Introduction To Robot Modelling (ENPM 662) Final-Project

CYBER-KNIFE

Authors:

Atharva Chandrashekhar Paralikar (117396560)

Sameep Vijay Pote (118218427)

Date: 10th December 2021

Contents

Introduction and Organization 3
Application 3
Robot Type 4
DOFs and Dimensions 4
CAD Model 4
Assumptions 6
Denavit-Hartenberg Parameters 6
Forward Kinematics 7
Inverse Kinematics 8
Forward Kinematics Validation 8
Inverse Kinematics Validation 10
Control Method 14
Workspace 14
Gazebo and Rviz Visualization 15
Problems Faced and Mitigation 16
Lessons Learned 16
Conclusions 16
References 17

Introduction and Organization

This document contains the report for project 2 of subject Introduction to Robot Modelling (ENPM662). The document is divided into a total of 10 different sections and each section will address a particular area of the project. The first section 'Introduction and Organization' describes the overall structure of this report. The section 'Application' describes the application for which the said project revolves around. It also provides a brief description of the motivation behind this topic. The next section 'Robot Description' provides detailed information about the structure of the robot, its physical dimensions, sensors to be used, motors, actuators, and material of the robot. It describes the type of Robot used the next section 'Robot DOFs and Dimensions' describes the geometry of the robot designed and simulated. The next section 'CAD Model' provides an account of the robot's CAD Design using SolidWorks. The next section 'DH Parameters' describes the Denavit-Hartenberg Parameters which are derived using the Sponge's Convention. The next section 'Forward Kinematics' describes the calculation of the transformation matrix. The next section would be 'Inverse Kinematics' and it will state the Jacobian Matrix calculation using code. The next two sections will be the 'Forward Kinematics Validation' and 'Inverse Kinematics Validation'. We Study the Workspace of the Robot in the 'Workspace Study' section which is followed by the 'Assumptions' section where we discuss the assumptions considered in designing and implementing this project. 'Control Method' Section describes the Control method used to control the position and velocity for joints. We will discuss the visualization of the Robot in Gazebo and RViz with screen capture of the robots in the said simulation environments. The next couple of sections, 'Problems faced' and 'Lessons Learned', describe the difficulties and challenges faced during this project and their mitigations. The 'Future Works' section describes the possibilities of further work that can be done on this project. We discuss conclusions and results in the penultimate section. The last section would be the references.

Application

The project revolves around designing and simulating a CyberKnife. CyberKnife is a non-surgical Robotic Radiosurgery system that destroys tumors using highly precise, targeted radiation, with minimal damage to surrounding healthy tissue. Unlike traditional radiation therapy that can take multiple sessions, this therapy is complete in fewer sessions. This therapy is as effective as other radiation therapies but without the side effects. This is a simple painless process and patients can resume normal activity almost immediately. Surgical Robots have gained prominence in recent years due to their accuracy, repeatability, and robustness. Cyber-Knife used in radio surgery has numerous advantages hence studying it is necessary and important. Due to the need of multiple orientations, this problem is a good lesson in handling such real-world problems. This application requires us to design a manipulator to be able to fire the laser beam at a particular point in space from multiple orientations.

Robot Type

The proposed robot design for this application is a 5-DOF manipulator with a Radio-surgery Laser as the last link. The first 4 links are used to position the Laser generator in multiple position around the task-space. The last link can rotate to vector the laser to the direction of desired location from multiple orientations.

