ENPM 662- INTRODUCTION TO ROBOT MODELING

PROJECT 2

TECHNICAL REPORT

ON

CNC TENDING ROBOT

GROUP 2

Suhas Nagaraj
UID : 119505373
Directory ID: suhas99

Swaraj M Rao
UID : 120127007
Directory ID: swarajmr

Contents

ABSTRACT

This project report presents the Simulation of ABB IRB 1600 Robot, on Gazebo, for pick and place application in an industrial environment. The project mainly focuses on the forward and inverse kinematics of the robot.

Section I of the report focuses on the introduction, description of the robot, the application, the description of the CAD model and the degrees of freedom of the robot used for this project.

Section II gives a detailed description of the forward kinematics using a systematic approach based on the Denavit-Hartenberg convention and the use of homogeneous transformations and their validations.

Section III discusses concepts of Inverse Kinematics, methods followed, the validation and its usage/application in our project.

Section IV discusses the assumptions considered, problems faced, lessons learned, future work, conclusions, and references.

INTRODUCTION

The seamless integration of robotic systems has emerged as a transformative and essential aspect within industrial environments, driven by an unwavering commitment to achieving heightened efficiency and unparalleled precision. These robotic systems have become integral players in a diverse array of tasks, ranging from the intricacies of pick and place operations, welding, and painting to the precision-demanding processes of assembling, disassembly, packaging and labeling, palletizing, as well as intricate product inspection and testing protocols. The intrinsic value of robots lies in their remarkable attributes, notably their elevated endurance, speed, and precision, rendering them indispensable for industries seeking not only increased productivity but also a reduction in errors and the streamlining of complex manufacturing procedures.

Diversity in industrial applications has led to the development of various types of robots, each designed to meet specific manufacturing requirements. Among these, articulated robots, characterized by their rotary joints mimicking the flexibility of a human arm, emerge as the most prevalent and versatile type for a broad spectrum of industrial applications. However, the industrial landscape also embraces Cartesian coordinate robots, operating within a three-dimensional space, spherical coordinate robots offering extensive range of motion, SCARA robots designed for high-precision tasks, Delta robots renowned for high-speed capabilities, cylindrical robots combining rotary and linear movements, collaborative robots (Cobots) promoting safe human-robot collaboration, and parallel robots utilizing multiple actuators for enhanced speed and precision.

The performance of these industrial robots is contingent upon a delicate interplay of several critical factors. Accuracy, ensuring meticulous alignment with intended positions, stands as a paramount requirement for tasks demanding surgical precision, serving as a bulwark against defects in the final products. Precision, on the other hand, guarantees the consistent repeatability of movements, a factor of utmost importance in maintaining uniformity and quality control in the realm of mass production. Speed emerges as a critical factor, particularly in high-volume manufacturing scenarios, as it accelerates cycle times and contributes significantly to overall operational efficiency. However, the need for speed must be meticulously managed to avoid compromising precision, emphasizing the importance of a well-calibrated balance.

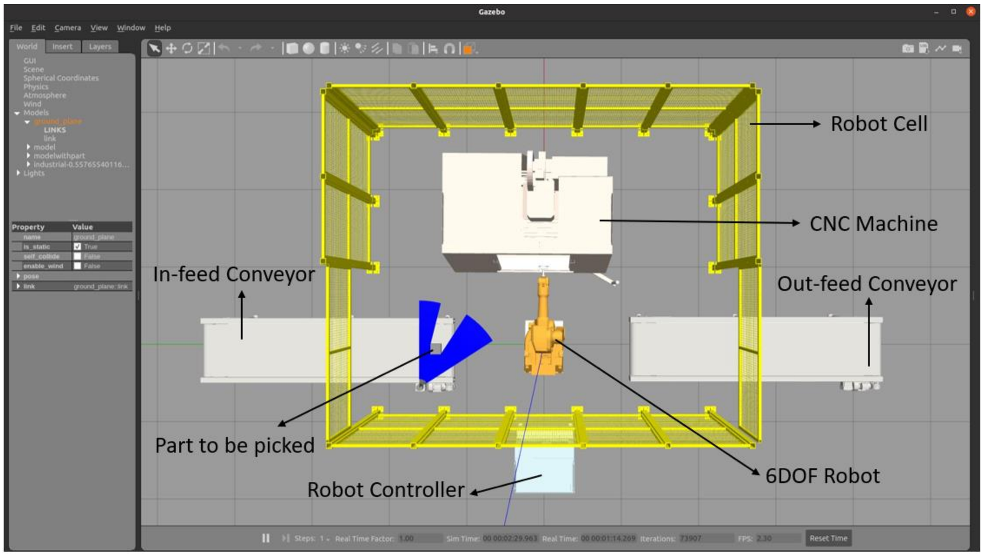
Moreover, the concept of cycle time, representing the total duration of a robot's operational cycle, emerges as a pivotal metric in gauging productivity. Shorter cycle times not only optimize resource utilization but also contribute to increased production output, thereby elevating the overall efficiency of industrial operations. In essence, the symbiotic relationship between accuracy, precision, speed, and cycle time underscores the dynamic and multifaceted nature of industrial robotics, playing a central role in shaping the landscape of modern manufacturing.

APPLICATION

The robot is employed for pick and place operations in the context of loading and unloading materials onto a machining center. The system layout used in the simulation for this project features two conveyors, with one designated for infeed of raw materials, typically in the form of billets, and the other for transporting machined components away from the machine. The workflow involves the robot picking up raw materials from the infeed conveyor and placing them onto the machine for machining. Subsequently, the robot retrieves the machined components and deposits them onto the outfeed conveyor. This process is orchestrated through precise programming to ensure accurate and efficient movements.

The same robot can be configured to perform a wide variety of applications ranging from industrial applications, healthcare, agriculture, military, and defense, and much more. In industries, the articulated robot is used for palletizing, pick and place, bin-picking, welding, assembly operations, and machine tending. Automated machine tending enables the loading and unloading of raw materials into machines. Again, this robot's enhanced reach and dexterity allow it to excel here. Modern machine tending robots can do everything from loading the part, opening, and closing the door, to even running the specified program by selecting it on the HMI(Human Machine Interface).

The implemented simulation layout in gazebo is as shown in the figure below:



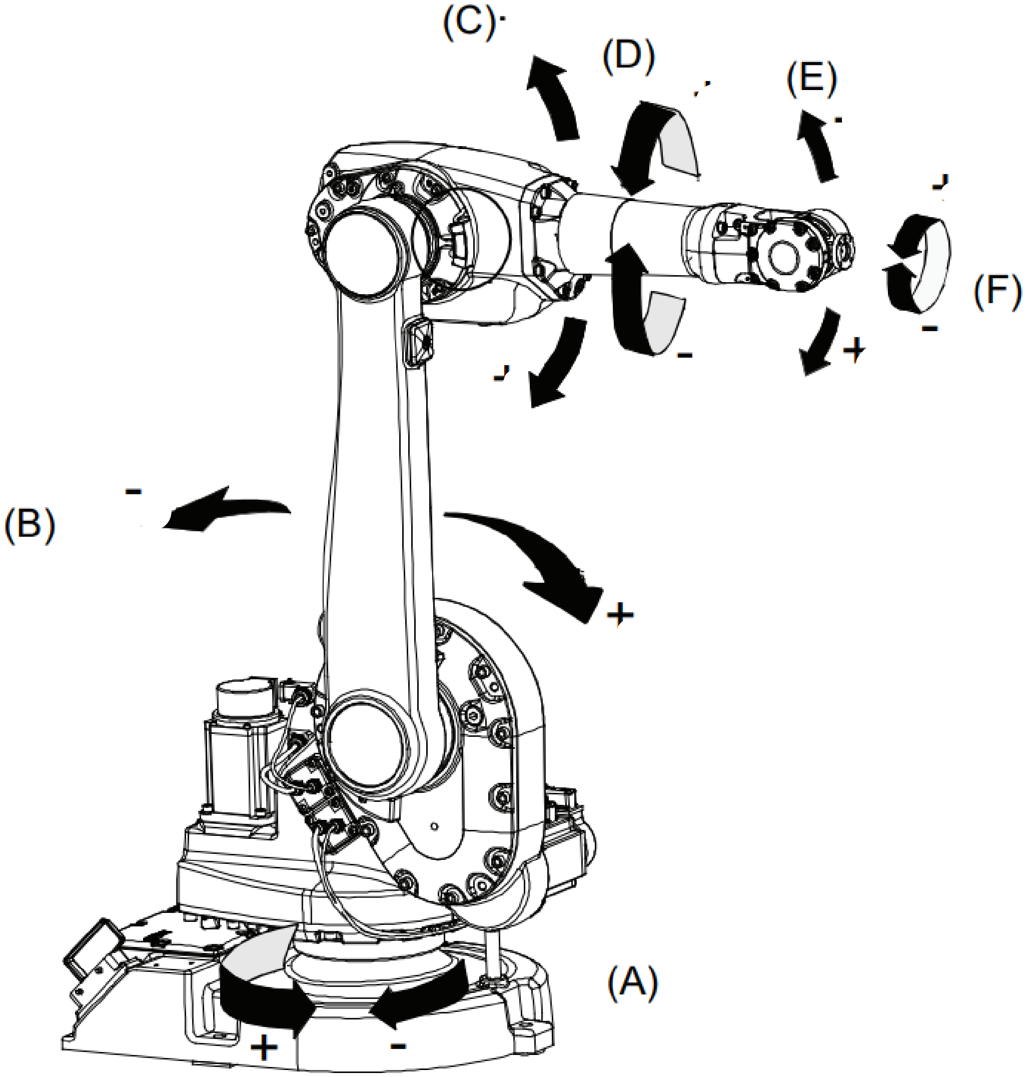
Property	Value
name	ground_plane
is_static	☑ True
self_collide	□ False
enable_wind	□ False
pose	
link	ground_plane::link

Steps: 1 Real Time Factor: 1.00 Sim Time: 00 00:02:29.963 Real Time: 00 00:01:14.269 Iterations: 73907 FPS: 2.30

ROBOT TYPE and DOF

The robot system used for the application is a 6 DOF robot manipulator with a vacuum gripper mounted as the end effector. The manufacturer of the robot is ABB and the robot model selected is IRB 1600 – 10/1.45, where IRB stands for Industrial Robot series and 1600 specifies the size of the robot arms and the model number. The robot is an articulated type of robot having six rotary joints and serial linkages.

Degrees of Freedom (DOF) typically refers to the number of movable joints of the robot. The IRB 1600 robot has six degrees of freedom. The figure shows all the six axes of the robot (A, B, C, D, E, F).



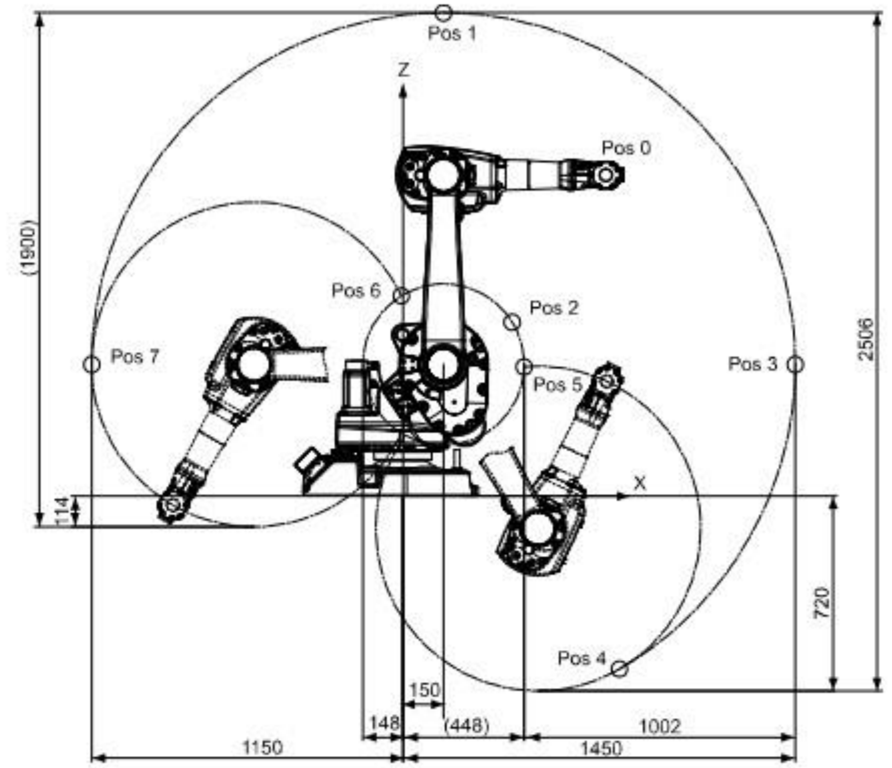
The description of the robot axes, the type of motion and the range of movement to avoid self-collision as provided by the manufacturer in the product manual is as follows.

Axis	Type of Motion	Range of Movement
(A) / 1	Rotation Motion	+180° to -180°
(B) / 2	Arm Motion	+120° to -90°
(C) / 3	Arm Motion	+65° to -245°
(D) / 4	Rotation Motion	+200° to -200°
(E) / 5	Bend Motion	+115° to -115°
(F) / 6	Turn Motion	+288° to -288°

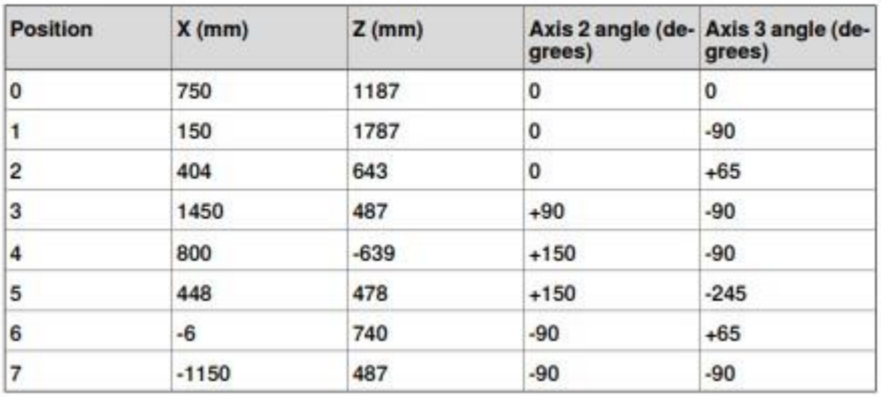
WORKSPACE STUDY

Workspace study is carried out to ensure successful integration of robots into a specific work environment. Some of the key reasons why this study is conducted are for task optimization, workspace configuration, efficiency and productivity improvement and risk assessment. The figure below represents the workspace configuration of the ABB IRB 1600-10 1.45m robot which has been used in this project.

