begin{bmatrix} {}^0_1z & {}^0_2 & z^0_3 & z^0_4 & z^0_5 & z^0_6z \end{bmatrix} \\ \mathcal{J}_v &= \begin{bmatrix} {}^0_1z \times \left({}^0_6p - {}^0_1p\right) & {}^0_2z \times \left({}^0_6p - {}^0_2p\right) & {}^0_3z \times \left({}^0_6p - {}^0_3p\right) & {}^0_4z \times \left({}^0_6p - {}^0_4p\right) & {}^0_5z \times \left({}^0_6p - {}^0_5p\right) & {}^0_6z \times \left({}^0_6p - {}^0_6p\right) \end{bmatrix} \end{aligned} \tag{2.4}$$

The determinant of the Jacobian matrix is computed using MATLAB's symbolic toolbox.

$$\det\left(\mathcal{J}\right) = -a_2s_5\left(a_3s_3 + d_4c_3\right)\left(a_3c_{23} - d_4s_{23} + a_2c_2\right) \tag{2.5}$$

Setting $\det\left(\mathcal{J}\right) = 0$ results the following.

$$\begin{aligned} &a_2 = 0 \\ &s_5 = 0 \\ &a_3s_3 + d_4c_3 = 0 \\ &a_3c_{23} - d_4s_{23} + a_2c_2 = 0 \end{aligned} \tag{2.6}$$

Now that we have established the singular configurations, we can proceed to solve the inverse kinematics problem. Let the desired position and orientation of the end-effector be given by ${}^0_6T_d$ as:

$${}^0_6T_d = \begin{bmatrix} r_{11} & r_{12} & r_{13} & q_x \\ r_{21} & r_{22} & r_{23} & q_y \\ r_{31} & r_{32} & r_{33} & q_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

where q_x, q_y and q_z are the desired end-effector position. We will use Equation 2.8 to solve for the joint angles.

$${}^0_6T = {}^0_6T_d \tag{2.8}$$

We can manipulate Equation 2.8 to isolate each joint angle. To do that we need to find the inveses of some intermediate homogenous transformation matrices.

$$\begin{aligned} T &= \begin{bmatrix} n_x & o_x & a_x & q_x \\ n_y & o_y & a_y & q_y \\ n_z & o_z & a_z & q_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ T^{-1} &= \begin{bmatrix} n_x & n_y & n_z & -n_xq_x - n_yq_y - n_zq_z \\ o_x & o_y & o_z & -o_xq_x - o_yq_y - o_zq_z \\ a_x & a_y & a_z & -a_xq_x - a_yq_y - a_zq_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{2.9}$$

2.2 Solution for θ_1

Equation 1.11, 1.5, and 2.9 are used to isolate θ_1 as follows.

$$\begin{aligned} &{}^1_0T\,{}^0_4T = {}^1_0T\,{}^0_6T_d\,{}^6_5T\,{}^5_4T \\ \implies &{}^1_4T = {}^1_4T_d \end{aligned} \tag{2.10}$$

The matrices are too big and are not written here. The sybolic analysis done using MATLAB is attached together with this report. Equating ${}^1_4T_{2,4} = {}^1_4T_{d2,4}$ results:

$$\begin{aligned} &q_yc_1 - q_xs_1 - d_6r_{23}c_1 + d_6r_{13}s_1 = 0 \\ &c_1\left(q_y - d_6r_{23}\right) = s_1\left(q_x - d_6r_{13}\right) \\ &\frac{s_1}{c_1} = \frac{q_y - d_6r_{23}}{q_x - d_6r_{13}} \end{aligned} \tag{2.11}$$

Therefore,

$$\theta_1 = \operatorname{atan2}\left(q_y - d_6r_{23}, q_x - d_6r_{13}\right) \tag{2.12}$$

To avoid sholder singularity, the following condition must be satiesfied.

$$q_y - d_6r_{23} \neq 0 \wedge q_x - d_6r_{13} \neq 0 \tag{2.13}$$

2.3 Solution for θ_2

We will follow the same procedure as in the previous section. Equation 2.10 will be used to solve for θ_2.

$$\begin{gathered} {}^1_4T_{1,4} = {}^1_4T_{d1,4} \\ a_3c_{23} - d_4s_{23} + a_2c_2 = q_xc_1 + q_ys_1 - d_6r_{13}c_1 - d_6r_{23}s_1 \end{gathered} \tag{2.14}$$

Let $E = q_xc_1 + q_ys_1 - d_6r_{13}c_1 - d_6r_{23}s_1$

$$a_3c_{23} - d_4s_{23} + a_2c_2 = E \tag{2.15}$$

$$\begin{gathered} {}^1_4T_{3,4} = {}^1_4T_{d3,4} \\ -d_4c_{23} - a_3s_{23} - a_2s_2 = q_z - d_1 - d_6r_{33} \end{gathered} \tag{2.16}$$

Let $F = d_6r_{33} + d_1 - q_z$

$$d_4c_{23} + a_3s_{23} + a_2s_2 = F \tag{2.17}$$

From Equation 2.15 and 2.17, we can get the following new sets of equations.