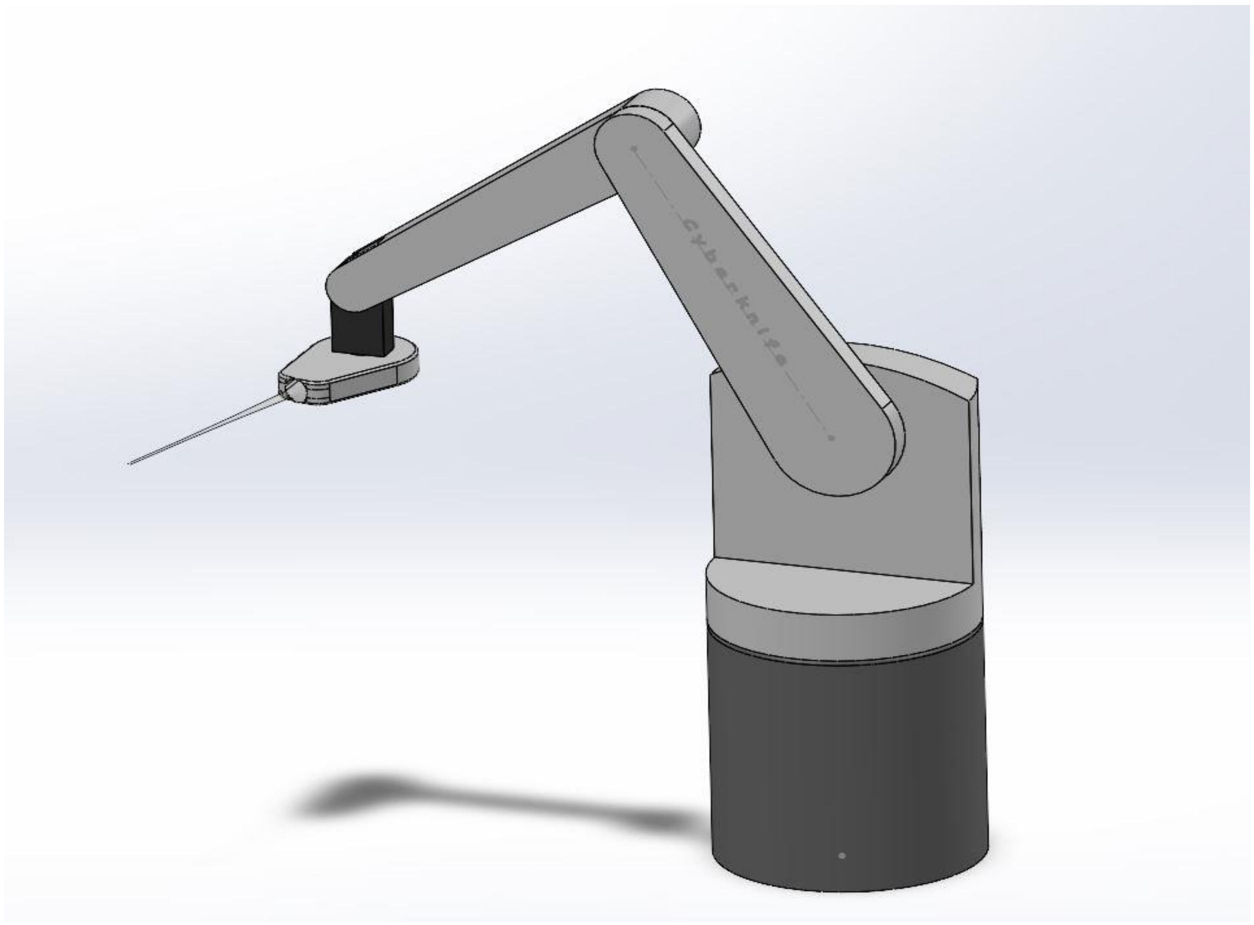
DOFs and Dimensions

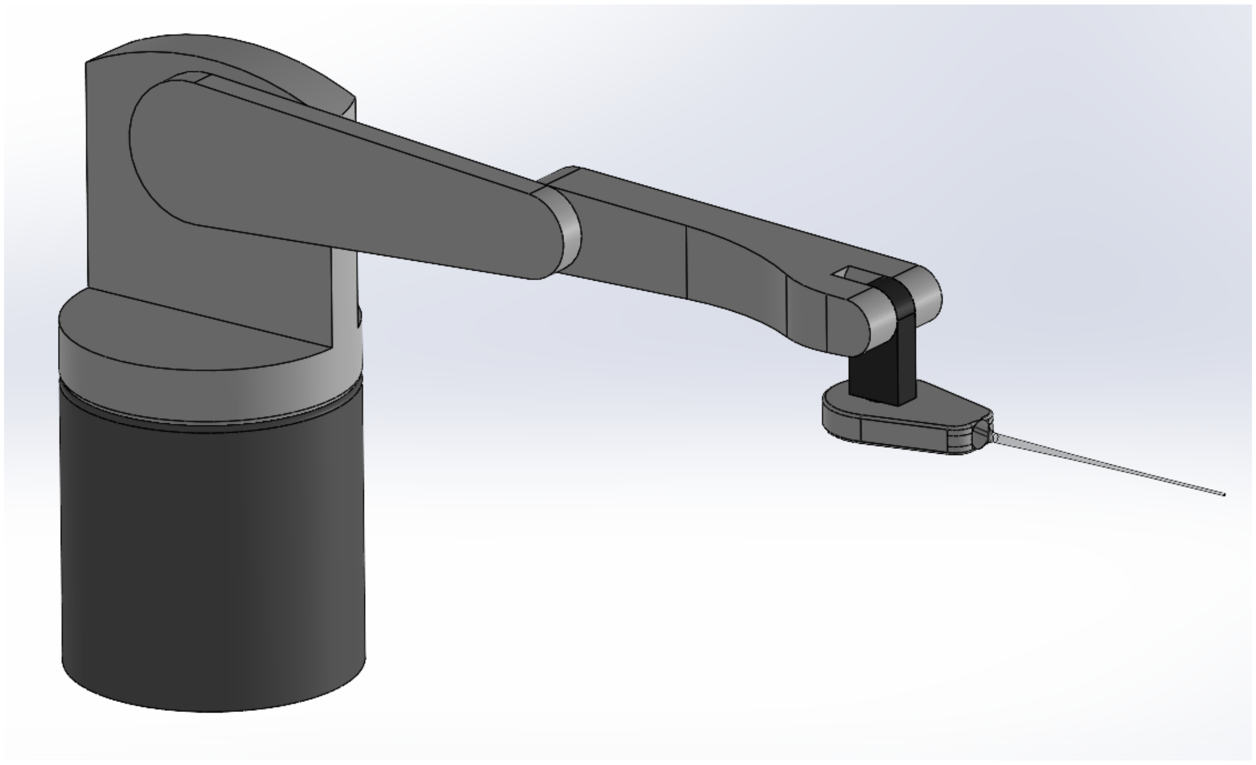
The robot has a total of 5 Degrees of Freedom. The Manipulator can be divided into two parts, 4 link arm with the LASER head at the end for a total of 5 revolute joints. The Robot dimensions are described in the table below:

Parameters	
Size	2324x1055x500mm
Weight	165Kgs
No. of Degrees of Freedom	5
Maximum reach	2080mm
LASER Length (taken from the point of joint)	700mm

CAD Model

The robot was designed using SolidWorks. The configuration of joints was inspired from a real CyberKnife robot. The parts were designed and assembled with comparable dimensions of a real CyberKnife. The URDF was created using the SW2URDF utility. Special care was taken while assigning the coordinate frame at each joint. The exported URDF meshes of the Robot and the URDF file were further used for simulation in Gazebo. The tool outputs all the necessary folders and files to launch the robot in an empty world.