Positions at wrist center 1.45 m reach



xx1000000915

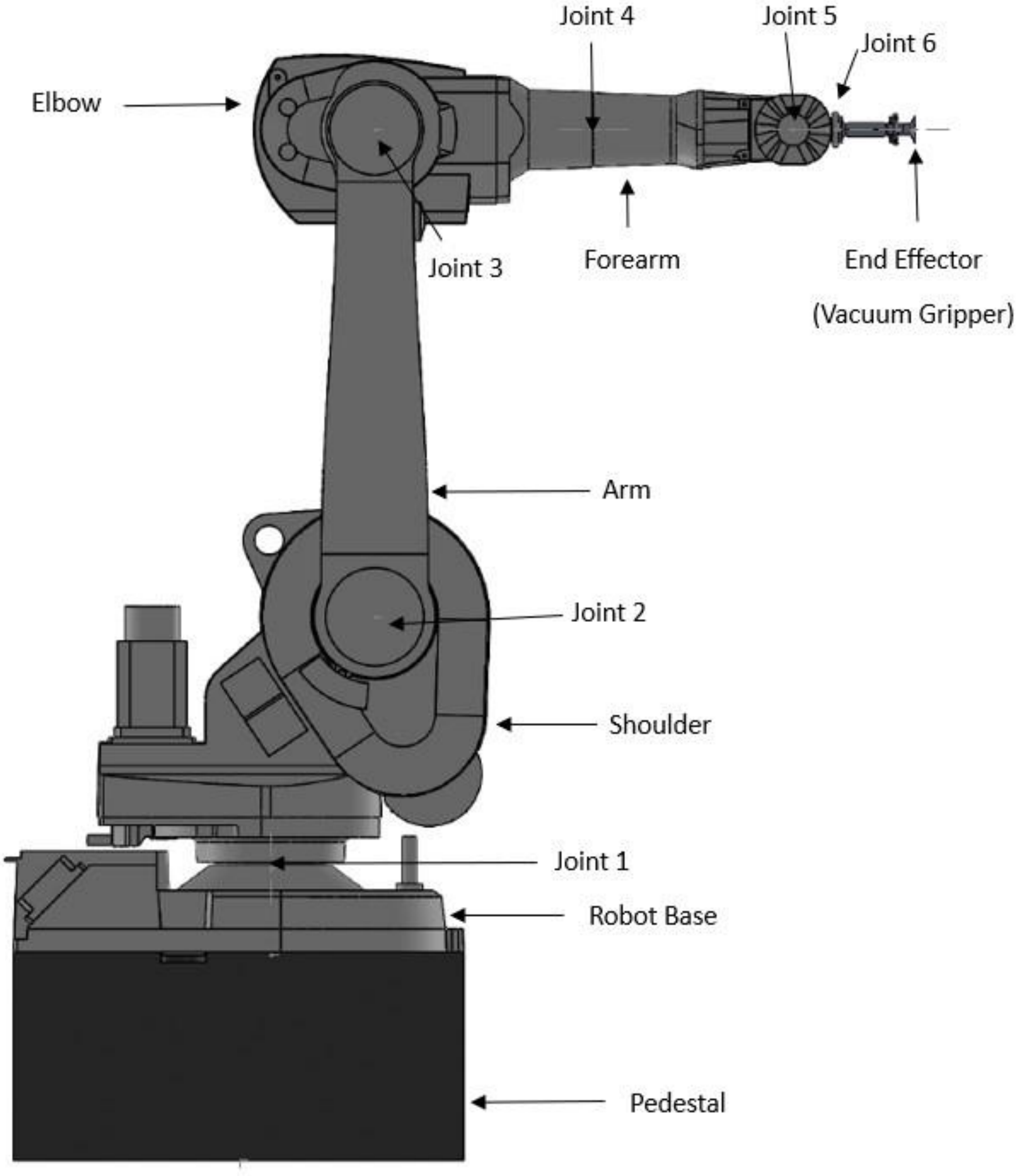


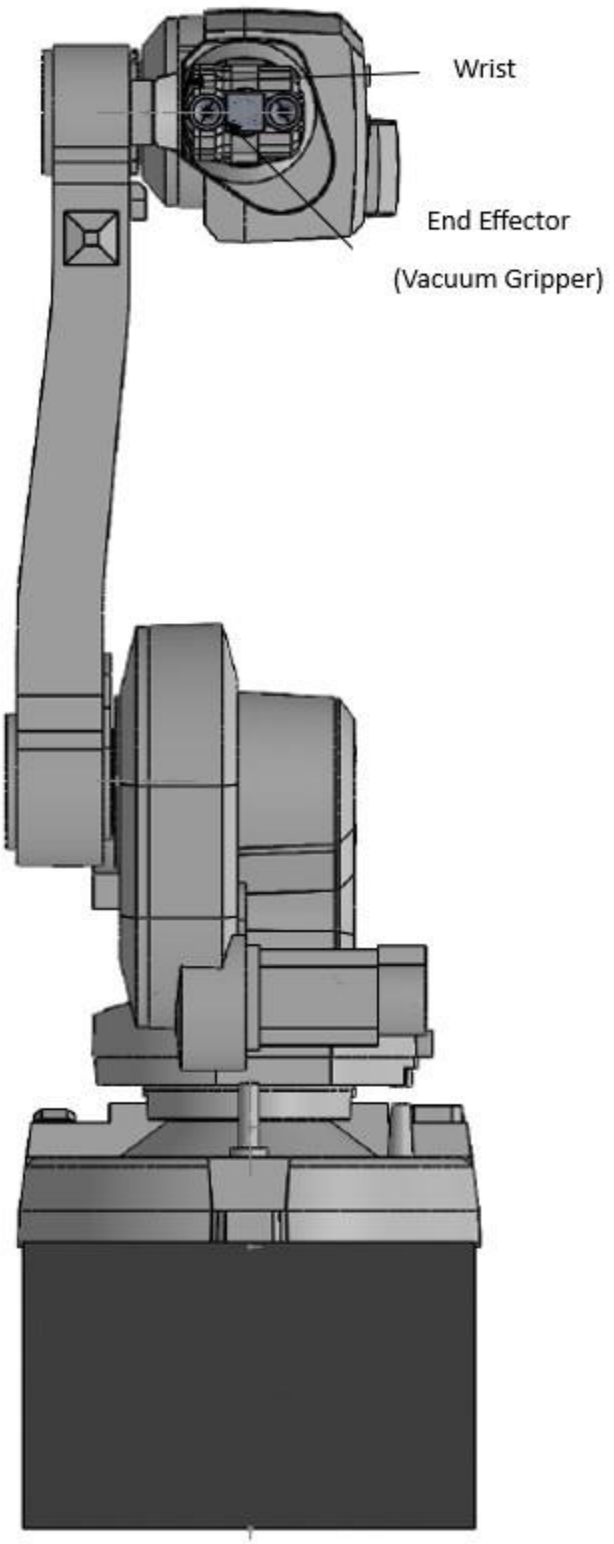
Position	X (mm)	Z (mm)	Axis 2 angle (degrees)	Axis 3 angle (degrees)
0	750	1187	0	0
1	150	1787	0	-90
2	404	643	0	+65
3	1450	487	+90	-90
4	800	-639	+150	-90
5	448	478	+150	-245
6	-6	740	-90	+65
7	-1150	487	-90	-90

CAD MODEL

The CAD model of the ABB IRB 1600 has been acquired from the manufacturer. Using SolidWorks, the assembly of individual components has been performed to construct the robot model.

Supplementary elements such as a Pedestal and an end effector (Vacuum Gripper) have been added to suit the requirements.




Wrist
End Effector
(Vacuum Gripper)

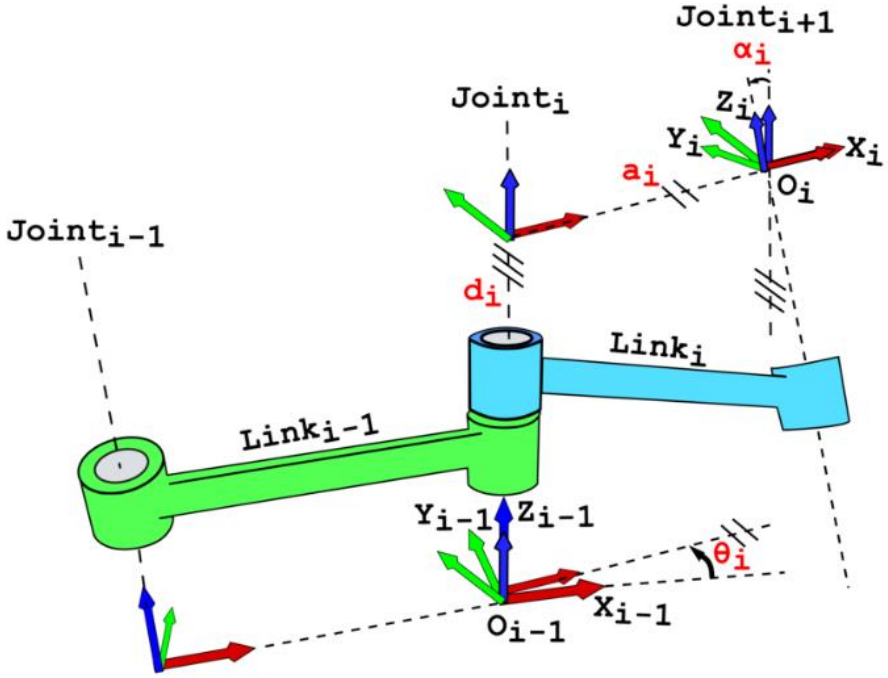
KINEMATICS

FRAME ASSIGNMENT & DH PARAMETERS

A commonly used convention for selecting frames in robotic applications is the DH convention, which simplifies the kinematic analysis of an n-link manipulator. The manipulator is a chain of rigid bodies with a frame or co-ordinate axes attached to each one.

The DH notation contains four parameters that specify the relative location of the current frame with respect to the previous frame. The parameters are as follows:

- a_i : Link Length, that is the length of the common normal, which is the distance from the intersection of the z_{i-1} axis with the x_i axis to the origin of the i^{th} frame along x_i axis.
- α_ι : Link Twist, that is the angle between z_{i-1} and z_i axes about the x_i axis.
- θ_i : Joint Angle, that is the angle between x_{i-1} and x_i axes about the z_{i-1} axis.
- d_i : Link Offset, which is the distance from origin O_{i-1} to the intersection of the z_{i-1} and x_i axes along the z_{i-1} axis.



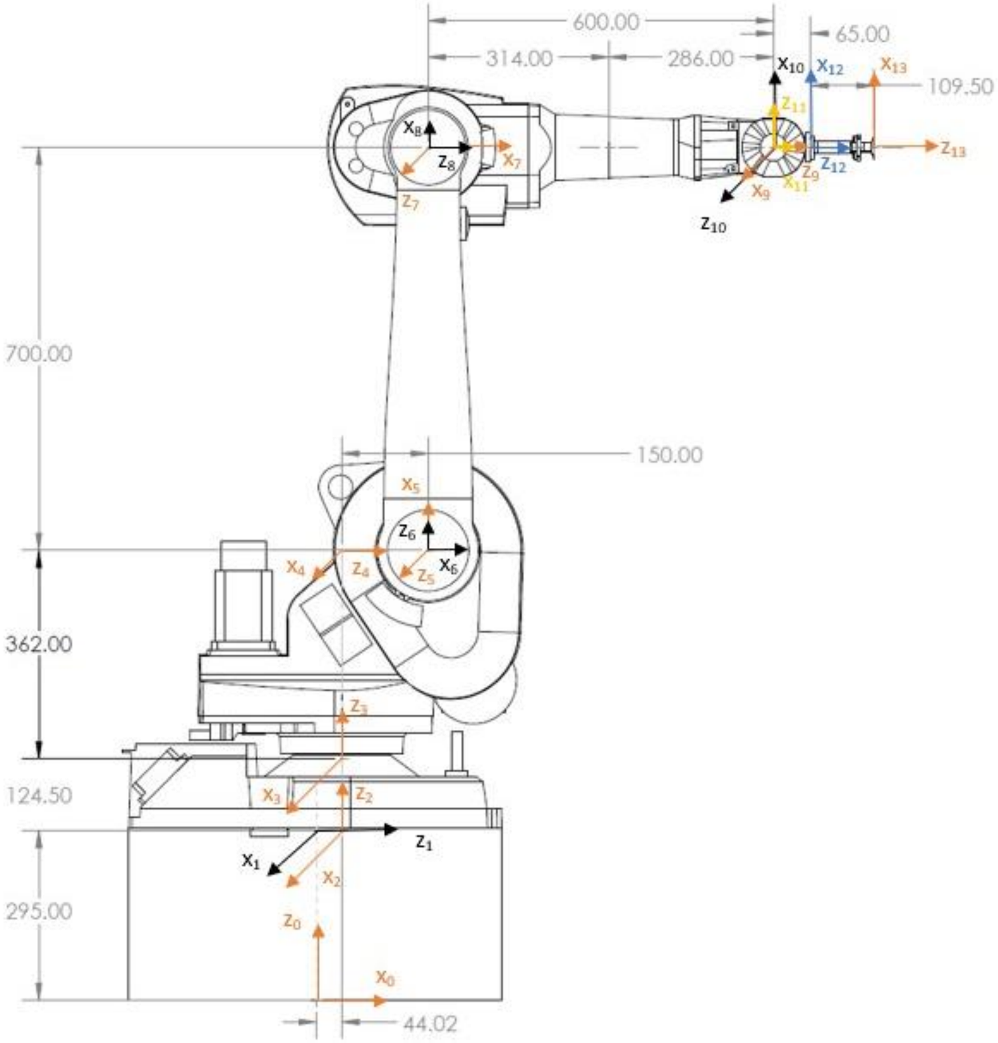
We have followed the following rules that guide us in drawing the frames:

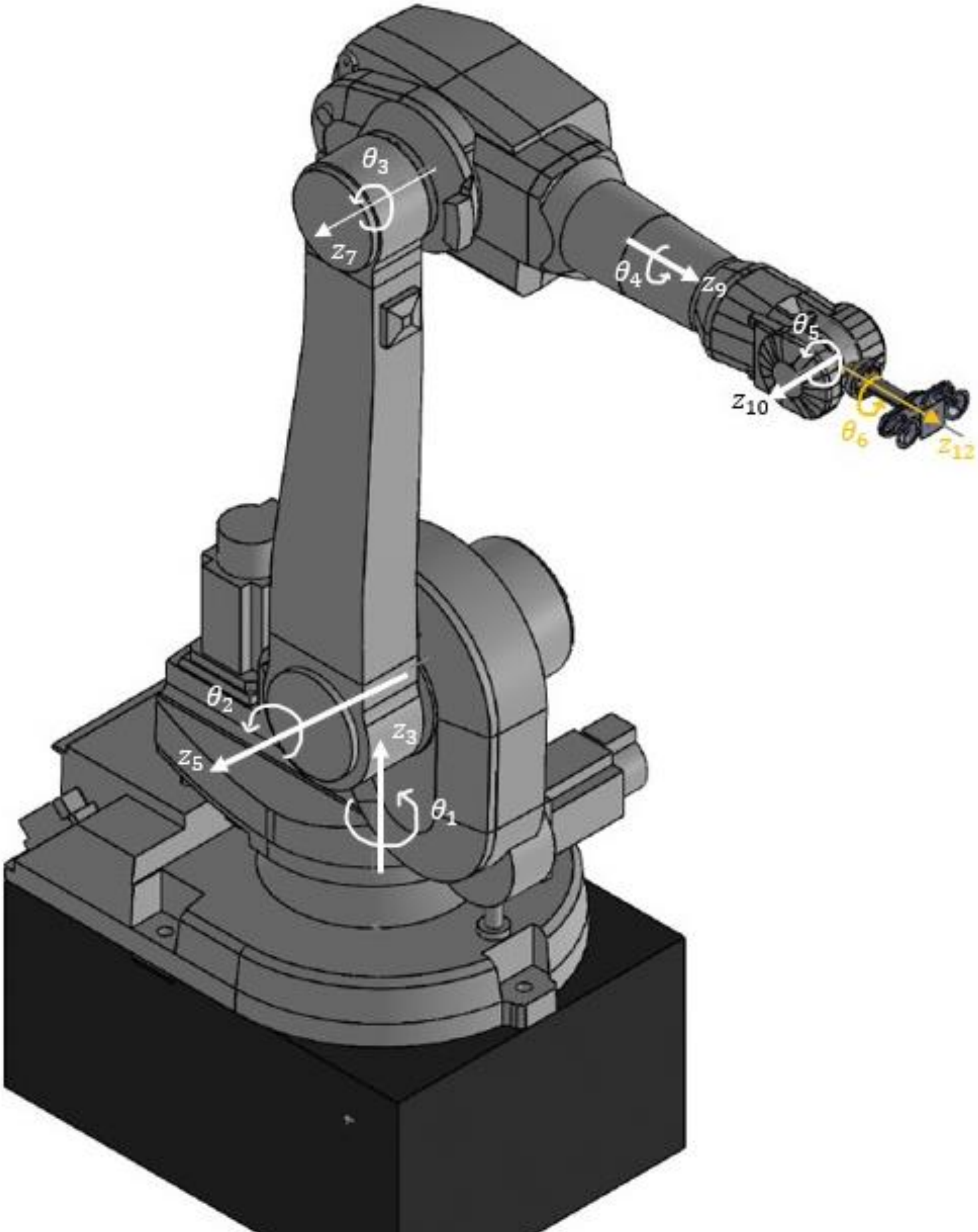
- For a revolute joint z axis is the axis of rotation
- The current x axis must be perpendicular to the z axis of the previous frame
- The current x axis must intersect with the z axis of the previous frame

The frame assignment for the links has been carried out as follows:

The frame 0 is the base frame, frame 2 is the base frame of the robot, the joint frames are 3, 5, 7, 9, 10 and 12, the end effector frame is 13 and the dummy frames used are 1, 4, 6, 8 and 11.

The expected joint rotations and directions for the robot used are as shown in the figure below.