$$\begin{aligned} a_3c_{23} - d_4s_{23} &= E - a_2c_2 \\ d_4c_{23} + a_3s_{23} &= F - a_2s_2 \end{aligned} \tag{2.18}$$

Squaring both sides of Equation 2.18 results:

$$\begin{aligned} a_3^2c_{23}^2 + d_4^2s_{23}^2 - 2a_3c_{23}d_4s_{23} &= E^2 + a_2^2c_2^2 - 2Ea_2c_2 \\ d_4^2c_{23}^2 + a_3^2s_{23}^2 + 2a_3c_{23}d_4s_{23} &= F^2 + a_2^2s_2^2 - 2Fa_2s_2 \end{aligned} \tag{2.19}$$

Adding Equation 2.18 results:

$$a_3^2 + d_4^2 = E^2 + F^2 + a_2^2 - 2Ea_2c_2 - 2Fa_2s_2 \tag{2.20}$$

Moving the unknowns to the left side of Equation 2.20 results:

$$Ec_2 + Fs_2 = \frac{E^2 + F^2 + a_2^2 - a_3^2 - d_4^2}{2a_2} \tag{2.21}$$

Equation 2.21 is obtained by assuming that $a_2 \neq 0$. As can be seen from Equation 2.6, $a_2 = 0$ is one of the singularity configurations. If $a_2 = 0$ then joint 2 and joint 3 will have the same axis of rotation resulting in loss of one degree of freedom.
Let

$$K = \frac{E^2 + F^2 + a_2^2 - a_3^2 - d_4^2}{2a_2} \tag{2.22}$$

Substituting Equation 2.22 into Equation 2.21 results:

$$Ec_2 + Fs_2 = K \tag{2.23}$$

Equation 2.23 can be solved by transforming it into polar form.

$$\begin{aligned} r &= \sqrt{E^2 + F^2} \\ \tan(\phi) &= \frac{F}{E} \Rightarrow \phi = \operatorname{atan2}(F, E) \end{aligned} \tag{2.24}$$

$$\begin{aligned} F &= rsin(\phi) \\ E &= rcos(\phi) \end{aligned} \tag{2.25}$$

Substituting Equation 2.25 into Equation 2.23 results:

$$\begin{aligned} &rcos(\phi)c_2 + rsin(\phi)s_2 = K \\ &rcos(\phi - \theta_2) = K \end{aligned} \tag{2.26}$$

Using Equation 2.24 and Equation 2.26 the value of θ_2 is solved to be:

$$\theta_2 = \operatorname{atan2}(F, E) \pm \arccos\left(\frac{K}{sqrtE^2 + F^2}\right) \tag{2.27}$$

Note, from Equation 2.6 and Equation 2.15 we can see that $E = 0$ is a singular confuguration and we need to take care of it on the programming.

2.4 Solution for θ_3

We can use Equation 1.7, Equation 1.13, Equation 2.7, and Equation 2.9 are used to isolate θ_3.

$$\begin{aligned} {}^2_0T {}^0_5T &= {}^2_0T {}^0_6T_d {}^6_5T \\ {}^2_5T &= {}^2_5T_d \end{aligned} \tag{2.28}$$

$$\begin{aligned} {}^2_5T_{24} &= {}^2_5T_{d24} \\ d_4c_3 + a_3s_3 &= d_1c_2 - q_zc_2 + d_6r_{33}c_2 - q_xc_1s_2 - q_ys_1s_2 + d_6r_13c_1s_2 + d_6r_{23}s_1s_2 \end{aligned} \tag{2.29}$$

By rearranging Equation 2.29 we can express it in terms of variable E and F that we have defined earier.

$$d_4c_3 + a_3s_3 = c_2(d_6r_{33} + d_1 - q_z) + s_2(c_1(d_6r_{13} - q_x) + s1(d_6r_{23} - q_y)) \tag{2.30}$$

Substituting E and F into Equation 2.30 results:

$$d_4c_3 + a_3s_3 = Fc_2 - Es_2 \tag{2.31}$$

We can generate another equation using element $1, 4$

$$\begin{gathered} {}^2_5T_{14} = {}^2_5T_{d14} \\ a_2 + a_3c_3 - d_4s_3 = d_1s_2 - q_zs_2 + d_6r_{33}s_2 + q_xc_1c_2 + q_yc_2s_1 - d_6r_{13}c_1c_2 - d_6r_{23}c_2s_1 \end{gathered} \tag{2.32}$$

By rearranging Equation 2.32 we can express it in terms of variable E and F as well.

$$a_2 + a_3c_3 - d_4s_3 = c_2(c_1(q_x - d_6r_{13}) + s_1(q_y - d_6r_{23})) + s_2(d_6r_{33} + d_1 - q_z) \tag{2.33}$$

Substituting E and F into Equation 2.33 and moving a_2 to the right results:

$$a_3c_3 - d_4s_3 = Ec_2 + Fs_2 - a_2 \tag{2.34}$$

Using the system of equation formed by Equation 2.31 and Equation 2.34 we can solve for θ_3.

$$\begin{aligned} d_4c_3 + a_3s_3 &= Fc_2 - Es_2 \\ a_3c_3 - d_4s_3 &= Ec_2 + Fs_2 - a_2 \end{aligned} \tag{2.35}$$