Cyberknife



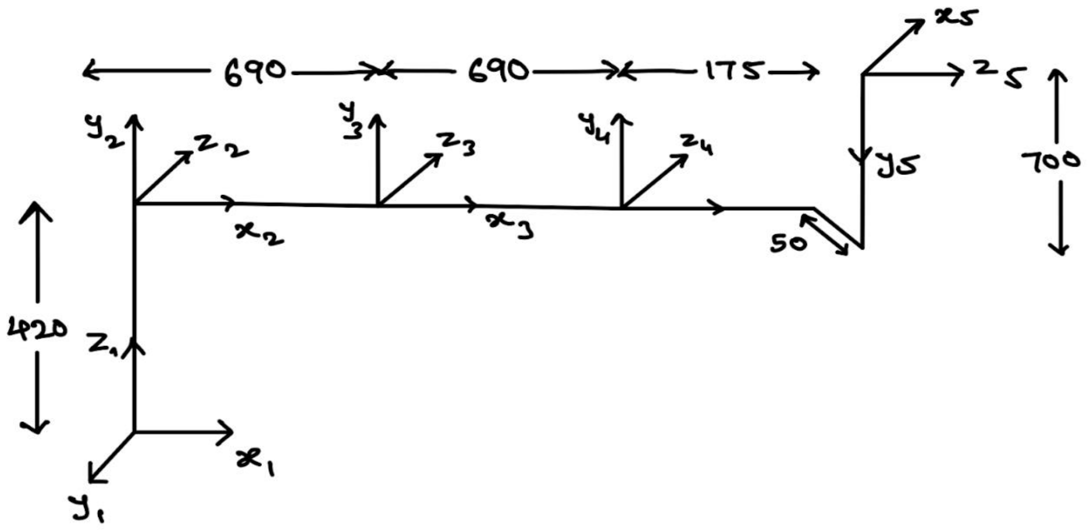
Assumptions

A few assumptions were considered while calculating the kinematics of the Robot. A combination of these assumptions and a few derived requirements of the Robot, we simplified the problem statement. The assumptions taken are as follows:

1. The lengthy point of the end-effector is the representative laser beam, and the length of which is arbitrarily assumed to be 700mm.
2. We assume that the object is placed at a fixed point in the task space during the entire operation.
3. The X-Ray in actual Cyber-knife is replaced by a LiDAR to get position of the representative object.
4. We assume Laser beam is limited to motion in a fixed plane at the height of the object parallel to the X-Y plane. Thus, we only find the possible positions of the manipulator in the plane. However, this can easily be expanded to all possible combinations of orientations are possible.

The above assumptions are taken to reduce the time taken for running simulation for all possible values of joint angles.

Denavit-Hartenberg Parameters



The DH parameters for our robot was calculated using the Sponge's Convention.

	α	a	d	θ
1	$-\frac{\pi}{2}$	0	420	θ_1
2	0	690	0	θ_2
3	0	690	0	θ_3
4	$\frac{\pi}{2}$	175	50	θ_4
5	0	0	700	θ_5

Forward Kinematics

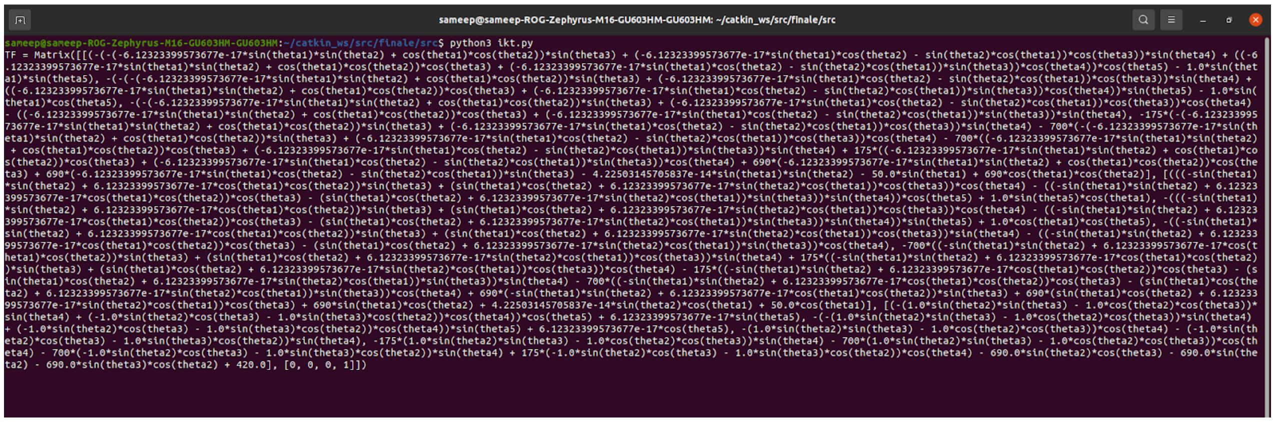
The forward kinematics of any robot can be derived using the DH parameters. We use Sponge's convention to derive these. The transformation matrix for each joint can be written as:

$$T_{i-1}^{i} = \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -s\theta_i c\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The final transformation matrix is given by,

$$T_0^5 = T_0^1 * T_1^2 * T_2^3 * T_3^4 * T_4^5$$

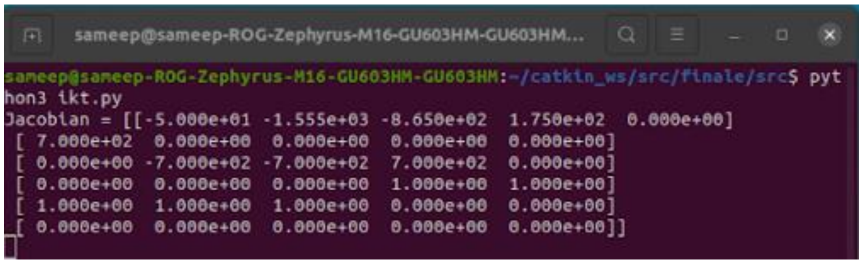
The final transformation matrix for our robot was calculated using a python script. It is given by,


sameep@sameep-ROG-Zephyrus-M16-GU603HM-GU603HM: ~/catkin_ws/src/finale/src

```
sameep@sameep-ROG-Zephyrus-M16-GU603HM-GU603HM:~/catkin_ws/src/finale/src$ python3 ikt.py
TF = Matrix([[(-(-(-6.12323399573677e-17*sin(theta1)*sin(theta2) + cos(theta1)*cos(theta2))*sin(theta3) + (-6.12323399573677e-17*sin(theta1)*cos(theta2) - sin(theta2)*cos(theta1))*cos(theta3))*sin(theta4) + ((-6
.12323399573677e-17*sin(theta1)*sin(theta2) + cos(theta1)*cos(theta2))*cos(theta3) + (-6.12323399573677e-17*sin(theta1)*cos(theta2) - sin(theta2)*cos(theta1))*sin(theta3))*cos(theta4))*cos(theta5) - 1.0*sin(thet
a1)*sin(theta5), -(-(-(-6.12323399573677e-17*sin(theta1)*sin(theta2) + cos(theta1)*cos(theta2))*sin(theta3) + (-6.12323399573677e-17*sin(theta1)*cos(theta2) - sin(theta2)*cos(theta1))*cos(theta3))*sin(theta4) +
((-6.12323399573677e-17*sin(theta1)*sin(theta2) + cos(theta1)*cos(theta2))*cos(theta3) + (-6.12323399573677e-17*sin(theta1)*cos(theta2) - sin(theta2)*cos(theta1))*sin(theta3))*cos(theta4))*sin(theta5) - 1.0*sin(
theta1)*cos(theta5), -(-(-6.12323399573677e-17*sin(theta1)*sin(theta2) + cos(theta1)*cos(theta2))*sin(theta3) + (-6.12323399573677e-17*sin(theta1)*cos(theta2) - sin(theta2)*cos(theta1))*cos(theta3))*cos(theta4)
- ((-6.12323399573677e-17*sin(theta1)*sin(theta2) + cos(theta1)*cos(theta2))*cos(theta3) + (-6.12323399573677e-17*sin(theta1)*cos(theta2) - sin(theta2)*cos(theta1))*sin(theta3))*sin(theta4), -175*(-(-6.123233995
73677e-17*sin(theta1)*sin(theta2) + cos(theta1)*cos(theta2))*sin(theta3) + (-6.12323399573677e-17*sin(theta1)*cos(theta2) - sin(theta2)*cos(theta1))*cos(theta3))*sin(theta4) - 700*(-(-6.12323399573677e-17*sin(th
eta1)*sin(theta2) + cos(theta1)*cos(theta2))*sin(theta3) + (-6.12323399573677e-17*sin(theta1)*cos(theta2) - sin(theta2)*cos(theta1))*cos(theta3))*cos(theta4) - 700*((-6.12323399573677e-17*sin(theta1)*sin(theta2)
 + cos(theta1)*cos(theta2))*cos(theta3) + (-6.12323399573677e-17*sin(theta1)*cos(theta2) - sin(theta2)*cos(theta1))*sin(theta3))*sin(theta4) + 175*((-6.12323399573677e-17*sin(theta1)*sin(theta2) + cos(theta1)*co
s(theta2))*cos(theta3) + (-6.12323399573677e-17*sin(theta1)*cos(theta2) - sin(theta2)*cos(theta1))*sin(theta3))*cos(theta4) + 690*(-6.12323399573677e-17*sin(theta1)*sin(theta2) + cos(theta1)*cos(theta2))*cos(the
ta3) + 690*(-6.12323399573677e-17*sin(theta1)*cos(theta2) - sin(theta2)*cos(theta1))*sin(theta3) - 4.22503145705837e-14*sin(theta1)*sin(theta2) - 50.0*sin(theta1) + 690*cos(theta1)*cos(theta2)], [(((-sin(theta1)
*sin(theta2) + 6.12323399573677e-17*cos(theta1)*cos(theta2))*sin(theta3) + (sin(theta1)*cos(theta2) + 6.12323399573677e-17*sin(theta2)*cos(theta1))*cos(theta3))*cos(theta4) - ((-sin(theta1)*sin(theta2) + 6.12323