θ_3
z_7
θ_4
z_9
θ_5
z_{10}
θ_6
z_{12}
θ_2
z_5
z_3
θ_1

The DH parameters obtained for our manipulator after assigning Coordinate frames as per DH convention is:

	a_i	α_i	θ_i	d_i
Frame 0 - Frame1	0	-90°	-90°	295
Frame 1 - Frame 2	0	90°	0°	44.02
Frame 2 - Frame 3	0	0°	0°	124.5
Frame 3 - Frame 4	0	-90°	θ_1	362
Frame 4 - Frame 5	0	-90°	-90°	150
Frame 5 - Frame 6	0	-90°	-90° + θ_2	0
Frame 6 - Frame 7	0	90°	0°	700
Frame 7 - Frame 8	0	90°	90° + θ_3	0
Frame 8 - Frame 9	0	0°	90°	314 + 286
Frame 9 - Frame 10	0	-90°	-90° + θ_4	0
Frame 10 - Frame 11	0	90°	θ_5	0
Frame 11 - Frame 12	0	0°	0°	65
Frame 12 - Frame 13	0	0°	θ_6	109.5

Using the following notation each link can be described by a coordinate transformation from the concurrent coordinate system to the previous coordinate system.

$$
{}^{n-1}T_n = \left[\begin{array}{ccc|c} \cos\theta_n & -\sin\theta_n\cos\alpha_n & \sin\theta_n\sin\alpha_n & r_n\cos\theta_n \\ \sin\theta_n & \cos\theta_n\cos\alpha_n & -\cos\theta_n\sin\alpha_n & r_n\sin\theta_n \\ 0 & \sin\alpha_n & \cos\alpha_n & d_n \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{ccc|c} & R & & T \\ \hline 0 & 0 & 0 & 1 \end{array}\right]
$$

Where, R is the 3 x 3 submatrix describing rotation and T is the 3 x 1 submatrix describing translation.

Using the DH table and notation, we obtain the following transformation matrices for the assigned frames and parameters for the ABB IRB 1600 robot :

$$
{}^{0}T_{1}= \begin{bmatrix} cos(-90^\circ) & -(sin(-90^\circ)*cos(-90^\circ)) & (sin(-90^\circ)*\ sin(-90^\circ)) & (0*cos(-90^\circ)) \\ sin(-90^\circ) & (cos(-90^\circ)*cos(-90^\circ)) & -(cos(-90^\circ)*sin(-90^\circ)) & (0*sin(-90^\circ)) \\ 0 & sin(-90^\circ) & cos(-90^\circ) & 295 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{0}T_{1} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 295 \\ 0 & 0 & 0 & 1 \end{bmatrix},
$$

$$
{}^{1}T_{2}= \begin{bmatrix} cos(0^\circ) & -(sin(0^\circ)*cos(90^\circ)) & (sin(0^\circ)*\ sin(90^\circ)) & (0*cos(0^\circ)) \\ sin(0^\circ) & cos(0^\circ)*cos(90^\circ) & -(cos(0^\circ)*sin(90^\circ)) & (0*sin(0^\circ)) \\ 0 & sin(90^\circ) & cos(90^\circ) & 44.02 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{1}T_{2} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 44.02 \\ 0 & 0 & 0 & 1 \end{bmatrix},
$$

$$
{}^{2}T_{3}= \begin{bmatrix} cos\ (0^\circ) & -(sin\ (0^\circ)*cos\ (0^\circ)) & (sin\ (0^\circ)*\ sin\ (0^\circ)) & (0*cos\ (0^\circ)) \\ sin\ (0^\circ) & (cos\ (0^\circ)*cos\ (0^\circ)) & -(\mathrm{cos}\ (0^\circ)*sin\ (0^\circ)) & (0*sin\ (0^\circ)) \\ 0 & sin\ (0^\circ) & \mathrm{cos}\ (0^\circ) & 124.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{2}T_{3} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 124.5 \\ 0 & 0 & 0 & 1 \end{bmatrix},
$$

$$
{}^{3}T_{4}= \begin{bmatrix} cos\ (\theta_1) & -(sin\ \ (\theta_1)*cos\ (-90^\circ)) & (sin\ \ (\theta_1)*\ sin\ \ (-90^\circ)) & (0*cos\ \ (\theta_1)) \\ sin\ (\theta_1) & (cos\ \ (\theta_1)*cos\ (-90^\circ)) & -(cos\ \ (\theta_1)*sin\ \ (-90^\circ)) & (0*sin\ \ (\theta_1)) \\ 0 & sin\ \ (-90^\circ) & cos\ \ (-90^\circ) & 362 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{3}T_{4} = \begin{bmatrix} cos(\theta_1) & 0 & -sin(\theta_1) & 0 \\ sin(\theta_1) & 0 & cos(\theta_1) & 0 \\ 0 & -1 & 0 & 362 \\ 0 & 0 & 0 & 1 \end{bmatrix},
$$

$^4T_5$=

$$\begin{bmatrix} cos\,(-90°) & -(sin\;(-90°) * cos\,(-90°)) & (sin\;(-90°) *\; sin\;(-90°)) & (0 * cos\;(-90°)) \\ sin\,(-90°) & (cos\;(-90°) * cos\,(-90°)) & -(\cos\;(-90°) * sin\;(-90°)) & (0 * sin\;(-90°)) \\ 0 & sin\;(-90°) & \cos\;(-90°) & 150 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^4T_5 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 150 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$^5T_6$=

$$\begin{bmatrix} cos\,(-90° + \theta_2) & -(sin\;(-90° + \theta_2) * cos\,(-90°)) & (sin\;(-90° + \theta_2) *\; sin\;(-90°)) & (0 * cos\,(-90° + \theta_2)) \\ sin\,(-90° + \theta_2) & (cos\;(-90° + \theta_2) * cos\,(-90°)) & -(\cos\;(-90° + \theta_2) * sin\;(-90°)) & (0 * sin\;(-90° + \theta_2)) \\ 0 & sin\;(-90°) & \cos\;(-90°) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^5T_6 = \begin{bmatrix} sin(\theta_2) & 0 & cos(\theta_2) & 0 \\ -cos(\theta_2) & 0 & sin(\theta_2) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^6T_7 = \begin{bmatrix} cos\,(0°) & -(sin\;(0°) * cos\,(90°)) & (sin\;(0°) *\; sin\;(90°)) & (0 * cos\;(0°)) \\ sin\,(0°) & (cos\;(0°) * cos\,(90°)) & -(\cos\;(0°) * sin\;(90°)) & (0 * sin\;(0°)) \\ 0 & sin\;(90°) & \cos\;(90°) & 700 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^6T_7 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 700 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$^7T_8$=

$$\begin{bmatrix} cos\,(90° + \theta_3) & -(sin\;(90° + \theta_3) * cos\,(90°)) & (sin\;(90° + \theta_3) *\; sin\;(90°)) & (0 * cos\,(90° + \theta_3)) \\ sin\,(90° + \theta_3) & (cos\;(90° + \theta_3) * cos\,(90°)) & -(\cos\;(90° + \theta_3) * sin\;(90°)) & (0 * sin\;(90° + \theta_3)) \\ 0 & sin\;(90°) & \cos\;(90°) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^7T_8 = \begin{bmatrix} -sin(\theta_3) & 0 & cos(\theta_3) & 0 \\ cos(\theta_3) & 0 & sin(\theta_3) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$^{8}T_{9}=\begin{bmatrix} cos\,(90°) & -(sin\ (90°) * cos\,(0°)) & (sin\ (90°) *\ sin\ (0°)) & (0 * cos\ (90°)) \\ sin\,(90°) & (cos\ (90°) * cos\,(0°)) & -(\cos\ (90°) * sin\ (0°)) & (0 * sin\ (90°)) \\ 0 & sin\ (0°) & \cos\ (0°) & 314 + 286 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^{8}T_{9}=\begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 600 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$^{9}T_{10}=$

$$\begin{bmatrix} cos\,(-90° + \theta_4) & -(sin\ (-90° + \theta_4) * cos\,(-90°)) & (sin\ (-90° + \theta_4) *\ sin\ (-90°)) & (0 * cos\ (-90° + \theta_4)) \\ sin\,(-90° + \theta_4) & (cos\ (-90° + \theta_4) * cos\,(-90°)) & -(\cos\ (-90° + \theta_4) * sin\ (-90°)) & (0 * sin\ (-90° + \theta_4)) \\ 0 & sin\ (-90°) & \cos\ (-90°) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^{9}T_{10}=\begin{bmatrix} sin(\theta_4) & 0 & cos(\theta_4) & 0 \\ -cos(\theta_4) & 0 & sin(\theta_4) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$^{10}T_{11}=\begin{bmatrix} cos\,(\theta_5) & -(sin(\theta_5) * cos\,(90°)) & (sin\,(\theta_5) *\ sin\,(90°)) & (0 * cos\,(\theta_5)) \\ sin\,(\theta_5) & (cos\,(\theta_5) * cos\,(90°)) & -(\cos\,(\theta_5) * sin\,(90°)) & (0 * sin\,(\theta_5)) \\ 0 & sin\,(90°) & \cos\,(90°) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^{10}T_{11}=\begin{bmatrix} cos(\theta_5) & 0 & sin(\theta_5) & 0 \\ sin(\theta_5) & 0 & -cos(\theta_5) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$^{11}T_{12}=\begin{bmatrix} cos\,(0°) & -(sin\,(0°) * cos\,(0°)) & (sin\,(0°) *\ sin\,(0°)) & (0 * cos\,(0°)) \\ sin\,(0°) & (cos(0°) * cos\,(0°)) & -(\cos\,(0°) * sin\,(0°)) & (0 * sin\,(0°)) \\ 0 & sin\,(0°) & \cos\,(0°) & 65 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^{11}T_{12}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 65 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$^{12}T_{13}=\begin{bmatrix} cos\,(\theta_6) & -(sin(\theta_6)*cos\,(0°)) & (sin\,(\theta_6)*\ sin\,(0°)) & (0*cos(\theta_6)) \\ sin\,(\theta_6) & (cos\,(\theta_6)*cos\,(0°)) & -(\cos(\theta_6)*sin\,(0°)) & (0*sin\,(\theta_6)) \\ 0 & sin\,(0°) & \cos\,(0°) & 109.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^{12}T_{13}=\begin{bmatrix} cos(\theta_6) & -sin(\theta_6) & 0 & 0 \\ sin(\theta_6) & cos(\theta_6) & 0 & 0 \\ 0 & 0 & 1 & 109.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Homogeneous transformation matrix for transformation of the relative transformation between the end effector frame and the base frame is calculated as follows,

$^0T_{13} = {}^0T_1 * {}^1T_2 *{}^2T_3 *{}^3T_4 *{}^4T_5 *{}^5T_6 *{}^6T_7 *{}^7T_8 *{}^8T_9 *{}^9T_{10} *{}^{10}T_{11} *{}^{11}T_{12} *{}^{12}T_{13}$

The matrix is written in python using Sympy library in symbolic form and the final transformation matrix is calculated using the above sequence.

The final transformation matrix is obtained through python code in symbolic form, and it is as shown below:

$$^0T_{13}=\begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 295 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 44.02 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 124.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} *$$

$$\begin{bmatrix} cos(\theta_1) & 0 & -sin(\theta_1) & 0 \\ sin(\theta_1) & 0 & cos(\theta_1) & 0 \\ 0 & -1 & 0 & 362 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 150 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} sin(\theta_2) & 0 & cos(\theta_2) & 0 \\ -cos(\theta_2) & 0 & sin(\theta_2) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} *$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 700 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} -sin(\theta_3) & 0 & cos(\theta_3) & 0 \\ cos(\theta_3) & 0 & sin(\theta_3) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 600 \\ 0 & 0 & 0 & 1 \end{bmatrix} *$$

$$\begin{bmatrix} sin(\theta_4) & 0 & cos(\theta_4) & 0 \\ -cos(\theta_4) & 0 & sin(\theta_4) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} cos(\theta_5) & 0 & sin(\theta_5) & 0 \\ sin(\theta_5) & 0 & -cos(\theta_5) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 65 \\ 0 & 0 & 0 & 1 \end{bmatrix} *$$

$$\begin{bmatrix} cos(\theta_6) & -sin(\theta_6) & 0 & 0 \\ sin(\theta_6) & cos(\theta_6) & 0 & 0 \\ 0 & 0 & 1 & 109.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

${}^{0}T_{13}$ =

$$(-(\sin(\theta_2)\cos(\theta_1)\cos(\theta_3)+\sin(\theta_3)\cos(\theta_1)\cos(\theta_2))\cos(\theta_4)+\sin(\theta_1)\sin(\theta_4))\cos(\theta_5)+(\sin(\theta_2)\sin(\theta_3)\cos(\theta_1)-\cos(\theta_1)\cos(\theta_2)\cos(\theta_3))\sin$$
$$(\theta_6)+((\sin(\theta_2)\cos(\theta_1)\cos(\theta_3)+\sin(\theta_3)\cos(\theta_1)\cos(\theta_2))\sin(\theta_4)+\sin(\theta_1)\cos(\theta_4))\sin(\theta_6)$$

$$((-(\sin(\theta_1)\sin(\theta_2)\cos(\theta_3)+\sin(\theta_1)\sin(\theta_3)\cos(\theta_2))\cos(\theta_4)-\sin(\theta_4)\cos(\theta_1))\cos(\theta_5)+(\sin(\theta_1)\sin(\theta_2)\sin(\theta_3)-\sin(\theta_1)\cos(\theta_2)\cos(\theta_3))\sin$$
$$(\theta_6)+((\sin(\theta_1)\sin(\theta_2)\cos(\theta_3)+\sin(\theta_1)\sin(\theta_3)\cos(\theta_2))\sin(\theta_4)-\cos(\theta_1)\cos(\theta_4))\sin(\theta_6)$$

$$(\sin(\theta_2)\sin(\theta_3)-\cos(\theta_2)\cos(\theta_3))\sin(\theta_4)\sin(\theta_6)$$
$$+(-(\sin(\theta_2)\sin(\theta_3)-\cos(\theta_2)\cos(\theta_3))\cos(\theta_4)\cos(\theta_5)+(-\sin(\theta_2)\cos(\theta_3)-\sin(\theta_3)\cos(\theta_2))\sin(\theta_5))\cos(\theta_6)$$
$$0$$

Column 1

$$((-(\sin(\theta_2)\cos(\theta_1)\cos(\theta_3)+\sin(\theta_3)\cos(\theta_1)\cos(\theta_2))\cos(\theta_4)+\sin(\theta_1)\sin(\theta_4))\cos(\theta_5)+(\sin(\theta_2)\sin(\theta_3)\cos(\theta_1)-\cos(\theta_1)\cos(\theta_2)\cos(\theta_3))\mathrm{si}$$
$$(\theta_6)+((\sin(\theta_2)\cos(\theta_1)\cos(\theta_3)+\sin(\theta_3)\cos(\theta_1)\cos(\theta_2))\sin(\theta_4)+\sin(\theta_1)\cos(\theta_4))\cos(\theta_6)$$

$$\cdot((-(\sin(\theta_1)\sin(\theta_2)\cos(\theta_3)+\sin(\theta_1)\sin(\theta_3)\cos(\theta_2))\cos(\theta_4)-\sin(\theta_4)\cos(\theta_1))\cos(\theta_5)+(\sin(\theta_1)\sin(\theta_2)\sin(\theta_3)-\sin(\theta_1)\cos(\theta_2)\cos(\theta_3))\sin$$
$$(\theta_6)+((\sin(\theta_1)\sin(\theta_2)\cos(\theta_3)+\sin(\theta_1)\sin(\theta_3)\cos(\theta_2))\sin(\theta_4)-\cos(\theta_1)\cos(\theta_4))\cos(\theta_6)$$

$$(\sin(\theta_2)\sin(\theta_3)-\cos(\theta_2)\cos(\theta_3))\sin(\theta_4)\cos(\theta_6)$$
$$-(-(\sin(\theta_2)\sin(\theta_3)-\cos(\theta_2)\cos(\theta_3))\cos(\theta_4)\cos(\theta_5)+(-\sin(\theta_2)\cos(\theta_3)-\sin(\theta_3)\cos(\theta_2))\sin(\theta_5))\sin(\theta_6)$$
$$0$$