Using variable change in a_3 and d_4 we can transform Equation 2.35 into polar form and ulimately simplify the system of equation.

$$\begin{aligned} & r = \sqrt{a_3^2 + d_4^2} \\ & \tan(\phi) = \frac{a_3}{d_4} \Rightarrow \phi = \operatorname{atan2}(a_3, d_4) \\ & a_3 = rsin(\phi) \\ & d_4 = rcos(\phi) \end{aligned} \tag{2.36}$$

Substituting Equation 2.36 into Equation 2.35 results:

$$\begin{aligned} rc_\phi c_3 + rs_\phi s_3 &= Fc_2 - Es_2 \\ rs_\phi c_3 - rc_\phi s_3 &= Ec_2 + Fs_2 - a_2 \end{aligned} \tag{2.37}$$

Equation 2.37 can be simplifies to:

$$\begin{aligned} rcos(\phi - \theta_3) &= Fc_2 - Es_2 \\ rsin(\phi - \theta_3) &= Ec_2 + Fs_2 - a_2 \end{aligned} \tag{2.38}$$

$$\tan(\phi - \theta_3) = \frac{Ec_2 + Fs_2 - a_2}{Fc_2 - Es_2} \tag{2.39}$$

Therefore,

$$\theta_3 = \operatorname{atan2}(a_3, d_4) - \operatorname{atan2}(Ec_2 + Fs_2 - a_2, Fc_2 - Es_2) \tag{2.40}$$

One of the singularity confuguration term, $d_4c_3 + a_3s_3$, appears while solving for θ_3. This fact is shown in Equation 2.35. This singularity confuguration occures when the robot is fully stretched and it must as well be taken into account when programming the robot.

2.5 Solution for θ_5

Equation 1.9, Equation 1.13, Equation 2.7, and Equation 2.9 are used to isolate θ_5.

$$\begin{gathered}{}^3_0T\,{}^0_5T = {}^3_0T\,{}^0_6T_d\,{}^6_5T \\ {}^3_5T = {}^3_5T_d\end{gathered} \tag{2.41}$$

$$\begin{gathered}{}^3_5T_{22} = {}^3_5T_{d22} \\ c_5 = r_{33}s_2s_3 - r_{33}c_2c_3 - r_{13}c_1c_2s_3 - r_{13}c_1c_3s_2 - r_{23}c_2s_1s_3 - r_{23}c_3s_1s_2\end{gathered} \tag{2.42}$$

Equation 2.42 can be simplified to:

$$c_5 = -s_{23}(r_{13}c_1 + r_{23}s_1) - r_{33}c_{23} \tag{2.43}$$

Let

$$A = r_{13}c_1 + r_{23}s_1 \tag{2.44}$$

$$c_5 = -r_{33}c_{23} - As_{23} \tag{2.45}$$

Using the trigonometric identity $c_5^2 + s_5^2 = 1$ we can solve for s_5 as follows:

$$s_5 = \pm\sqrt{1 - c_5^2} \tag{2.46}$$

Therefore,

$$\begin{aligned}\theta_5 &= \operatorname{atan2}(s_5, c_5) \\ \theta_5 &= \pm\operatorname{atan2}\left(\sqrt{1 - (-r_{33}c_{23} - As_{23})^2}, -r_{33}c_{23} - As_{23}\right)\end{aligned} \tag{2.47}$$

2.6 Solution for θ_4

Equation 2.41 will be used to solve for θ_4 as well.

$$\begin{gathered}{}^3_5T_{32} = {}^3_5T_{d32} \\ s_4s_5 = r_{23}c_1 - r_{13}s_1\end{gathered} \tag{2.48}$$

Let $B = r_{23}c_1 - r_{13}s_1$

$$s_4s_5 = B \tag{2.49}$$

We can construct another Equation using element $1,2$ of the same matrix as follows.

$$\begin{gathered}{}^3_5T_{12} = {}^3_5T_{d12} \\ -c_4s_5 = r_{13}c_1c_2c_3 - r_{33}c_3s_2 - r_{33}c_2s_3 + r_{23}c_2c_3s_1 - r_{13}c_1s_2s_3 - r_{23}s_1s_2s_3\end{gathered} \tag{2.50}$$

Equation 2.50 can be simplified to:

$$\begin{aligned}-c_4s_5 &= (r_{13}c_1 + r_{23}s_1)c_{23} - r_{33}s_{23} \\ \Rightarrow c_4s_5 &= r_{33}s_{23} - Ac_{23}\end{aligned} \tag{2.51}$$

From Equation 2.49 and Equation 2.51 we can construct Equation 2.52.

$$\begin{aligned}s_4s_5 &= B \\ c_4s_5 &= r_{33}s_{23} - Ac_{23}\end{aligned} \tag{2.52}$$

Therefore, assuming that $s_5 \neq 0$, we can solve for θ_4 as follows:

$$\theta_4 = \operatorname{atan2}(B, r_{33}s_{23} - Ac_{23}) \tag{2.53}$$

Note that $s_5 = 0$ is one of the singularity configurations. We will solve for θ_4 in this case later.