399573677e-17*cos(theta1)*cos(theta2))*cos(theta3) - (sin(theta1)*cos(theta2) + 6.12323399573677e-17*sin(theta2)*cos(theta1))*sin(theta3))*sin(theta4))*cos(theta5) + 1.0*sin(theta5)*cos(theta1), -(((-sin(theta1)
*sin(theta2) + 6.12323399573677e-17*cos(theta1)*cos(theta2))*sin(theta3) + (sin(theta1)*cos(theta2) + 6.12323399573677e-17*sin(theta2)*cos(theta1))*cos(theta3))*cos(theta4) - ((-sin(theta1)*sin(theta2) + 6.12323
399573677e-17*cos(theta1)*cos(theta2))*cos(theta3) - (sin(theta1)*cos(theta2) + 6.12323399573677e-17*sin(theta2)*cos(theta1))*sin(theta3))*sin(theta4))*sin(theta5) + 1.0*cos(theta1)*cos(theta5), -((-sin(theta1)*
sin(theta2) + 6.12323399573677e-17*cos(theta1)*cos(theta2))*sin(theta3) + (sin(theta1)*cos(theta2) + 6.12323399573677e-17*sin(theta2)*cos(theta1))*cos(theta3))*sin(theta4) - ((-sin(theta1)*sin(theta2) + 6.123233
99573677e-17*cos(theta1)*cos(theta2))*cos(theta3) - (sin(theta1)*cos(theta2) + 6.12323399573677e-17*sin(theta2)*cos(theta1))*sin(theta3))*cos(theta4), -700*((-sin(theta1)*sin(theta2) + 6.12323399573677e-17*cos(t
heta1)*cos(theta2))*sin(theta3) + (sin(theta1)*cos(theta2) + 6.12323399573677e-17*sin(theta2)*cos(theta1))*cos(theta3))*sin(theta4) + 175*((-sin(theta1)*sin(theta2) + 6.12323399573677e-17*cos(theta1)*cos(theta2)
)*sin(theta3) + (sin(theta1)*cos(theta2) + 6.12323399573677e-17*sin(theta2)*cos(theta1))*cos(theta3))*cos(theta4) - 175*((-sin(theta1)*sin(theta2) + 6.12323399573677e-17*cos(theta1)*cos(theta2))*cos(theta3) - (s
in(theta1)*cos(theta2) + 6.12323399573677e-17*sin(theta2)*cos(theta1))*sin(theta3))*sin(theta4) - 700*((-sin(theta1)*sin(theta2) + 6.12323399573677e-17*cos(theta1)*cos(theta2))*cos(theta3) - (sin(theta1)*cos(the
ta2) + 6.12323399573677e-17*sin(theta2)*cos(theta1))*sin(theta3))*cos(theta4) + 690*(-sin(theta1)*sin(theta2) + 6.12323399573677e-17*cos(theta1)*cos(theta2))*sin(theta3) + 690*(sin(theta1)*cos(theta2) + 6.123233
99573677e-17*sin(theta2)*cos(theta1))*cos(theta3) + 690*sin(theta1)*cos(theta2) + 4.22503145705837e-14*sin(theta2)*cos(theta1) + 50.0*cos(theta1)], [(-(1.0*sin(theta2)*sin(theta3) - 1.0*cos(theta2)*cos(theta3))*
sin(theta4) + (-1.0*sin(theta2)*cos(theta3) - 1.0*sin(theta3)*cos(theta2))*cos(theta4))*cos(theta5) + 6.12323399573677e-17*sin(theta5), -(-(1.0*sin(theta2)*sin(theta3) - 1.0*cos(theta2)*cos(theta3))*sin(theta4)
+ (-1.0*sin(theta2)*cos(theta3) - 1.0*sin(theta3)*cos(theta2))*cos(theta4))*sin(theta5) + 6.12323399573677e-17*cos(theta5), -(1.0*sin(theta2)*sin(theta3) - 1.0*cos(theta2)*cos(theta3))*cos(theta4) - (-1.0*sin(th
eta2)*cos(theta3) - 1.0*sin(theta3)*cos(theta2))*sin(theta4), -175*(1.0*sin(theta2)*sin(theta3) - 1.0*cos(theta2)*cos(theta3))*sin(theta4) - 700*(1.0*sin(theta2)*sin(theta3) - 1.0*cos(theta2)*cos(theta3))*cos(th
eta4) - 700*(-1.0*sin(theta2)*cos(theta3) - 1.0*sin(theta3)*cos(theta2))*sin(theta4) + 175*(-1.0*sin(theta2)*cos(theta3) - 1.0*sin(theta3)*cos(theta2))*cos(theta4) - 690.0*sin(theta2)*cos(theta3) - 690.0*sin(the
ta2) - 690.0*sin(theta3)*cos(theta2) + 420.0], [0, 0, 0, 1]])
```