Column 2

$$(-(\sin(\theta_2)\cos(\theta_1)\cos(\theta_3)+\sin(\theta_3)\cos(\theta_1)\cos(\theta_2))\cos(\theta_4)+\sin(\theta_1)\sin(\theta_4))\sin(\theta_5)$$
$$-(\sin(\theta_2)\sin(\theta_3)\cos(\theta_1)-\cos(\theta_1)\cos(\theta_2)\cos(\theta_3))\cos(\theta_5)$$

$$(-(\sin(\theta_1)\sin(\theta_2)\cos(\theta_3)+\sin(\theta_1)\sin(\theta_3)\cos(\theta_2))\cos(\theta_4)-\sin(\theta_4)\cos(\theta_1))\sin(\theta_5)$$
$$-(\sin(\theta_1)\sin(\theta_2)\sin(\theta_3)-\sin(\theta_1)\cos(\theta_2)\cos(\theta_3))\cos(\theta_5)$$

$$-(\sin(\theta_2)\sin(\theta_3)-\cos(\theta_2)\cos(\theta_3))\sin(\theta_5)\cos(\theta_4)-(-\sin(\theta_2)\cos(\theta_3)-\sin(\theta_3)\cos(\theta_2))\cos(\theta_5)$$

$$0$$

Column 3

$$174.5\,(-(\sin(\theta_2)\cos(\theta_1)\cos(\theta_3)+\sin(\theta_3)\cos(\theta_1)\cos(\theta_2))\cos(\theta_4)+\sin(\theta_1)\sin(\theta_4))\sin(\theta_5)$$
$$-174.5\,(\sin(\theta_2)\sin(\theta_3)\cos(\theta_1)-\cos(\theta_1)\cos(\theta_2)\cos(\theta_3))\cos(\theta_5)-600\sin(\theta_2)\sin(\theta_3)\cos(\theta_1)-700\sin(\theta_2)\cos(\theta_1)+600\cos(\theta_1)\cos$$
$$(\theta_2)\cos(\theta_3)+150\cos(\theta_1)+44.02$$
$$174.5\,(-(\sin(\theta_1)\sin(\theta_2)\cos(\theta_3)+\sin(\theta_1)\sin(\theta_3)\cos(\theta_2))\cos(\theta_4)-\sin(\theta_4)\cos(\theta_1))\sin(\theta_5)$$
$$-174.5\,(\sin(\theta_1)\sin(\theta_2)\sin(\theta_3)-\sin(\theta_1)\cos(\theta_2)\cos(\theta_3))\cos(\theta_5)-600\sin(\theta_1)\sin(\theta_2)\sin(\theta_3)-700\sin(\theta_1)\sin(\theta_2)+600\sin(\theta_1)\cos$$
$$(\theta_2)\cos(\theta_3)+150\sin(\theta_1)$$
$$-174.5\,(\sin(\theta_2)\sin(\theta_3)-\cos(\theta_2)\cos(\theta_3))\sin(\theta_5)\cos(\theta_4)-174.5\,(-\sin(\theta_2)\cos(\theta_3)-\sin(\theta_3)\cos(\theta_2))\cos(\theta_5)+600\sin(\theta_2)\cos(\theta_3)$$
$$+600\sin(\theta_3)\cos(\theta_2)+700\cos(\theta_2)+781.5$$
$$1$$

Column 4

FORWARD KINEMATICS VALIDATION

In the context of robotics, the term "forward kinematics" refers to the process of utilizing kinematics equations to determine the position of the end-effector based on the values that are specified for the joint parameters.

When all joint parameters are known and a kinematic chain consisting of links and joints with multiple degrees of freedom is given, we determine the position and orientation of the end-effector in the operational workspace. The term for this is Forward Kinematics.

Forward kinematics validation is performed for the obtained transformation matrix by substituting the joint angle values to the obtained transformation matrix, which is in symbolic form in Python code. The obtained transformation matrix will represent the end effector frame's translation and rotation relative to the base frame. This is compared with the geometric configuration for the set of assumed joint angle values.

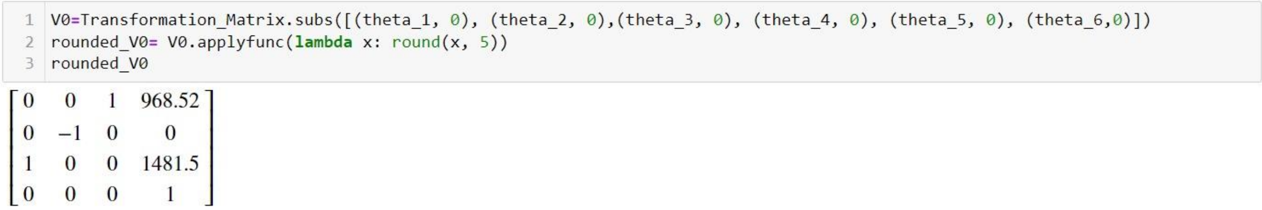
The validation process aims to assess the accuracy of the Denavit-Hartenberg (DH) parameters and the resultant transformation matrix. This validation involves a comparative analysis of outcomes sourced from three distinct channels:

1. The Solidworks application, where manual orientation of the robot is performed, and the distance and orientation of the end effector are measured.
2. Utilizing a Python script to input joint angle values, the orientation and position terms are then extracted from the resulting transformation matrix.
3. Leveraging Gazebo, the validation extends to the use of an odometry plugin. Joint angles are published to the position controller, and data on position and orientation of the end effector concerning the base frame is obtained by subscribing to the /odom topic.

Validation 0 : Home Position

$[\,\theta_1 = 0°, \theta_2 = 0°, \theta_3 = 0°, \theta_4 = 0°, \theta_5 = 0°, \theta_6 = 0°\,]$

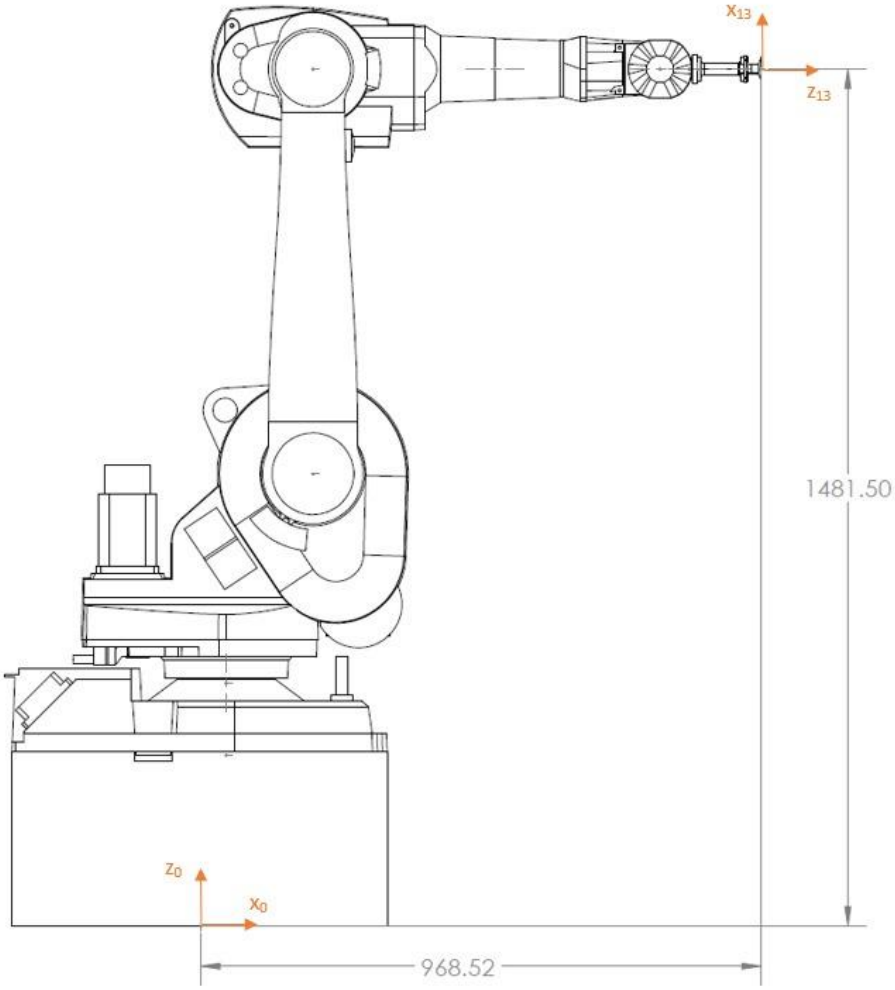
The values of theta’s are substituted in the transformation matrix, and the following homogeneous transformation matrix is obtained,



```
V0=Transformation_Matrix.subs([(theta_1, 0), (theta_2, 0),(theta_3, 0), (theta_4, 0), (theta_5, 0), (theta_6,0)])
rounded_V0= V0.applyfunc(lambda x: round(x, 5))
rounded_V0
```

$$\begin{bmatrix} 0 & 0 & 1 & 968.52 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 1481.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

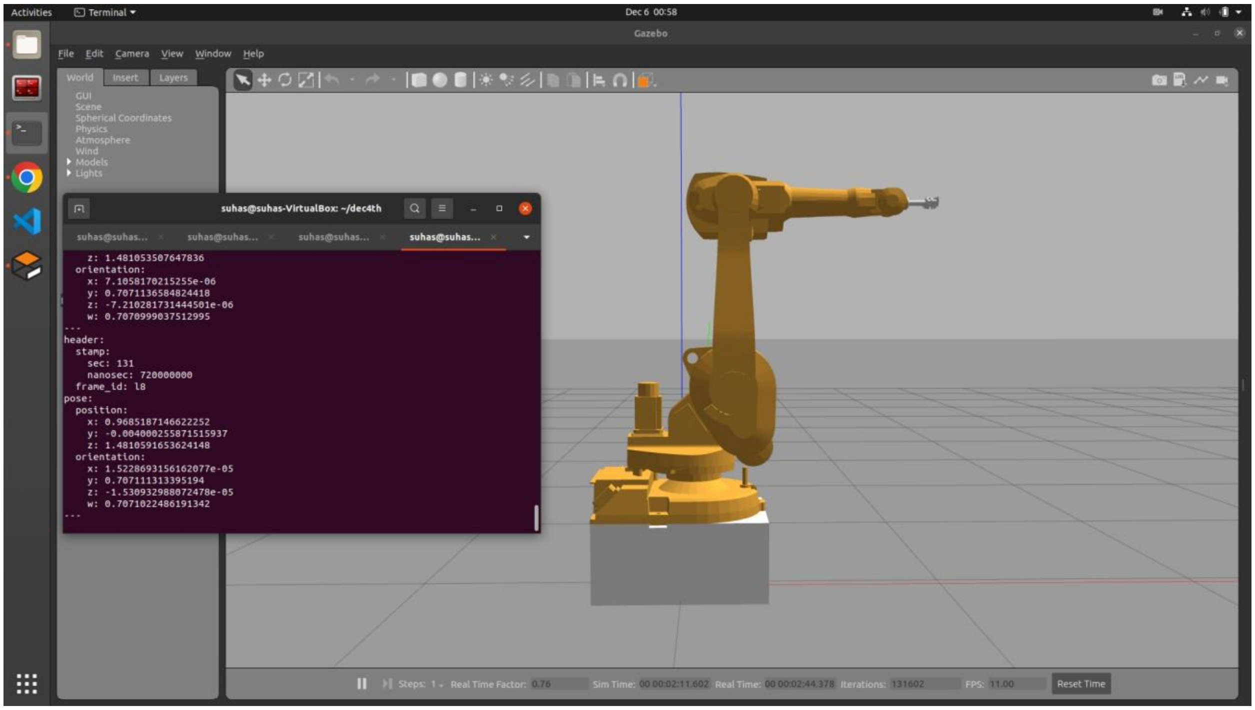
<u>Geometric Validation (SolidWorks):</u>



From the figure:

- $X_{13}.X_0 = 0$, $Y_{13}.\ X_0 = 0$, $Z_{13}.\ X_0 = 1$
- $X_{13}.Y_0 = 0$, $Y_{13}.\ Y_0 = -1$, $Z_{13}.\ Y_0 = 0$
- $X_{13}.Z_0 = 1$, $Y_{13}.\ Z_0 = 0$, $Z_{13}.\ Z_0 = 0$
- X Translation: 968.52 mm
- Y Translation : 0 mm
- Z Translation: 1481.5 mm

Odom Validation (Gazebo):


Activities
Terminal
Dec 6 00:58
Gazebo
File Edit Camera View Window Help
World Insert Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
Lights
suhas@suhas-VirtualBox: ~/dec4th
suhas@suhas...
suhas@suhas...
suhas@suhas...
suhas@suhas...
Steps: 1 Real Time Factor: 0.76 Sim Time: 00 00:02:11.602 Real Time: 00 00:02:44.378 Iterations: 131602 FPS: 11.00 Reset Time

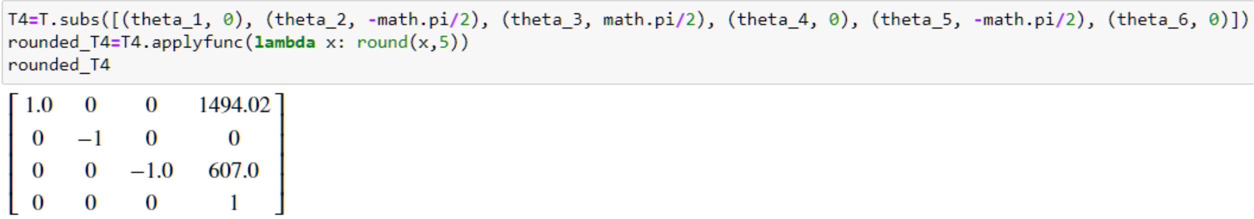
```
    z: 1.481053507647836
  orientation:
    x: 7.1058170215255e-06
    y: 0.7071136584824418
    z: -7.210281731444501e-06
    w: 0.7070999037512995
---
header:
  stamp:
    sec: 131
    nanosec: 720000000
  frame_id: l8
pose:
  position:
    x: 0.9685187146622252
    y: -0.004000255871515937
    z: 1.4810591653624148
  orientation:
    x: 1.5228693156162077e-05
    y: 0.707111313395194
    z: -1.530932988072478e-05
    w: 0.7071022486191342
---
```

The values obtained from Geometric Validation and from Odom sensor match with the values obtained from the homogeneous transformation matrix. Thus, the transformation matrix computed in symbolic form is correct.