2.7 Solution for θ_6

We can isolate θ_6 using Equation 1.13, Equation 1.14, Equation 2.7, and Equation 2.9.

$$\begin{aligned} {}^5_6T &= {}^5_0T {}^0_6T_d \\ {}^5_6T &= {}^5_6T_d \end{aligned} \tag{2.54}$$

$$\begin{aligned} {}^5_6T_{11} &= {}^5_6T_{d11} \\ c_6 = r_{11}c_5s_1s_4 - r_{31}s_{23}c_4c_5 &- r_{11}s_{23}c_1s_5 - r_{21}s_{23}s_1s_5 - r_{21}c_1c_5s_4 \\ &- r_{31}c_{23}s_5 + r_{21}c_{23}c_4c_5s_1 + r_{11}c_{23}c_1c_4c_5 \end{aligned} \tag{2.55}$$

Collecting like terms of Equation 2.55 results:

$$\begin{aligned} c_6 =& r_{11}\left(c_5s_1s_4 + c_1(c_2(c_3c_4c_5 - s_3s_5) - s_2(c_4c_5s_3 + c_3s_5))\right) \\ &- r_{21}\left(c_4c_5s_1s_2s_3 + c_1c_5s_4 + c_3s_1s_2s_5 + c_2s_1(s_3s_5 - c_3c_4c_5)\right) \\ &+ r_{31}\left(s_3(s_2s_5 - c_2c_4c_5) - c_3(c_4c_5s_2 + c_2s_5)\right) \end{aligned} \tag{2.56}$$

Using element $1,2$ of the same matrix we can construct another equation as follows:

$$\begin{aligned} {}^5_6T_{12} &= {}^5_6T_{d12} \\ -s_6 = r_{12}c_5s_1s_4 - r_{32}s_{23}c_4c_5 &- r_{12}s_{23}c_1s_5 - r_{22}s_{23}s_1s_5 - r_{22}c_1c_5s_4 \\ &- r_{32}c_{23}s_5 + r_{22}c_{23}c_4c_5s_1 + r_{12}c_{23}c_1c_4c_5 \end{aligned} \tag{2.57}$$

Collecting like terms of Equation 2.57 results:

$$\begin{aligned} -s_6 =& r_{12}(c_5s_1s_4 + c_1(c_2(c_3c_4c_5 - s_3s_5) - s_2(c_4c_5s_3 + c_3s_5))) \\ &- r_{22}(c_4c_5s_1s_2s_3 + c_1c_5s_4 + c_3s_1s_2s_5 + c_2s_1(s_3s_5 - c_3c_4c_5)) \\ &+ r_32(s_3(s_2s_5 - c_2c_4c_5) - c_3(c_4c_5s_2 + c_2s_5)) \end{aligned} \tag{2.58}$$

Therefore, using Equation 2.56 and Equation 2.58 θ_6 can be solved as follows:

$$\theta_6 = \text{atan2}(s_6, c_6) \tag{2.59}$$

Now let's solve for θ_4 and θ_6 when $s_5 = 0$, i.e., when there is a wrist singularity. This condition results when z_4 and z_6 coincides. Resulting in cancellation of motion of joint 4 and joint 6. Equation 1.5, Equation 1.15, Equation 2.7, and Equation 2.9 are used to isolate θ_4 and θ_6.

$$\begin{aligned} {}^1_0T {}^0_6T &= {}^1_0T {}^0_6T_d \\ {}^1_6T &= {}^1_6T_d \end{aligned} \tag{2.60}$$

$$\begin{aligned} {}^1_6T_{21} &= {}^1_6T_{d21} \\ -c_4s_6 - c_5c_6s_4 &= r_{21}c_1 - r_{11}s_1 \end{aligned} \tag{2.61}$$

Using element $2,2$ of the same matrix we can construct another equation as follows:

$$\begin{aligned} {}^1_6T_{22} &= {}^1_6T_{d22} \\ c_5s_4s_6 - c_4c_6 &= r_{22}c_1 - r_{12}s_1 \end{aligned} \tag{2.62}$$

Simplifying Equation 2.61 and Equation 2.62 results Equation 2.63.

$$\begin{aligned} -\sin(\theta_6 \pm \theta_4) &= r_{21}c_1 - r_{11}s_1 \\ -\cos(\theta_6 \pm \theta_4) &= r_{22}c_1 - r_{12}s_1 \end{aligned} \tag{2.63}$$

In wrist singularity confuguration θ_4 and θ_6 produces a motion that will effectively cancel each other. Therefore, it is logical to fix the value of one of those angles to its previous value ans solve for the other. By fixing θ_4 to its previous value, θ_{4prev}, we can solve for θ_6 as follows:

$$\begin{aligned} \theta_4 &= \theta_{4prev} \\ \theta_6 &= \text{atan2}\left(-(r_{21}c_1 - r_{11}s_1), -(r_{22}c_1 - r_{12}s_1)\right) \mp \theta_{4prev} \end{aligned} \tag{2.64}$$

3 kinematics tests

Both forward and inverse kinematics are tested against Peter Corke's Robotics Toolbox MATLAB, use script *test_kinematics*.
IMPORTANT: for the test it is required that Peter Corke's Robotics Toolbox is installed. This is because we used the result from this toolbox as the ground truth.