Inverse Kinematics

The Inverse kinematics was calculated using the Jacobian method using a python script. The Jacobian was calculated as follows:

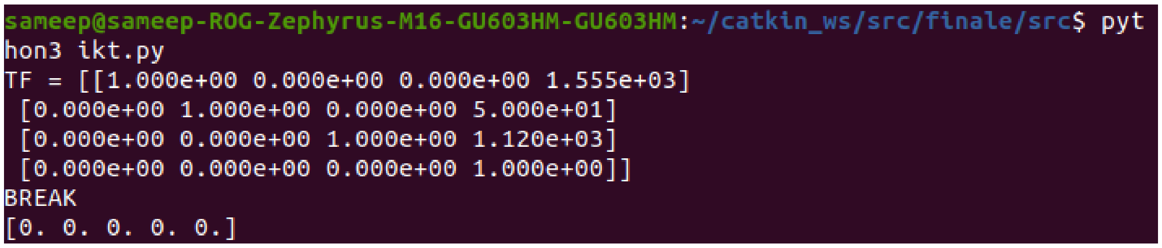


```
sameep@sameep-ROG-Zephyrus-M16-GU603HM-GU603HM:~/catkin_ws/src/finale/src$ pyt
hon3 ikt.py
Jacobian = [[-5.000e+01 -1.555e+03 -8.650e+02  1.750e+02  0.000e+00]
 [ 7.000e+02  0.000e+00  0.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00 -7.000e+02 -7.000e+02  7.000e+02  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  1.000e+00  1.000e+00]
 [ 1.000e+00  1.000e+00  1.000e+00  0.000e+00  0.000e+00]
 [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00  0.000e+00]]
```