Validation 1 : Orientation 1

[$\theta_1 = 0°, \theta_2 = -90°, \theta_3 = 90°, \theta_4 = 0°, \theta_5 = -90°, \theta_6 = 0°$]

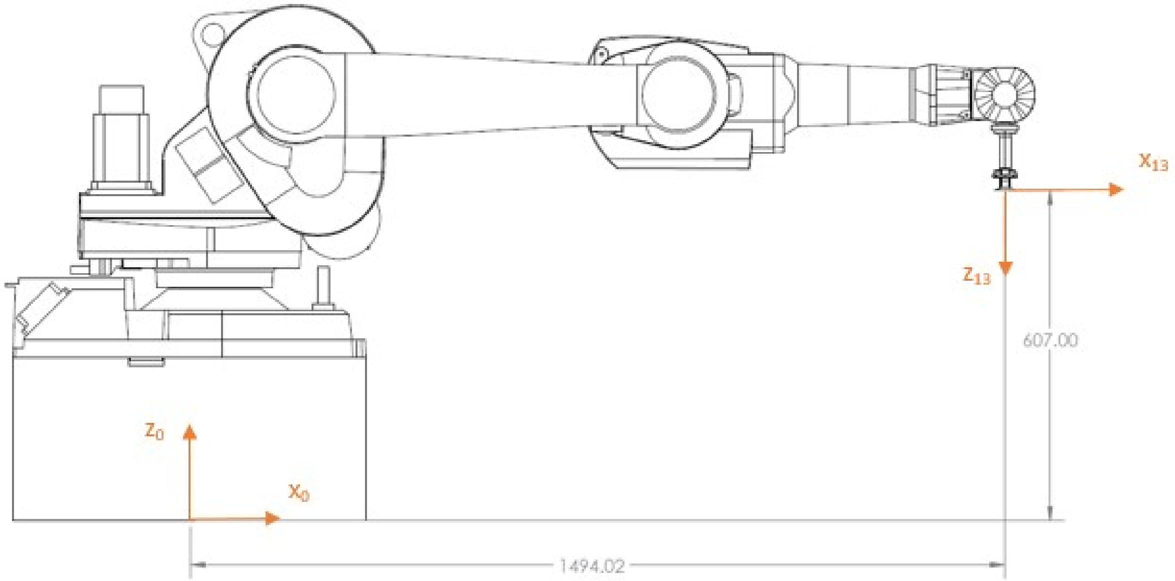
The values of theta's are substituted in the transformation matrix, and the following homogeneous transformation matrix is obtained,



```
T4=T.subs([(theta_1, 0), (theta_2, -math.pi/2), (theta_3, math.pi/2), (theta_4, 0), (theta_5, -math.pi/2), (theta_6, 0)])
rounded_T4=T4.applyfunc(lambda x: round(x,5))
rounded_T4
```

$$\begin{bmatrix} 1.0 & 0 & 0 & 1494.02 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1.0 & 607.0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

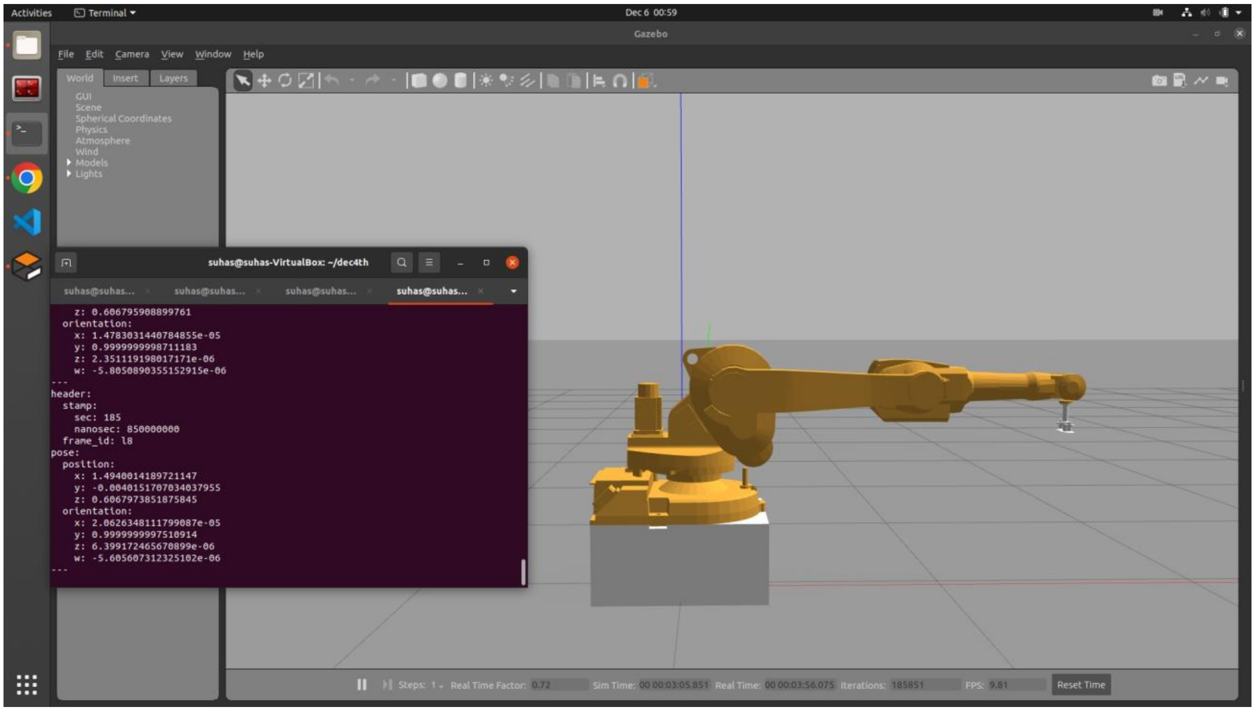
<u>Geometric Validation (SolidWorks):</u>



From the figure:

- $X_{13}.X_0 = 1$, $Y_{13}. X_0 = 0$, $Z_{13}. X_0 = 0$
- $X_{13}.Y_0 = 0$, $Y_{13}. Y_0 = -1$, $Z_{13}. Y_0 = 0$
- $X_{13}.Z_0 = 0$, $Y_{13}. Z_0 = 0$, $Z_{13}. Z_0 = -1$
- X Translation: 1494.02 mm
- Y Translation : 0 mm
- Z Translation: 607 mm

Odom Validation (Gazebo):



```
    z: 0.606795908899761
  orientation:
    x: 1.4783031440784855e-05
    y: 0.9999999998711183
    z: 2.351119198017171e-06
    w: -5.8050890355152915e-06
---
header:
  stamp:
    sec: 185
    nanosec: 850000000
  frame_id: l8
pose:
  position:
    x: 1.4940014189721147
    y: -0.0040151707034037955
    z: 0.6067973851875845
  orientation:
    x: 2.0626348111799087e-05
    y: 0.9999999997510914
    z: 6.399172465670899e-06
    w: -5.605607312325102e-06
---
```

The values obtained from Geometric Validation and from Odom sensor match with the values obtained from the homogeneous transformation matrix. Thus, the transformation matrix computed in symbolic form is correct.

Validation 2 : Orientation 2

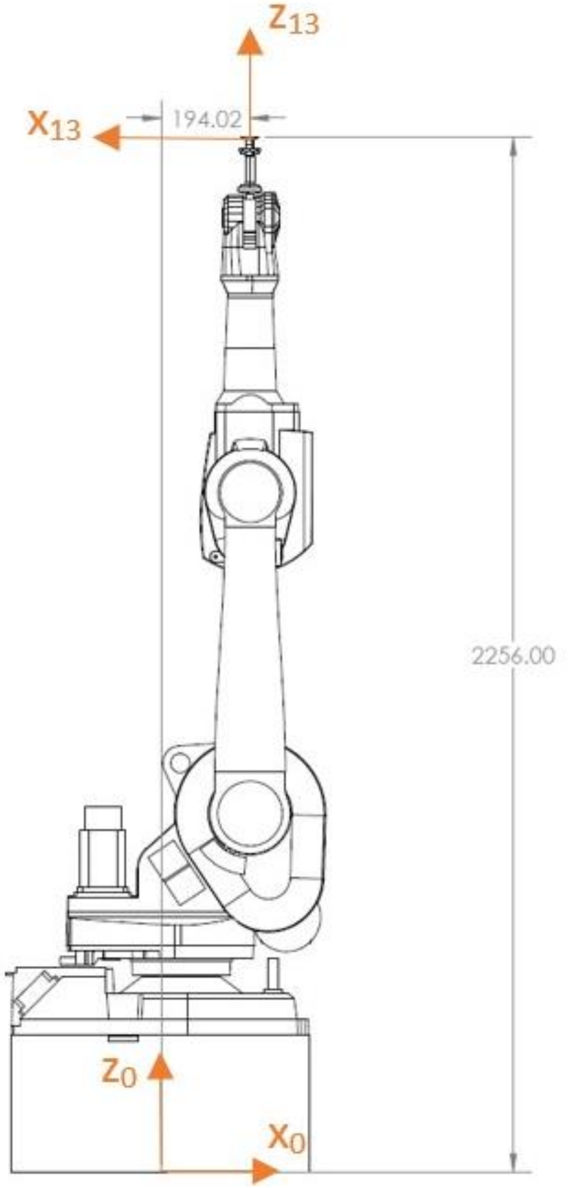
[$\theta_1 = 0°$, $\theta_2 = 0°$, $\theta_3 = 90°$, $\theta_4 = 0°$, $\theta_5 = 0°$, $\theta_6 = 0°$]

The values of theta's are substituted in the transformation matrix, and the following homogeneous transformation matrix is obtained,

```
T2=T.subs([(theta_1, 0), (theta_2, 0), (theta_3, math.pi/2), (theta_4, 0), (theta_5, 0), (theta_6, 0)])
rounded_T2=T2.applyfunc(lambda x: round(x,5))
rounded_T2
```

$$\begin{bmatrix} -1.0 & 0 & 0 & 194.02 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1.0 & 2256.0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

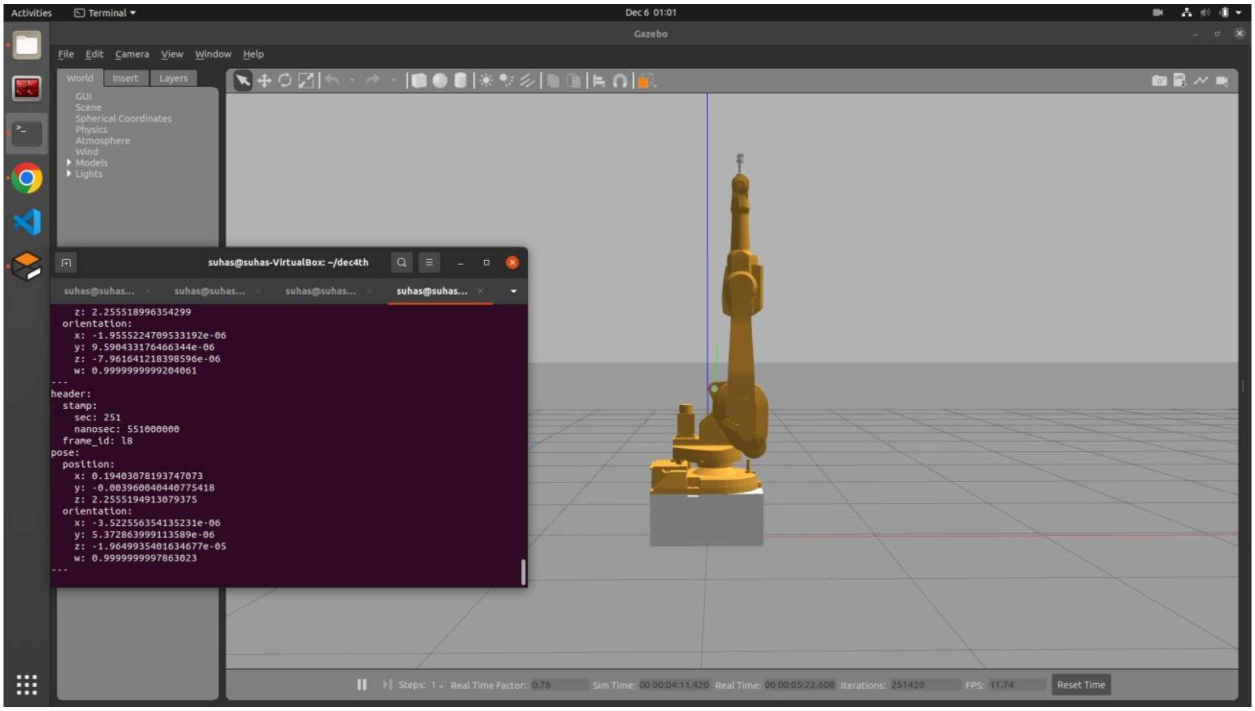
Geometric Validation (SolidWorks):



From the figure:

- $X_{13}.X_0 = -1$, $Y_{13}. X_0= 0$, $Z_{13}. X_0= 0$
- $X_{13}.Y_0 = 0$, $Y_{13}. Y_0= -1$, $Z_{13}. Y_0= 0$
- $X_{13}.Z_0 = 0$, $Y_{13}. Z_0= 0$, $Z_{13}. Z_0= 1$
- X Translation: 192.02 mm
- Y Translation : 0 mm
- Z Translation: 2256.00 mm

Odom Validation (Gazebo):



```
    z: 2.255518996354299
  orientation:
    x: -1.9555224709533192e-06
    y: 9.590433176466344e-06
    z: -7.961641218398596e-06
    w: 0.9999999999204061
---
header:
  stamp:
    sec: 251
    nanosec: 551000000
  frame_id: l8
pose:
  position:
    x: 0.19403078193747073
    y: -0.003960040440775418
    z: 2.2555194913079375
  orientation:
    x: -3.522556354135231e-06
    y: 5.372863999113589e-06
    z: -1.9649935401634677e-05
    w: 0.9999999997863023
---
```

Steps: 1 Real Time Factor: 0.76 Sim Time: 00 00:04:11.420 Real Time: 00 00:05:22.608 Iterations: 251420 FPS: 11.74

The values obtained from Geometric Validation and from Odom sensor match with the values obtained from the homogeneous transformation matrix. Thus, the transformation matrix computed in symbolic form is correct.

Validation 3 : Orientation 3

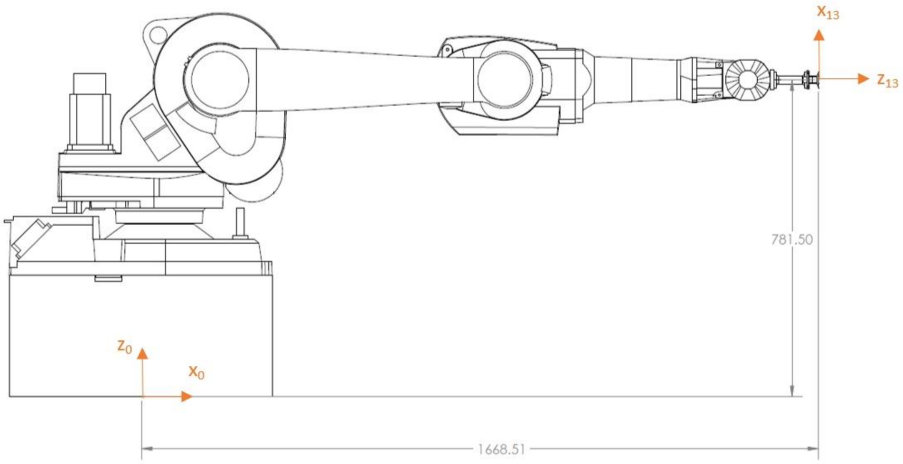
[$\theta_1 = 0°$, $\theta_2 = -90°$, $\theta_3 = 90°$, $\theta_4 = 0°$, $\theta_5 = 0°$, $\theta_6 = 0°$]

The values of theta's are substituted in the transformation matrix, and the following homogeneous transformation matrix is obtained,

```
T3=T.subs([(theta_1, 0), (theta_2, -math.pi/2), (theta_3, math.pi/2), (theta_4, 0), (theta_5, 0), (theta_6, 0)])
rounded_T3=T3.applyfunc(lambda x: round(x,5))
rounded_T3
```

$$\begin{bmatrix} 0 & 0 & 1.0 & 1668.52 \\ 0 & -1 & 0 & 0 \\ 1.0 & 0 & 0 & 781.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

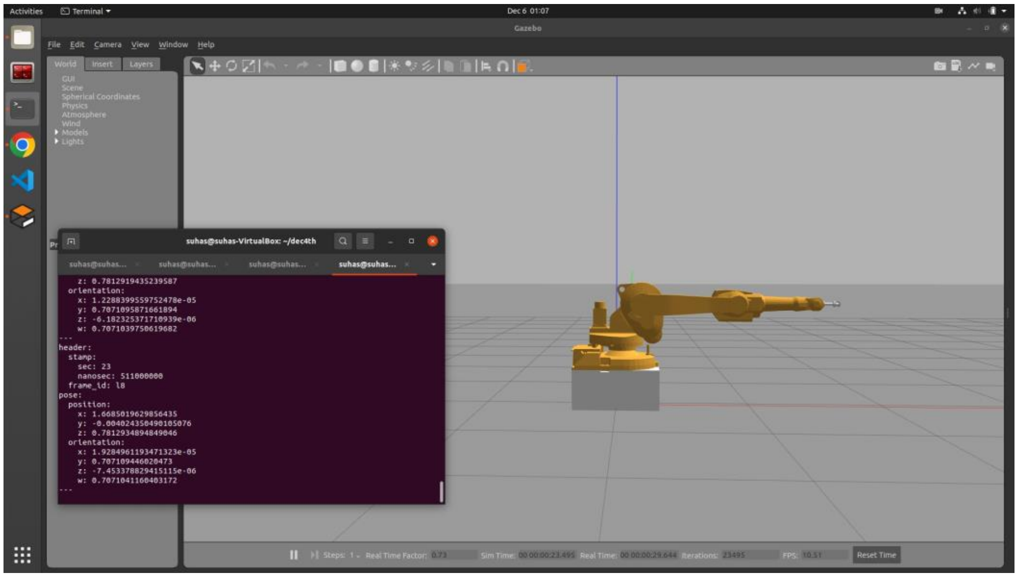
Geometric Validation (SolidWorks):



From the figure:

- $X_{13}.X_0 = 0$, $Y_{13}. X_0= 0$, $Z_{13}. X_0= 1$
- $X_{13}.Y_0 = 0$, $Y_{13}. Y_0= -1$, $Z_{13}. Y_0= 0$
- $X_{13}. Z_0 = 1$, $Y_{13}. Z_0= 0$, $Z_{13}. Z_0= 0$
- X Translation : 1668.51 mm
- Y Translation : 0 mm
- Z Translation : 781.50 mm