Usage:
```
>> test_kinematics
```

3.1 Direct kinematics test

Ten randomly generated joint angles are used to test the direct kinematics. We have used tolerance of $1e\_6$, and for this tolerance all the tests are passed. The test result is shown in Figure 4.

```
>> test_kinematics
Direct Kinematics 1 passed.
Direct Kinematics 2 passed.
Direct Kinematics 3 passed.
Direct Kinematics 4 passed.
Direct Kinematics 5 passed.
Direct Kinematics 6 passed.
Direct Kinematics 7 passed.
Direct Kinematics 8 passed.
Direct Kinematics 9 passed.
Direct Kinematics 10 passed.
All direct kinematics tests passed.
```

Figure 4: Direct kinematics test

3.2 Inverse kinematics test

Similarly, ten homogeneous transformation matrices generated from 10 randomly generated joint angles are used to test the inverse kinematics. After calculating the inverse kinematics we recalculate the homogeneous transformation matrix and compare it with the original homogeneous transformation matrix. The same tolerance of $1e\_6$ is used and all the tests are passed. The test result is shown in Figure 5.

```
Inverse Kinematics Test 1 passed.
Inverse Kinematics Test 2 passed.
Inverse Kinematics Test 3 passed.
Inverse Kinematics Test 4 passed.
Inverse Kinematics Test 5 passed.
Inverse Kinematics Test 6 passed.
Inverse Kinematics Test 7 passed.
Inverse Kinematics Test 8 passed.
Inverse Kinematics Test 9 passed.
Inverse Kinematics Test 10 passed.
All inverse kinematics tests passed.
```

Figure 5: Inverse kinematics test

4 Trajectory planning

IMPORTANT: to simulate the trajectory it is required that Peter Corke's Robotics Toolbox is installed.

Usage:

```
>> test_trajectory('j', true)
>> test_trajectory('t', true)
```

Problem: to find a trajectory connecting initial and final points.
There are different methods for trajectory jeneration. Some of them are:

- LSPB
- Polynomial (Qubic, Quintic)

In this section we will perform the trajectory planning of simple pick and place task using LSPB. We used both joint space and task space trajectory planning as shown in Figure 6 below.

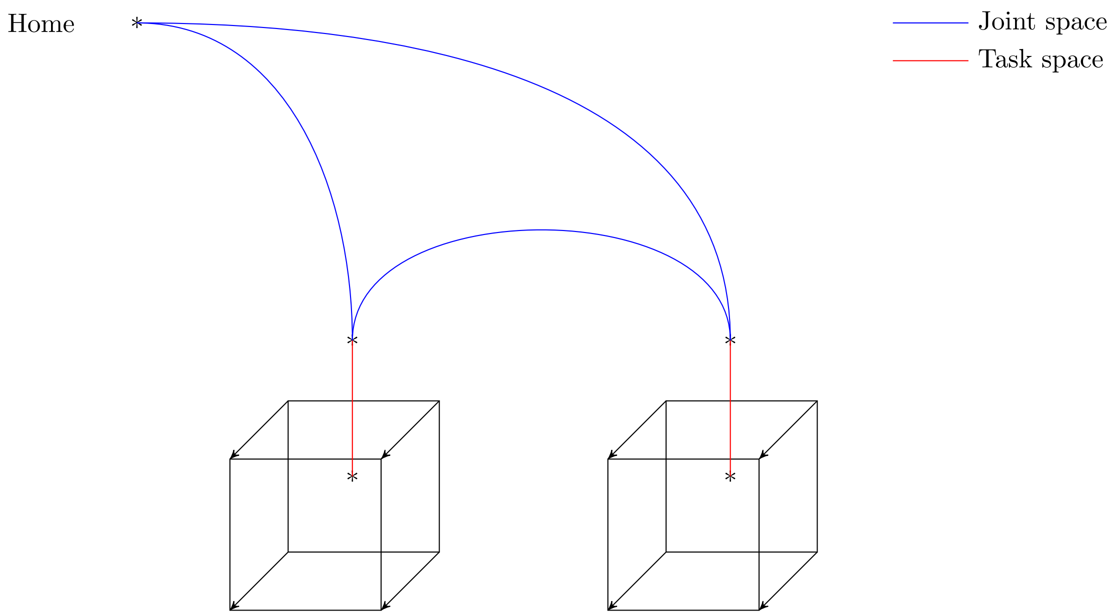

Figure 6: Trajectory planning

The task definition can easly be seen from Figure 6. In the next sub-sections we are going to see the implementation of task space and joint space trajectory planning.

4.1 Joint space trajectory planning using LSPB

What is LSPB?
In short, Linear Segments with Parabolic Blends trajectory, is a symmetric trajectory with trapezoidal shape of velocity profile, with constant velocity in central part of the path.

- This type of trajectory is appropriate when a constant velocity is desired along a portion of the path.
- The LSPB trajectory is such that the velocity is initially ramped up to its desired value and then ramped down when it approaches the goal position.