Odom Validation (Gazebo):



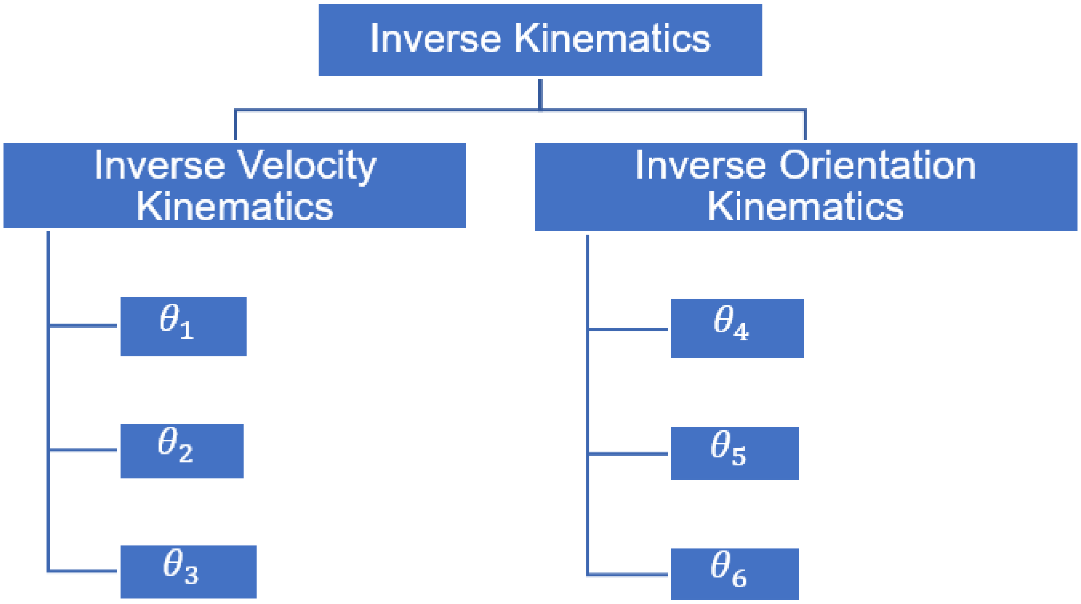
```
    z: 0.7812919435239587
  orientation:
    x: 1.2288399559752478e-05
    y: 0.7071095871661894
    z: -6.182325371710939e-06
    w: 0.7071039750619682
---
header:
  stamp:
    sec: 23
    nanosec: 511000000
  frame_id: l8
pose:
  position:
    x: 1.6685019629856435
    y: -0.004024350490105076
    z: 0.7812934894849046
  orientation:
    x: 1.9284961193471323e-05
    y: 0.707109446020473
    z: -7.453378829415115e-06
    w: 0.7071041160403172
---
```

The values obtained from Geometric Validation and from Odom sensor match with the values obtained from the homogeneous transformation matrix. Thus, the transformation matrix computed in symbolic form is correct.

INVERSE KINEMATICS

Inverse Kinematics is the process of ascertaining the joint parameters by considering the desired position and orientation of the end-effector. It serves as a valuable tool for establishing the intended movements and routes for robotic arms or manipulators to attain predefined target positions. This methodology empowers robots to follow predetermined trajectories by computing the essential joint angles at every point along the trajectory.

In this project, we have followed a 2-step approach of Inverse kinematics to achieve intended movements of the robot.



The main reason why this approach is followed is because the robot that we have considered has a spherical wrist. The main characteristic of spherical wrist is that it only contributes to the orientation of the end effector and not the positioning. The position of the end effector is mainly controlled by the first 3 joints of the robot (arm) θ_1 , θ_2 & θ_3. Hence, we can split the problem of inverse kinematics into two parts where we do the positioning and orientation aspects separately. As shown above, we have adopted inverse velocity kinematics for determining the first 3 joint angles (θ_1 , θ_2 & θ_3) and inverse orientation kinematics (of spherical wrist) to determine the next 3 joint angles (θ_4 , θ_5 & θ_6).

Inverse Velocity Kinematics to obtain θ_1 , θ_2 & θ_3

For the inverse velocity kinematics, first the Jacobian is computed from the DH parameters. The Jacobian in this case will be a function of only the first 3 joint angles, (θ_1 , θ_2 & θ_3). The process is as explained below:

Calculating the matrix T_i

$T_1 = \mathrm{T}_1^0 * \mathrm{T}_2^1 * \mathrm{T}_3^2 * \mathrm{T}_4^3$

$T_2 = \mathrm{T}_1^0 * \mathrm{T}_2^1 * \mathrm{T}_3^2 * \mathrm{T}_4^3 * \mathrm{T}_5^4 * \mathrm{T}_6^5 * \mathrm{T}_7^6$

$T_3 = \mathrm{T}_1^0 * \mathrm{T}_2^1 * \mathrm{T}_3^2 * \mathrm{T}_4^3 * \mathrm{T}_5^4 * \mathrm{T}_6^5 * \mathrm{T}_7^6 * \mathrm{T}_8^7 * \mathrm{T}_9^8$

Calculating Z_i (First 3 row values of the 3rd column of the T matrix)

$$Z_1 = T_1[0:3,2] = \begin{bmatrix} \cos(\theta_1) \\ \sin(\theta_1) \\ 0 \end{bmatrix}$$

$$Z_2 = T_2[0:3,2] = \begin{bmatrix} \sin(\theta_1) \\ -\cos(\theta_1) \\ 0 \end{bmatrix}$$

$$Z_3 = T_3[0:3,2] = \begin{bmatrix} -\sin(\theta_2)\sin(\theta_3)\cos(\theta_1) + \cos(\theta_1)\cos(\theta_2)\cos(\theta_3) \\ -\sin(\theta_1)\sin(\theta_2)\sin(\theta_3) + \sin(\theta_1)\cos(\theta_2)\cos(\theta_3) \\ \sin(\theta_2)\cos(\theta_3) + \sin(\theta_3)\cos(\theta_2) \end{bmatrix}$$

Calculating h(q_1, q_2, ….. q_n) (First 3 row values of the 4rd column of the T matrix representing the transformation from base frame to the origin of the spherical wrist)

h(q_1, q_2, ….. q_n) = T_3 [0:3,3] =

$$\begin{bmatrix} -600\sin(\theta_2)\sin(\theta_3)\cos(\theta_1) - 700\sin(\theta_2)\cos(\theta_1) + 600\cos(\theta_1)\cos(\theta_2)\cos(\theta_3) + 150\cos(\theta_1) + 44.02 \\ -600\sin(\theta_1)\sin(\theta_2)\sin(\theta_3) - 700\sin(\theta_1)\sin(\theta_2) + 600\sin(\theta_1)\cos(\theta_2)\cos(\theta_3) + 150\sin(\theta_1) \\ 600\sin(\theta_2)\cos(\theta_3) + 600\sin(\theta_3)\cos(\theta_2) + 700\cos(\theta_2) + 781.5 \end{bmatrix}$$

Calculating $\partial \mathbf{h}/\partial \mathbf{q}_i$

i. $\partial h/\partial q_1 = \partial h/\partial \theta_1 =$

$$\begin{bmatrix} 600\sin(\theta_1)\sin(\theta_2)\sin(\theta_3) + 700\sin(\theta_1)\sin(\theta_2) - 600\sin(\theta_1)\cos(\theta_2)\cos(\theta_3) - 150\sin(\theta_1) \\ -600\sin(\theta_2)\sin(\theta_3)\cos(\theta_1) - 700\sin(\theta_2)\cos(\theta_1) + 600\cos(\theta_1)\cos(\theta_2)\cos(\theta_3) + 150\cos(\theta_1) \\ 0 \end{bmatrix}$$

ii. $\partial h/\partial q_2 = \partial h/\partial \theta_2 =$

$$\begin{bmatrix} -600\sin(\theta_2)\cos(\theta_1)\cos(\theta_3) - 600\sin(\theta_3)\cos(\theta_1)\cos(\theta_2) - 700\cos(\theta_1)\cos(\theta_2) \\ -600\sin(\theta_1)\sin(\theta_2)\cos(\theta_3) - 600\sin(\theta_1)\sin(\theta_3)\cos(\theta_2) - 700\sin(\theta_1)\cos(\theta_2) \\ -600\sin(\theta_2)\sin(\theta_3) - 700\sin(\theta_2) + 600\cos(\theta_2)\cos(\theta_3) \end{bmatrix}$$

iii. $\partial h/\partial q_3 = \partial h/\partial \theta_3 =$

$$\begin{bmatrix} -600\sin(\theta_2)\cos(\theta_1)\cos(\theta_3) - 600\sin(\theta_3)\cos(\theta_1)\cos(\theta_2) \\ -600\sin(\theta_1)\sin(\theta_2)\cos(\theta_3) - 600\sin(\theta_1)\sin(\theta_3)\cos(\theta_2) \\ -600\sin(\theta_2)\sin(\theta_3) + 600\cos(\theta_2)\cos(\theta_3) \end{bmatrix}$$

Writing the Jacobian Matrix

$$J = \begin{bmatrix} \partial h/\partial \theta_1 & \partial h/\partial \theta_2 & \partial h/\partial \theta_3 \\ Z_1 & Z_2 & Z_3 \end{bmatrix}$$

$$\boxed{\begin{matrix} 600\sin(\theta_1)\sin(\theta_2)\sin(\theta_3) + 700\sin(\theta_1)\sin(\theta_2) - 600\sin(\theta_1)\cos(\theta_2)\cos(\theta_3) - 150\sin(\theta_1) \\ -600\sin(\theta_2)\sin(\theta_3)\cos(\theta_1) - 700\sin(\theta_2)\cos(\theta_1) + 600\cos(\theta_1)\cos(\theta_2)\cos(\theta_3) + 150\cos(\theta_1) \\ 0 \\ \cos(\theta_1) \\ \sin(\theta_1) \\ 0 \end{matrix}}$$

Column 1

$$\boxed{\begin{matrix} -600\sin(\theta_2)\cos(\theta_1)\cos(\theta_3) - 600\sin(\theta_3)\cos(\theta_1)\cos(\theta_2) - 700\cos(\theta_1)\cos(\theta_2) \\ -600\sin(\theta_1)\sin(\theta_2)\cos(\theta_3) - 600\sin(\theta_1)\sin(\theta_3)\cos(\theta_2) - 700\sin(\theta_1)\cos(\theta_2) \\ -600\sin(\theta_2)\sin(\theta_3) - 700\sin(\theta_2) + 600\cos(\theta_2)\cos(\theta_3) \\ \sin(\theta_1) \\ -\cos(\theta_1) \\ 0 \end{matrix}}$$

Column 2

$$\boxed{\begin{matrix} -600\sin(\theta_2)\cos(\theta_1)\cos(\theta_3) - 600\sin(\theta_3)\cos(\theta_1)\cos(\theta_2) \\ -600\sin(\theta_1)\sin(\theta_2)\cos(\theta_3) - 600\sin(\theta_1)\sin(\theta_3)\cos(\theta_2) \\ -600\sin(\theta_2)\sin(\theta_3) + 600\cos(\theta_2)\cos(\theta_3) \\ -\sin(\theta_2)\sin(\theta_3)\cos(\theta_1) + \cos(\theta_1)\cos(\theta_2)\cos(\theta_3) \\ -\sin(\theta_1)\sin(\theta_2)\sin(\theta_3) + \sin(\theta_1)\cos(\theta_2)\cos(\theta_3) \\ \sin(\theta_2)\cos(\theta_3) + \sin(\theta_3)\cos(\theta_2) \end{matrix}}$$

Column 3

The Jacobian is a 6X3 matrix and will be a function of θ_1 , θ_2 & θ_3 only.

Computing the joint angles

- The linear velocities (Vector $\dot{X}$) describing the desired trajectory, is given as an input to the equation:
  $\dot{q} = J^{-1} * \dot{X}$.
- The output of this equation will be a vector of joint angular velocities $\dot{q_1}$, $\dot{q_2}$ & $\dot{q_3}$.
- Here, the initial joint angle values ($q(t)$) are substituted to the Jacobian and as the Jacobian is not a square matrix, we use Pseudo-Inverse concept to compute its inverse.
- Numerical integration method is used to obtain the joint angles q_1, q_2 & q_3 from joint angular velocities $\dot{q_1}$, $\dot{q_2}$ & $\dot{q_3}$

  $$q(t + dt) = q(t) + \dot{q} * dt$$

- From this we obtain the joint angles θ_1 , θ_2 & θ_3 ------> $q_i(t + dt)$

Inverse Orientation Kinematics to obtain θ_4 , θ_5 & θ_6

After obtaining the θ_1 , θ_2 & θ_3 values from inverse velocity kinematics, the joint angle values are substituted in the transformation matrix T_3 and the orientation component of the matrix, R_3^0, is extracted. Here the R_3^0 represents the orientation of the frame at origin of the spherical wrist with respect to the base frame.

R_3^0 =

$$\begin{bmatrix} \sin(\theta_1) & \sin(\theta_2)\cos(\theta_1)\cos(\theta_3) + \sin(\theta_3)\cos(\theta_1)\cos(\theta_2) & -\sin(\theta_2)\sin(\theta_3)\cos(\theta_1) + \cos(\theta_1)\cos(\theta_2)\cos(\theta_3) \\ -\cos(\theta_1) & \sin(\theta_1)\sin(\theta_2)\cos(\theta_3) + \sin(\theta_1)\sin(\theta_3)\cos(\theta_2) & -\sin(\theta_1)\sin(\theta_2)\sin(\theta_3) + \sin(\theta_1)\cos(\theta_2)\cos(\theta_3) \\ 0 & \sin(\theta_2)\sin(\theta_3) - \cos(\theta_2)\cos(\theta_3) & \sin(\theta_2)\cos(\theta_3) + \sin(\theta_3)\cos(\theta_2) \end{bmatrix}$$

The desired orientation of the end effector is defined in the form of a rotation matrix as R_d, with respect to the base frame.

The desired orientation rotation matrix can be represented as: $R_d = R_3^0 * R_6^3$

But the values of R_d and R_3^0 are known numerically, and, from the DH Table, the rotation matrix R_6^3 is given by: R_6^3 = Rotational matrix component of ($^9T_{10}$ *$^{10}T_{11}$ *$^{11}T_{12}$ *$^{12}T_{13}$) =

$$\begin{bmatrix} \sin(\theta_4)\cos(\theta_5)\cos(\theta_6)+\sin(\theta_6)\cos(\theta_4) & -\sin(\theta_4)\sin(\theta_6)\cos(\theta_5)+\cos(\theta_4)\cos(\theta_6) & \sin(\theta_4)\sin(\theta_5) \\ \sin(\theta_4)\sin(\theta_6)-\cos(\theta_4)\cos(\theta_5)\cos(\theta_6) & \sin(\theta_4)\cos(\theta_6)+\sin(\theta_6)\cos(\theta_4)\cos(\theta_5) & -\sin(\theta_5)\cos(\theta_4) \\ -\sin(\theta_5)\cos(\theta_6) & \sin(\theta_5)\sin(\theta_6) & \cos(\theta_5) \end{bmatrix}$$

Substituting, we get:

$R^3_6 = (R^0_3)^{-1} * R_d$ or $R^3_6 = (R^0_3)^T * R_d$

Here the LHS of the equation is the matrix where the elements are a function of θ_4 , θ_5 & θ_6, whereas the RHS is a matrix in numerical form. Therefore, simplifying, we get a system of linear equations which we can solve to get θ_4 , θ_5 & θ_6.

In our case, we used the following equations to compute the values of θ_4 , θ_5 & θ_6.

a. R^3_6 [2,2] = ($(R^0_3)^T * R_d$) [2,2]
   $\cos(\theta_5)$ = ($(R^0_3)^T * R_d$) [2,2]
   $\theta_5 = \cos^{-1}((\ (R^0_3)^T * R_d\ )[2,2])$

b. R^3_6 [2,1] / R^3_6 [2,0] = ($(R^0_3)^T * R_d$)[2,1] / ($(R^0_3)^T * R_d$) [2,0]
   $-tan\theta_6 =$ ($(R^0_3)^T * R_d$) [2,1] / ($(R^0_3)^T * R_d$) [2,0]
   θ_6 = - $\tan^{-1}((\ (R^0_3)^T * R_d\ )[2,1]\ /\ (\ (R^0_3)^T * R_d\ )\ [2,0])$

c. R^3_6 [0,2] / R^3_6 [1,2] = ($(R^0_3)^T * R_d$)[0,2] / ($(R^0_3)^T * R_d$) [0,2]
   $-tan\theta_4 =$ ($(R^0_3)^T * R_d$) [0,2] / ($(R^0_3)^T * R_d$) [0,2]
   θ_4 = - $\tan^{-1}((\ (R^0_3)^T * R_d\ )[2,1]\ /\ (\ (R^0_3)^T * R_d\ )\ [2,0])$

INVERSE KINEMATICS VALIDATION

The inverse kinematics method that we followed and the equations that we obtained were validated by considering the following case:

Case: The robot to expected to move in a straight line in X-Z plane with slope of -3/2. The robot should move by 400 mm in the X direction and -600mm in the Z direction (where X and Z directions are with respect to the base frame). For inverse velocity kinematics, the velocity in X direction is taken as 20mm/sec and the velocity in Z direction is taken as -30mm/sec for a period of 20 seconds.

Method followed:

The velocity matrix is defined as

$$\dot{X} = \begin{bmatrix} 20 \\ 0 \\ -30 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Now from t-0 to t=20, for a timestep of t (t is assumed as 0.1 seconds), we perform the following calculations:

- Substitute current values of (q1 , q2 , q3 , q4 , q5 , q6) to the Jacobian equation J
- Use the Inverse Velocity Kinematics equation to calculate the Joint angular velocities

  $\dot{q}$ = $J^{-1} *$ V

  $$\begin{bmatrix} \dot{q1} \\ \dot{q2} \\ \dot{q3} \\ 0 \\ 0 \\ 0 \end{bmatrix} = J^{-1} * \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- Use the joint angular velocities obtained from the above equation to perform numeric integration and get the new joint angle values.

  i.e. q_{new} = $q_{current}$ + $\dot{q}$ * t

- Plug in the new q values obtained after numeric integration into the transformation matrix T_3 and extract the position of the spherical wrist origin with respect to the base. This is done by extracting the last column of the transformation matrix T_3 after substitution, .i.e. $P_{spherical\ wrist\ origin}$

- Now to get the position of the end effector, use the formula:

  $$P_{end\ effector} = P_{spherical\ wrist\ origin} + d * R * [0 \quad 0 \quad 1]^T$$

  d is the distance from the spherical wrist origin to the end effector
  R is the rotation matrix representing the orientation of the end effector with respect to the base frame.

  In our case,
  d = 65mm + 109.5mm = 174.5 mm
  R = $\begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$

  Hence,

  $$P_{end\ effector} = P_{spherical\ wrist\ origin} + 174.5 * \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

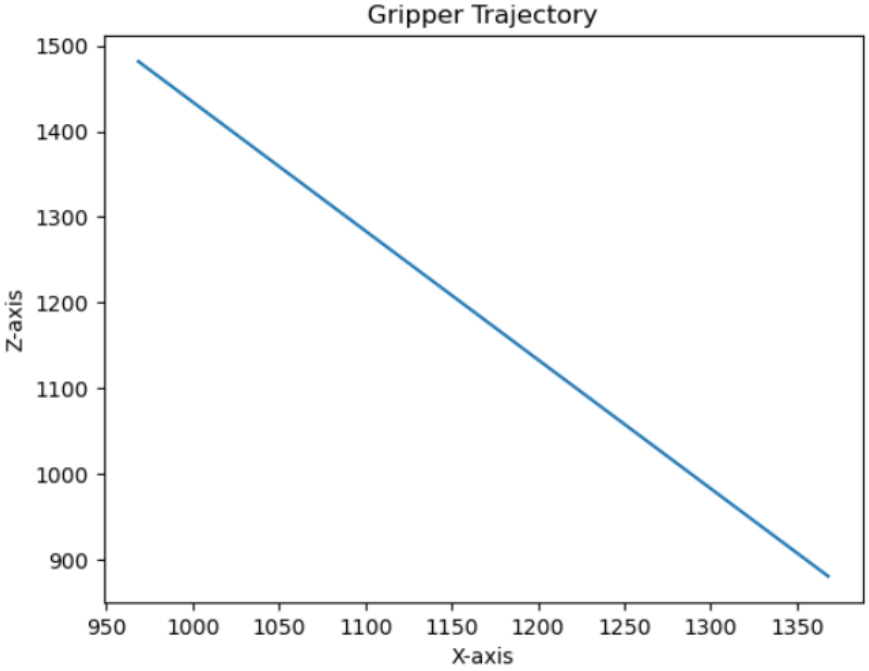
  $$P_{end\ effector} = P_{spherical\ wrist\ origin} + 174.5 * \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

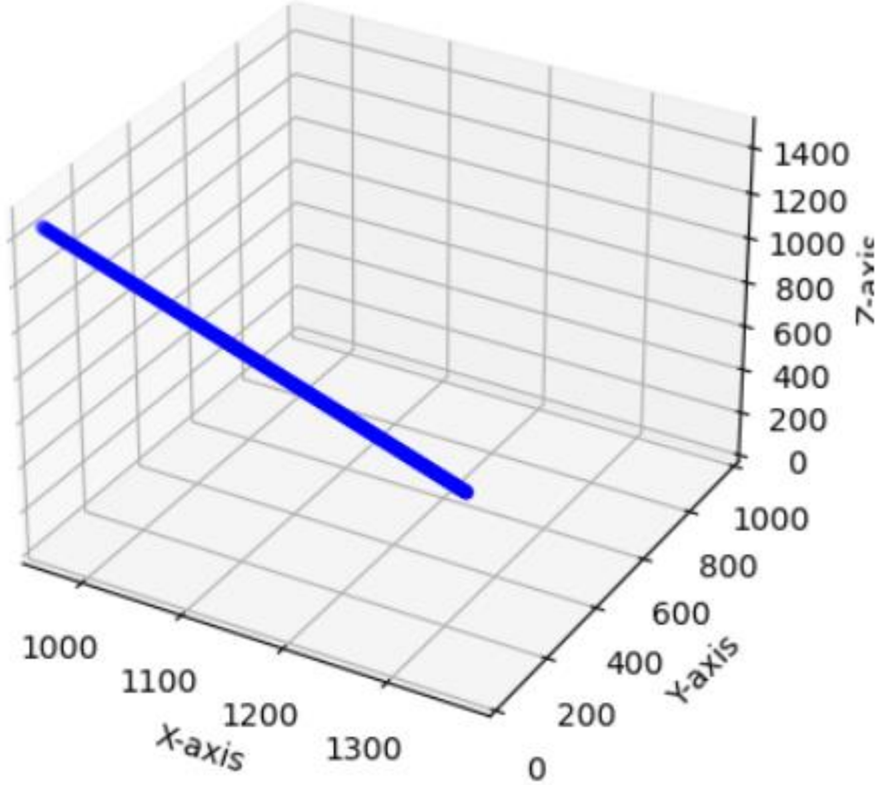
  $$P_{end\ effector} = P_{spherical\ wrist\ origin} + \begin{bmatrix} 174.5 \\ 0 \\ 0 \end{bmatrix}$$

  Note: Here, the $P_{end\ effector}$ is a 3x1 vector representing the X-Y-Z positions of the end effector with respect to the base frame

- The Values of $P_{end\ effector}$ are stored at every iteration and it is plotted at the end.

We obtained the following graphs from the Python Sympy code, for the case considered for inverse kinematics:





Inverse Kinematics Validation from actual simulation of the robot in Gazebo:

- In the Publisher Node script that we set up; the following things are considered:

$$\dot{X} = \begin{bmatrix} 20 \\ 0 \\ -30 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$R_d = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

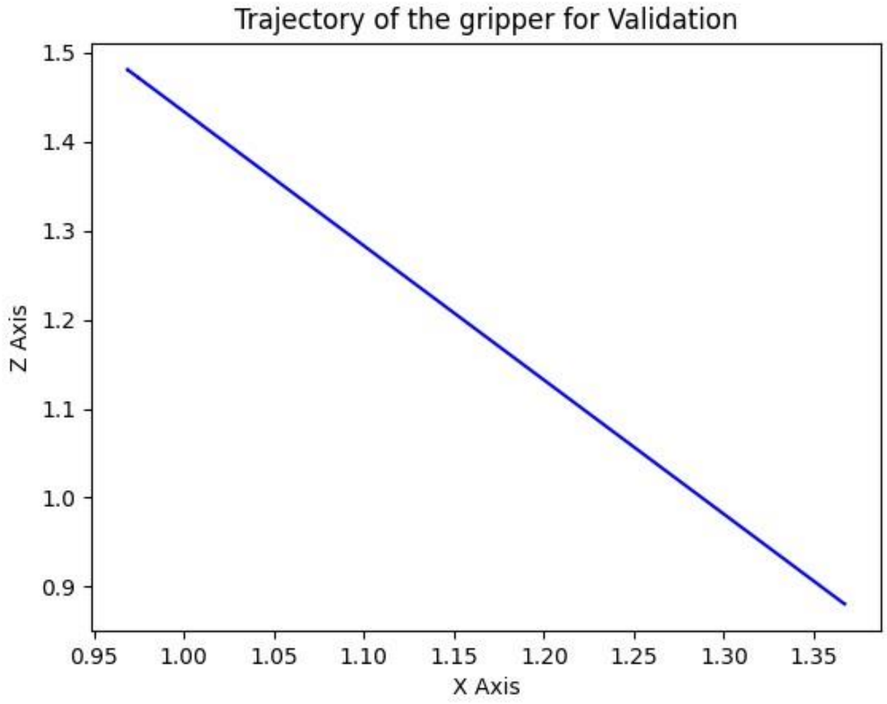
Timestep, dt = 0.1

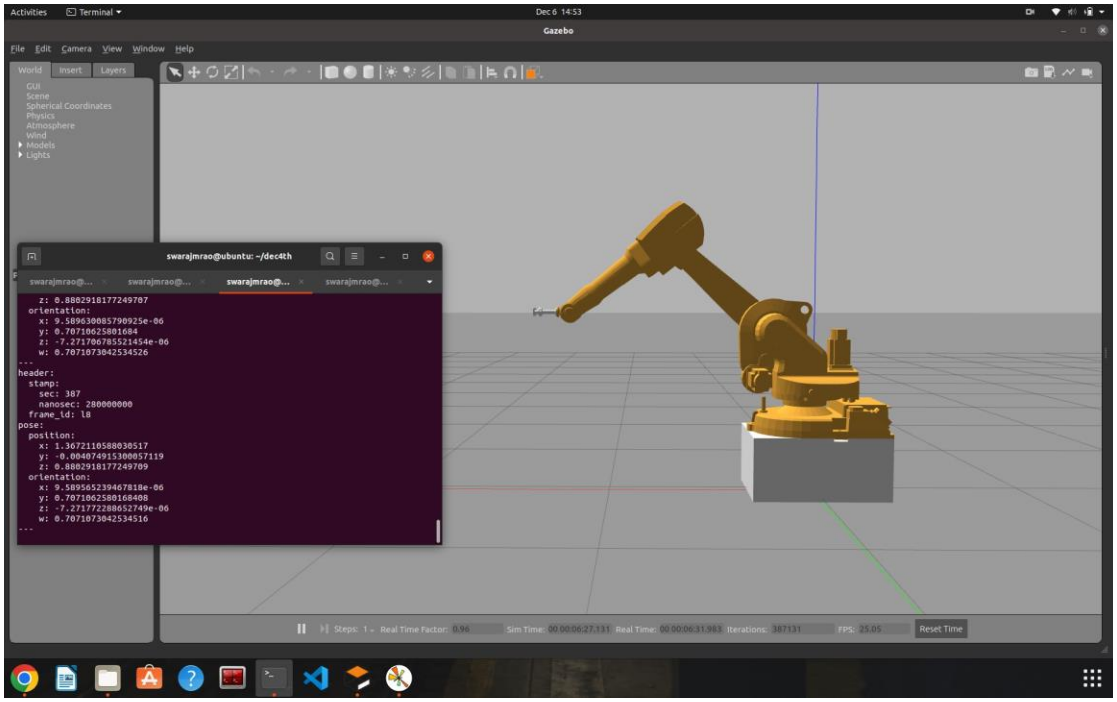
Total time, T = 20 seconds

[q1 , q2 , q3 , q4 , q5 , q6] =

[0.0001, -0.0001 , 0.0001 , -0.00000001 , 0.00000001 , -0.00000001] at (t = 0)

The graph obtained after running the script is as given below





```
    z: 0.8802918177249707
  orientation:
    x: 9.589630085790925e-06
    y: 0.70710625801684
    z: -7.271706785521454e-06
    w: 0.7071073042534526
---
header:
  stamp:
    sec: 387
    nanosec: 280000000
  frame_id: l8
pose:
  position:
    x: 1.3672110588030517
    y: -0.004074915300057119
    z: 0.8802918177249709
  orientation:
    x: 9.589565239467818e-06
    y: 0.7071062580168408
    z: -7.271772288652749e-06
    w: 0.7071073042534516
---
```

It can be observed that the trajectory obtained from the Python Sympy script and the trajectory obtained from ROS Gazebo are the same with no deviation. This further proves that the simulation that we have set up in Gazebo is highly accurate, where both the orientation and the positioning of the end effector is controlled precisely.

The gazebo validation is uploaded here: https://youtu.be/bNBnC_7Ich0

This validates the inverse kinematics method and the equations that we have set up.

SIMULATION

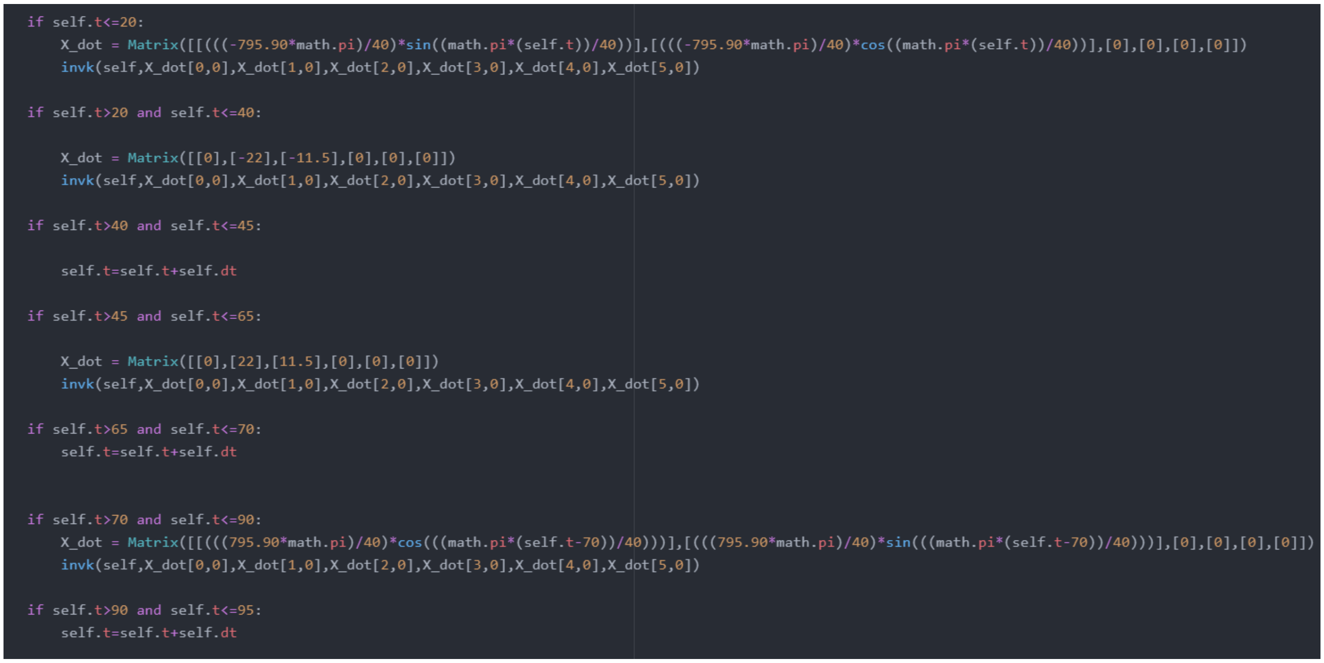
As explained earlier in the report, The workflow involves the robot picking up raw materials from the infeed conveyor and placing them onto the machine for machining. Subsequently, the robot retrieves the machined components and deposits them onto the outfeed conveyor. This process is orchestrated through precise programming to ensure accurate and efficient movements.