- To achieve this we specify the desired trajectory in three parts.
- For $t \in [t_0 \quad t_b]$ and $t \in [t_f - t_b \quad t_f]$ the trajectory is described by a quadratic polynomial.
- For $t \in [t_b \quad t_f - t_b]$ a constant velocity, v, is taken.
- The blend time t_b is chosen such that the position curve is symmetric.

The LSPB trajectory is defined by the following equation, assuming $t_0 = 0$:

$$\begin{cases} \theta(t) = \theta_0 + \frac{\alpha}{2}t^2 & 0 \leq t \leq t_b \\ \theta(t) = \frac{\theta_0 + \theta_f - vt_f}{2} + vt_f & t_b < t \leq t_f - t_b \\ \theta(t) = \theta_f - \frac{\alpha t_f^2}{2} + \alpha t_f - \frac{\alpha}{2}t^2 & t_f - t_b < t \leq t_f \end{cases} \tag{4.1}$$

4.1.1 Constraints of LSPB

While driving Equation 4.1 it is assumed that $0 \leq t_b \leq \frac{t_f}{2}$, Other wise the trajectory can't be achieved. From this condition we can derive the following constraints:

$$\begin{aligned} \frac{q_f - q_0}{v} < t_f \leq \frac{2(q_f - q_0)}{v} \\ \implies \frac{q_f - q_0}{t_f} < v \leq \frac{2(q_f - q_0)}{t_f} \end{aligned} \tag{4.2}$$

If $t_b = \frac{t_f}{2}$ then the trajectory reduces to mimimum time trajectory (Bang-Bang trajectory). We have used equation 4.1 to define the trajectory in joint space and the result for sample trajectory between the following initial and final joint angles is shown in Figure 7 and 8.

For a single joint: $q_0 = 0 \, and \, q_f = \pi$

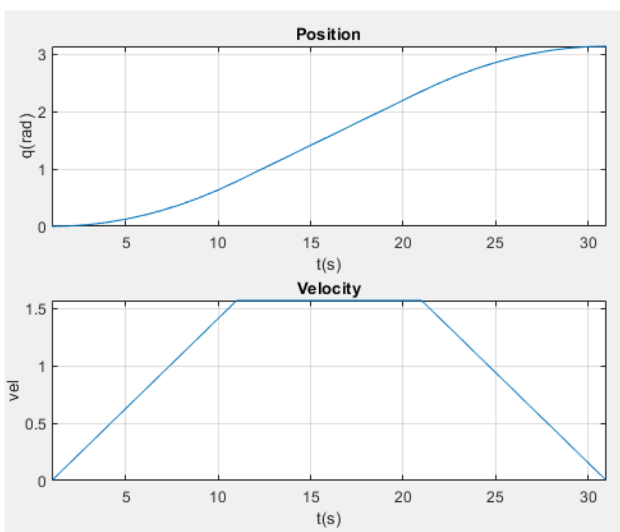

Figure 7: LSPB trajectory for a single joint

For all the joints:

$$\theta_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$
$$\theta_f = \begin{bmatrix} \pi & \pi/2 & \pi/4 & \pi/6 & \pi/8 & \pi/10 \end{bmatrix}$$

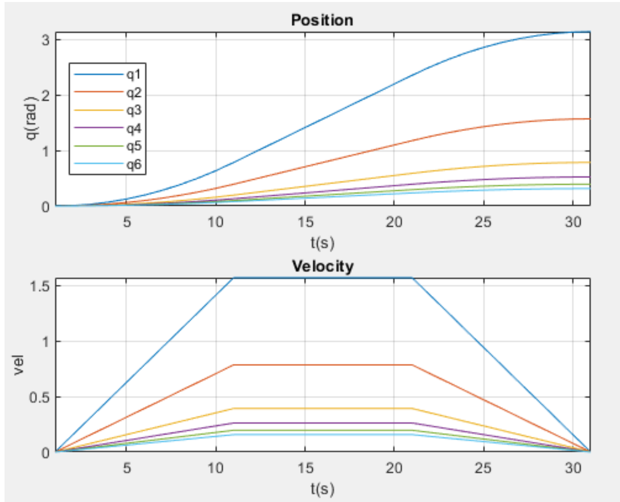

Figure 8: Synchronized LSPB trajectory

For synchronizing the trajectories we have used the maximum t_f of all the joints. This computation of time is implemented in the function *get_time*.

4.2 Task space trajectory planning

For the task space trajectory planning we have followed the following steps:

- Express the end-effector pose in terms of roll, pitch and yaw angles.
- Use linear interpolation to compute the trajectory for each of the roll, pitch and yaw angles as well as the X, Y, Z coordinates of the end effector (position).
- Convert each roll, pitch and yaw angles back to the rotation matrix.
- Generate the homogeneous transformation matrix using the rotation matrix and the position.
- Perform inverse kinematics to compute the joint angles for each of the homogeneous transformation matrix.

Note, if we don't generate enough points in the interpolation we can use LSPB to get more joint angles between each consequetive points. The end-effector pose for straight line task space trajectory is shown in Figure 9.