The video links of the simulations are attached below:

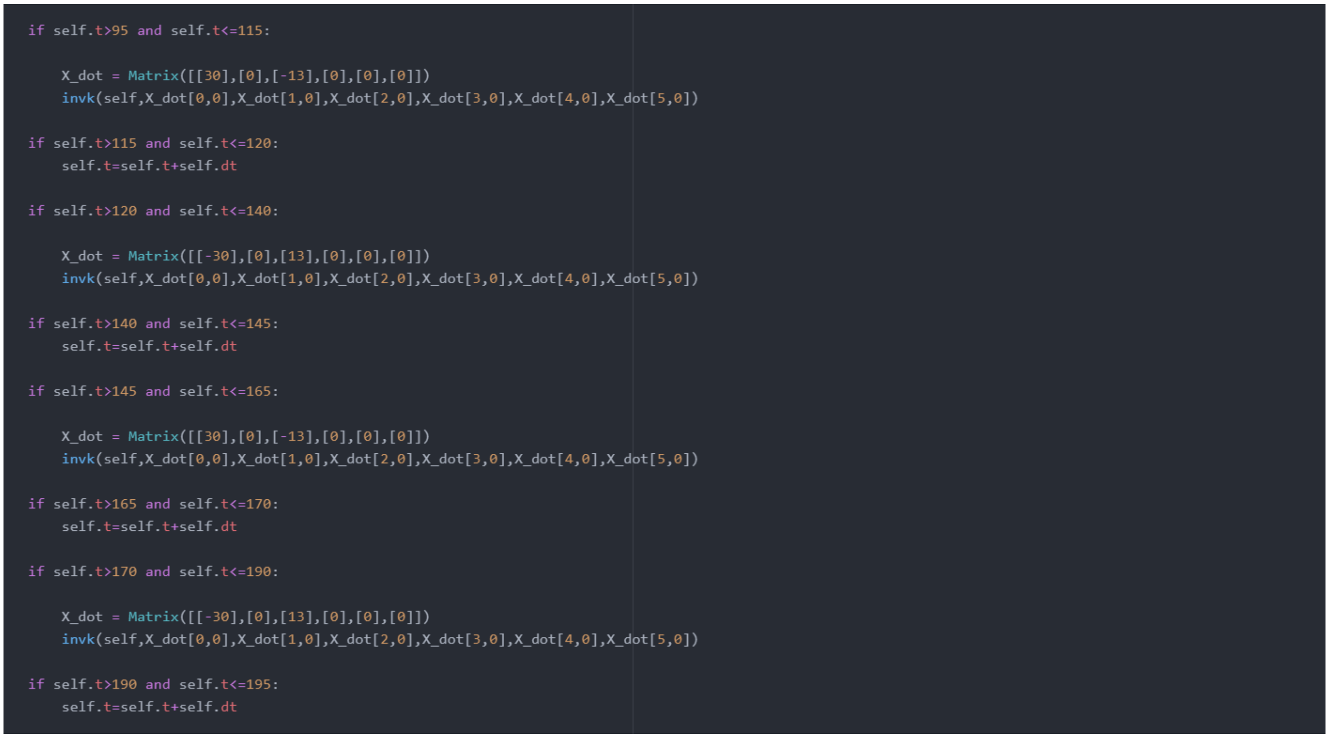
Process Simulation: https://youtu.be/9_zAnCNJitk

Running the trajectory in empty world: https://youtu.be/zwI4443UIOg

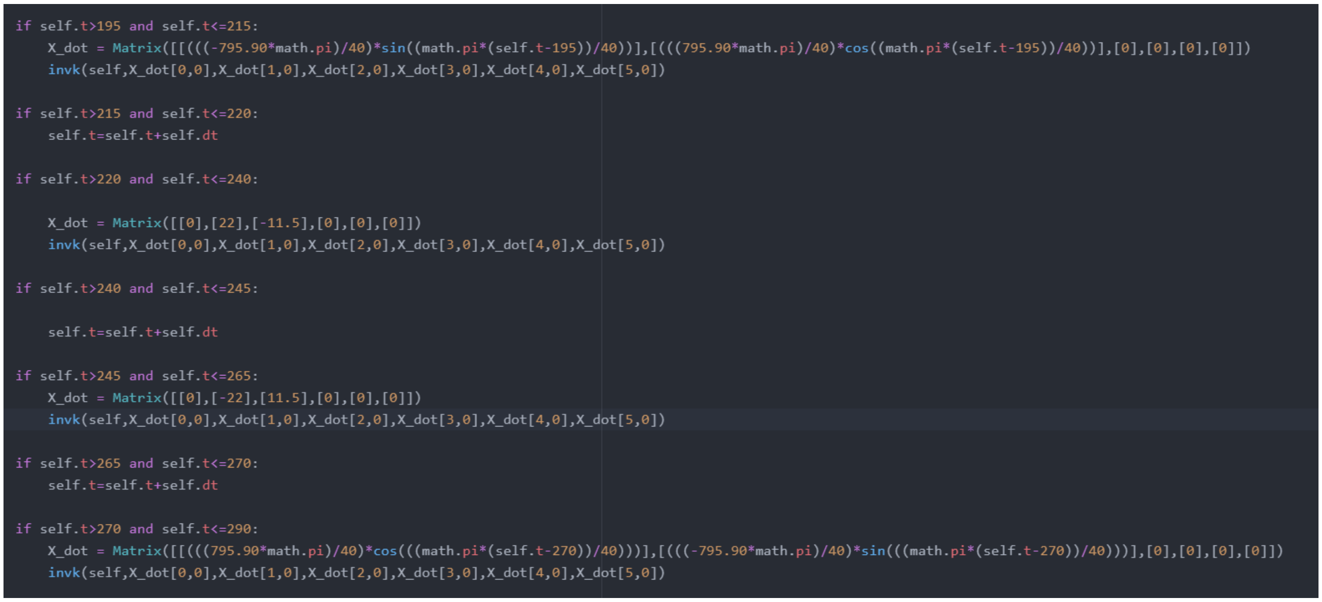
Trajectories followed by the robot:



```
if self.t<=20:
    X_dot = Matrix([[(((-795.90*math.pi)/40)*sin((math.pi*(self.t))/40))],[(((-795.90*math.pi)/40)*cos((math.pi*(self.t))/40))],[0],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])

if self.t>20 and self.t<=40:

    X_dot = Matrix([[0],[-22],[-11.5],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])

if self.t>40 and self.t<=45:

    self.t=self.t+self.dt

if self.t>45 and self.t<=65:

    X_dot = Matrix([[0],[22],[11.5],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])

if self.t>65 and self.t<=70:
    self.t=self.t+self.dt


if self.t>70 and self.t<=90:
    X_dot = Matrix([[(((795.90*math.pi)/40)*cos(((math.pi*(self.t-70))/40)))],[(((795.90*math.pi)/40)*sin(((math.pi*(self.t-70))/40)))],[0],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])

if self.t>90 and self.t<=95:
    self.t=self.t+self.dt
```



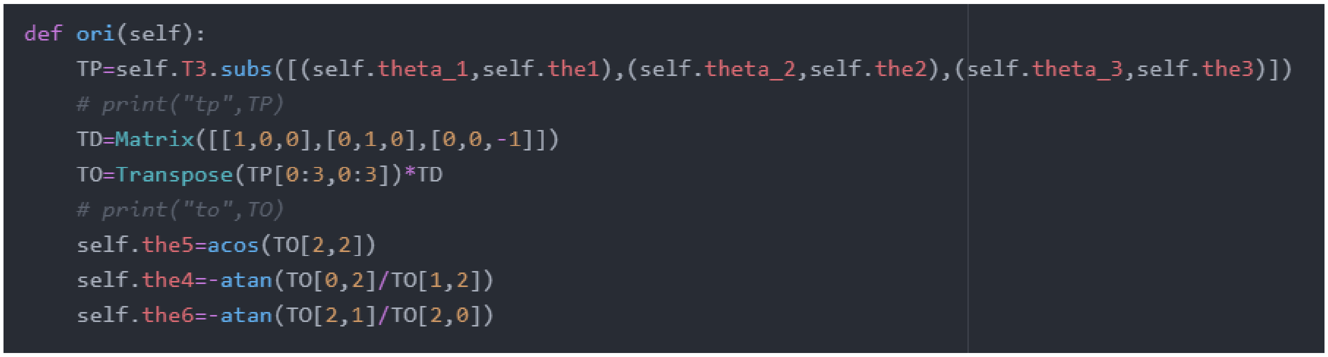
```
if self.t>95 and self.t<=115:

    X_dot = Matrix([[30],[0],[-13],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])

if self.t>115 and self.t<=120:
    self.t=self.t+self.dt

if self.t>120 and self.t<=140:

    X_dot = Matrix([[-30],[0],[13],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])

if self.t>140 and self.t<=145:
    self.t=self.t+self.dt

if self.t>145 and self.t<=165:

    X_dot = Matrix([[30],[0],[-13],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])

if self.t>165 and self.t<=170:
    self.t=self.t+self.dt

if self.t>170 and self.t<=190:

    X_dot = Matrix([[-30],[0],[13],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])

if self.t>190 and self.t<=195:
    self.t=self.t+self.dt
```



```
if self.t>195 and self.t<=215:
    X_dot = Matrix([[(((-795.90*math.pi)/40)*sin((math.pi*(self.t-195))/40))],[(((795.90*math.pi)/40)*cos((math.pi*(self.t-195))/40))],[0],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])

if self.t>215 and self.t<=220:
    self.t=self.t+self.dt

if self.t>220 and self.t<=240:

    X_dot = Matrix([[0],[22],[-11.5],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])

if self.t>240 and self.t<=245:

    self.t=self.t+self.dt

if self.t>245 and self.t<=265:
    X_dot = Matrix([[0],[-22],[11.5],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])

if self.t>265 and self.t<=270:
    self.t=self.t+self.dt

if self.t>270 and self.t<=290:
    X_dot = Matrix([[(((795.90*math.pi)/40)*cos(((math.pi*(self.t-270))/40)))],[(((-795.90*math.pi)/40)*sin(((math.pi*(self.t-270))/40)))],[0],[0],[0],[0]])
    invk(self,X_dot[0,0],X_dot[1,0],X_dot[2,0],X_dot[3,0],X_dot[4,0],X_dot[5,0])
```

Orientation maintained:

$$R_d = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$ ---------------> (End Effector facing exactly down)



```
def ori(self):
    TP=self.T3.subs([(self.theta_1,self.the1),(self.theta_2,self.the2),(self.theta_3,self.the3)])
    # print("tp",TP)
    TD=Matrix([[1,0,0],[0,1,0],[0,0,-1]])
    TO=Transpose(TP[0:3,0:3])*TD
    # print("to",TO)
    self.the5=acos(TO[2,2])
    self.the4=-atan(TO[0,2]/TO[1,2])
    self.the6=-atan(TO[2,1]/TO[2,0])
```

ASSUMPTIONS

The following assumptions are considered:

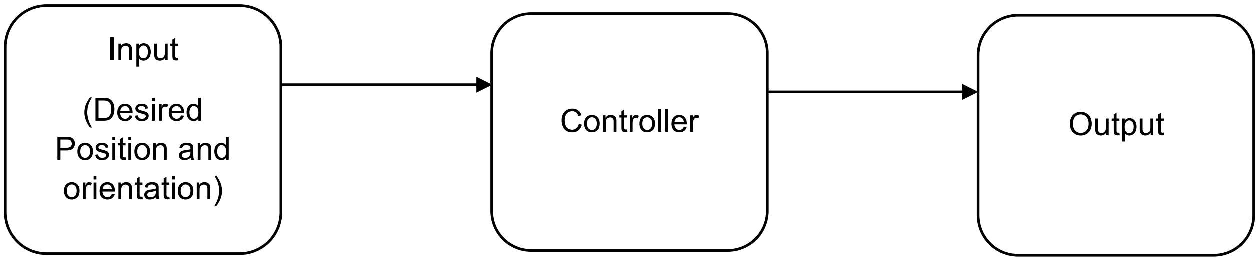
- It is considered that the arm configuration of the 6 DOF Articulated Robot is mainly responsible for the positioning of the end-effector, whereas the spherical wrist configuration is mainly responsible for the orientation of the end-effector.
- All links are considered as rigid bodies and the joints frictionless.
- It is assumed that the simulation time is equal to the real-world time.
- Initial joint positions are considered as small values to avoid singularities in the simulation.
- Pseudo inverse is applied for computing the Jacobian Matrix to avoid singularities.
- We have assumed that the mass of the item being grabbed is lightweight to meet the requirements of the gazebo vacuum gripper plugin. However, the weight of the object is significantly higher, and an actual vacuum gripper would have no difficulty in gripping the part.
- Mass distribution of the manipulator is homogeneous, i.e., the center of mass is located at the geometric center of each link.

CONTROL METHOD

For the management of the robot model, an open-loop controller has been incorporated. This controller operates by directing motion without actively incorporating feedback from the system.

The input to the system is provided in the form of desired position and orientation of end effector. The controller, in turn, utilizes this input to compute the corresponding joint angles. This calculation is instrumental in guiding the robot to follow a predetermined trajectory and ultimately reach the specified final position.

In essence, the open-loop control system described here adheres to a predefined set of instructions, utilizing velocity input to orchestrate the robot's motion through the calculation of joint angles. This approach, although lacking real-time feedback incorporation, serves the purpose of executing precise movements based on the programmed trajectory.



PROBLEMS FACED

Some of the issues we faced while working on the project:

- Unable to save the world file
- Issue with misalignment of axis and frame in URDF Reference while exporting the file from SolidWorks.
- Not able to trigger the vacuum gripper state through the python script.
- Singularity and multiple solutions issue while implementing the inverse velocity kinematics.
- Parts vibrating when the vacuum gripper was activated in world file.
- The vacuum gripper upon activation was unable to grip the part placed from the conveyor.
- Not able to save the world file after building the world in the model builder environment in Gazebo.
- The LIDAR sensor which has been added to a link is not working as there is no topic on ros2 that we can subscribe to, to receive its data.

LESSONS LEARNED

Engaging in this project presented a significant chance to apply our academic expertise in a practical manner. There are numerous insights to be gained from the project. Here are a few examples:

- Choosing the suitable robot for the intended use case.
- By utilizing SOLIDWORKS, we have acquired the ability to model and assemble robot components into the required model.
- Successfully determined the correct method for exporting the assembly using the URDF exporter.
- Employ the created URDF model to simulate it in Gazebo.
- Establishing coordinate frames and determining DH parameters to calculate transformation matrices.
- Utilizing Forward and Inverse Kinematics on the chosen model.
- Generating a virtual environment in Gazebo by adding custom objects
- Execution of odometry and Vacuum Gripper Plugin

CONCLUSIONS

Robots are playing a major role in transforming various industries and shaping the future and to be able to program the robots to perform the required actions programming of a robot is a viral step. Through this project we have mainly demonstrated implementation of velocity and position kinematics to govern the movement of an articulated robot. With a slight tweak to the inverse kinematics equations the robot movements can be modified to perform a variety of tasks.

FUTURE WORK

The simulated environment can be modified to include running conveyors, to which a sensor can be added. This sensor enables the system to obtain the pose of the part or billet on the conveyor. This information can then be utilized to program the robot to pick the part from the moving conveyor.

The same robot can be reconfigured to perform a wide range of operations ranging from arc welding, assembly, painting and coating, palletizing, quality inspection, material removal, laboratory research applications etc.

Also, another degree of freedom can be added to the system in the form of a linear gantry. This can provide benefits such as increased workspace area, optimized floor space area, improved accessibility to the different area within the workspace, enhanced reach and flexibility, reduced risk of interface with objects on the ground, simplified cable management, increased rigidity, improved visibility, and adaptability to various applications.

REFERENCES

1. Mark W. Spong, Seth Hutchinson, and M. Vidyasagar “Robot Modeling and Control” Pages 65 – 146.

1. ABB IRB 1600 - Articulated Robots. Retrieved from https://new.abb.com/products/robotics/robots/articulated-robots/irb-1600

2. ABB IRB 1600 Workspace study retrieved from https://search.abb.com/library/Download.aspx?DocumentID=PR10282EN_R8&LanguageCode=en&DocumentPartId=&Action=Launch

3. Belt Conveyor for World Environment sourced from https://grabcad.com/library/belt-conveyor-49

4. Controller for World Environment sourced from https://grabcad.com/library/abb-robot-irc5-controller-1