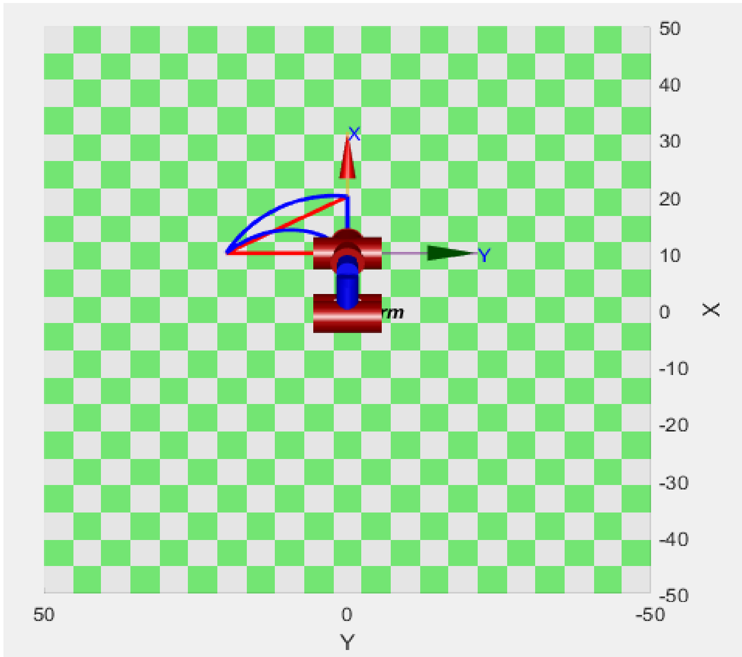

Figure 9: Task space Vs. Joint space trajectory, end-effector pose for triangular profile.

Finally the pick and place task (contains both joint and task space trajectories) trajectory end-effector pose is shown in Figure 10.

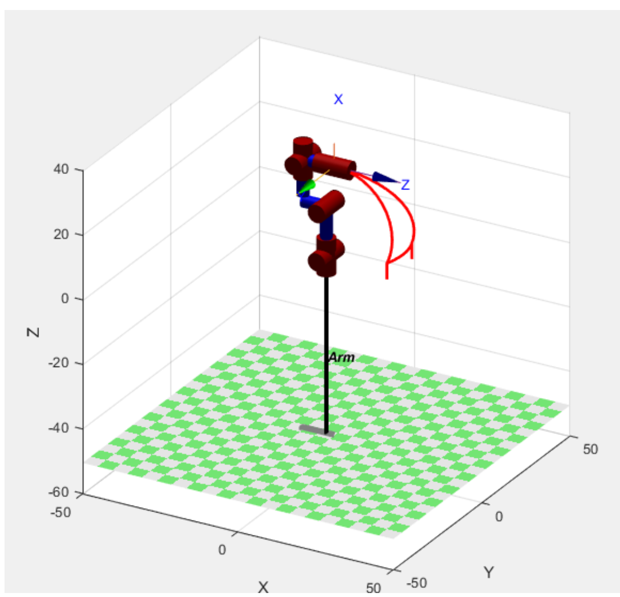

Figure 10: Pick and place task

5 Hardware Implementation

This section presents the step by step implementation of simple pick and place task in the real robot.

Usage:

```
>> test_pick_and_place
```

5.1 Hardware Setup

At this stage we adjust the manipulator to the desired home position and recorded the joint angles corresponding to this home pose using Dynamixel wizard. We also recorded the maximum and minimum joint angles for each motor using Dynamixel wizard. The range of motion and velocity for each motor is shown in Table 2.

Table 2: Range of motion and velocity for each motor

Motor Type	**Range of Motion**	**Velocity**
Ax	$[0-1023] \rightarrow [0-300^0]$	$[0-1023] \rightarrow [0-114rpm]$
Mx	$[0-4094] \rightarrow [0-360^0]$	$[0-1023] \rightarrow [0-116.62rpm]$

5.2 Brief Description of the Code

The functions used in the hardware implementation are listed below,

- *joint2motor*: converts joint angles from radian to manipulator units.
- *traj_pos_joint2motor*: converts matrix of joint angles from radian to manipulator units by calling *joint2motor* in a loop.
- *vel_rpm2dxl*: converts joint velocity from *rpm* to manipulator units.
- *traj_vel_joint2motor*: converts matrix of joint velocities from *rpm* to manipulator units, by calling *vel_rpm2dxl* in a loop.
- *open_port*: establish communication by opening the port and setting the baud rate, specify the port from devide manager here.
- *write_data*: sends joint angles and velocities to all the motors simultaneously.
- *sync_send*: sends matrix of joint angles and velocities by calling *write_data* in a loop.
- *gripper*: Opens and closes the gripper as required.
- *close_port*: for terminating session by closing the port.
- *enable_torque*: enables the torque of the manipulator motors, and also used to see if all the motors are working properly.
- *disable_torque*: disables the torque of the manipulator motors.

The script *test_pick_and_place* uses all the above functions to perform the pick and place task.

5.3 Pick and place task graph (Basic Algorithm)

The task execution process is sequential and straightforward and is shown in Figure 11.

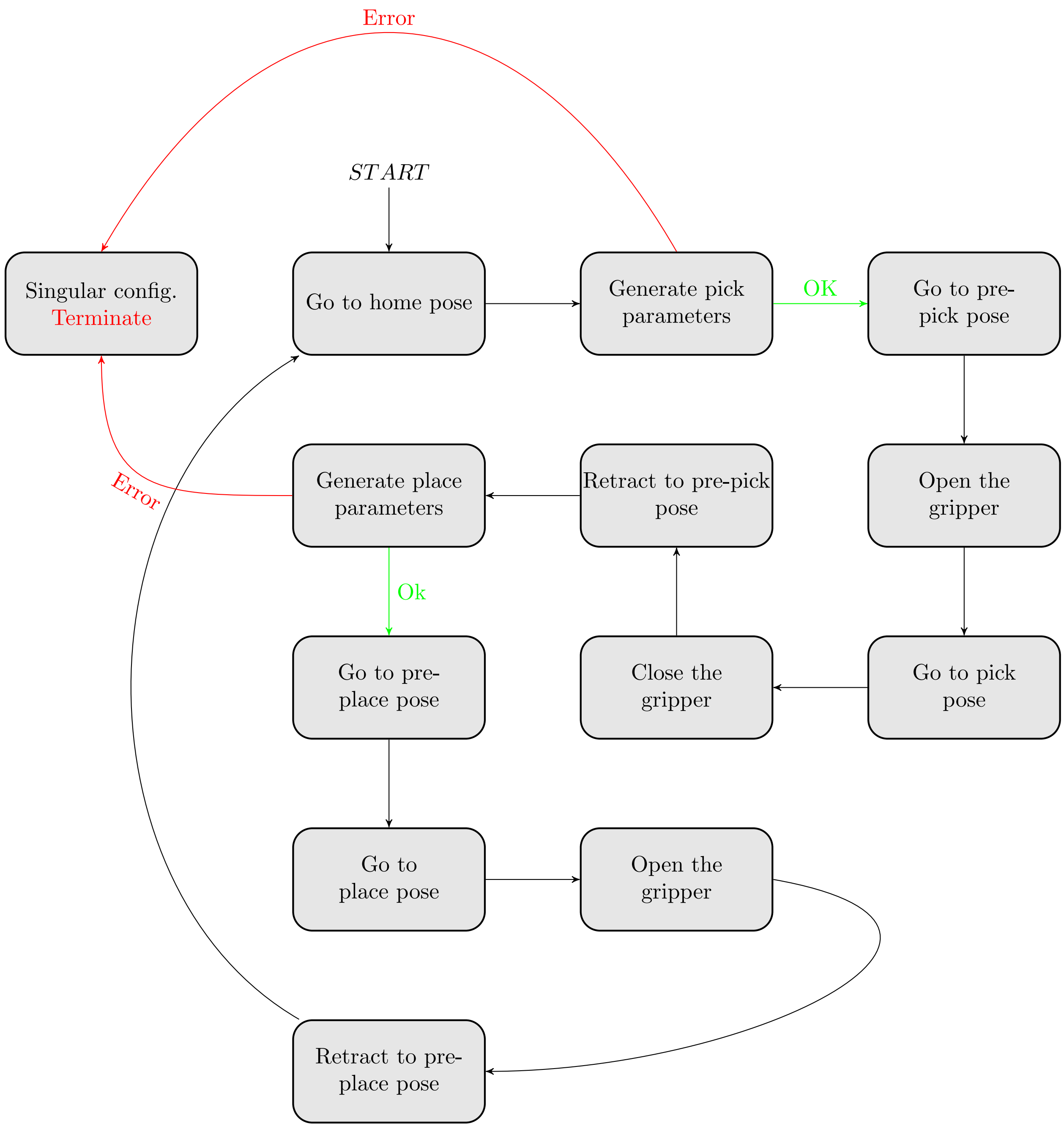

Figure 11: Task graph

Please note that, an error is generated in the *inv_kine* function if a pose that results in singular confuguration is provided.

Please also note that the first initialization step before performing any task is to open the port, set baud rate, and enable torque.
The last steps are to disable the torque and close the port.

Description of the task graph:

1. Go to home pose: Initially the manipulator could be at any pose. Therefore, using the home pose joint angles recorded previously, we should go to the home pose first.
2. Generate pick parameters: Here we generate the trajectory to got to the pre-pick and pick poses. While performing the inverse kinematics we could encounter an error if singular configuration is provided.
3. Go to pre-pick pose: Done by sending the trajectory that connects home pose and pre-pick pose.
4. Open the gripper.
5. Go to pick pose.
6. Close the gripper.
7. Retract to pre-pick pose.
8. Generate place parameters: done in the same way as pick parameters are generated.
9. Go to pre-place pose: done in the same way as pre-pick pose is reached.
10. Go to place pose.
11. Open the gripper.
12. Retract to pre-place pose.

Finally, the pick and place execution can be put in a loop to perform the task multiple times.

6 Conclusion

In this project we have analyzed and tested the kinematics of 6 dof anthropomorphic arm with spherical wrist, done joint space trajectory planning using LSPB and task space trajectory planning using linear interpolation. Finally, we have simulated our work using Peter Corke's Robotics Toolbox and shown the implementation in the real robot.