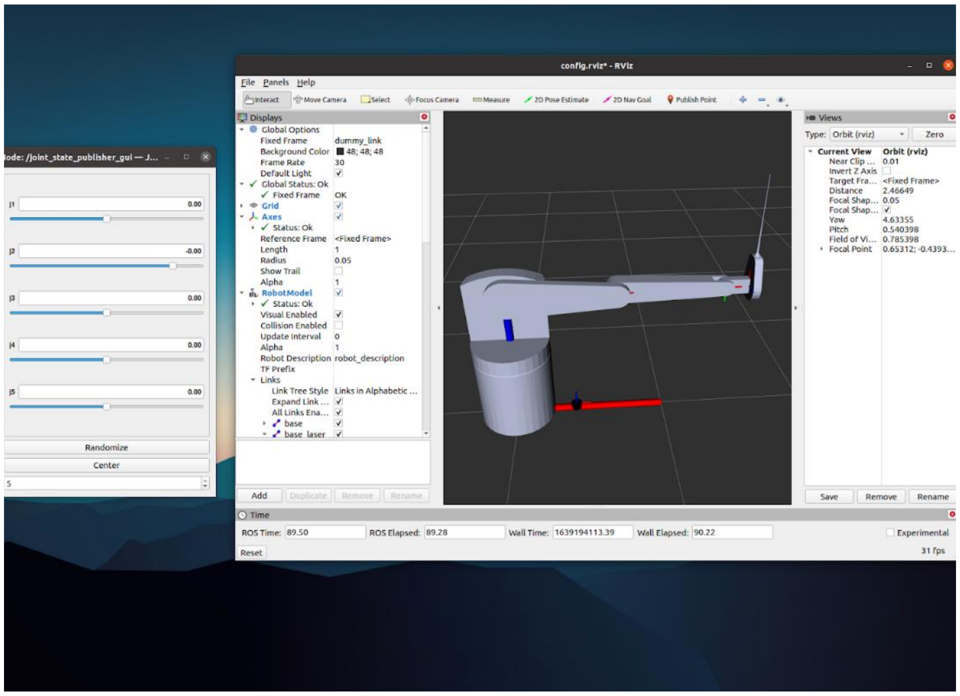
Forward Kinematics Validation

Forward kinematics was validated using the inbuilt joint State Publisher GUI in Rviz for know positions of robot. The Joint angles can be entered manually, and the robot moves in Rviz to the final position thus the forward kinematics can be validated.

Position 1: (theta = [0,0,0,0,0])

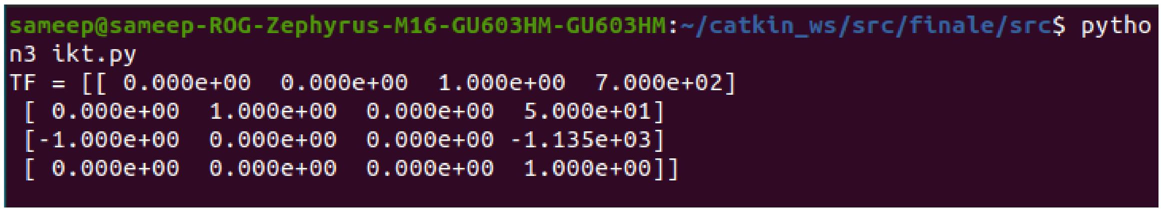


```
sameep@sameep-ROG-Zephyrus-M16-GU603HM-GU603HM:~/catkin_ws/src/finale/src$ pyt
hon3 ikt.py
TF = [[1.000e+00 0.000e+00 0.000e+00 1.555e+03]
 [0.000e+00 1.000e+00 0.000e+00 5.000e+01]
 [0.000e+00 0.000e+00 1.000e+00 1.120e+03]
 [0.000e+00 0.000e+00 0.000e+00 1.000e+00]]
BREAK
[0. 0. 0. 0. 0.]
```

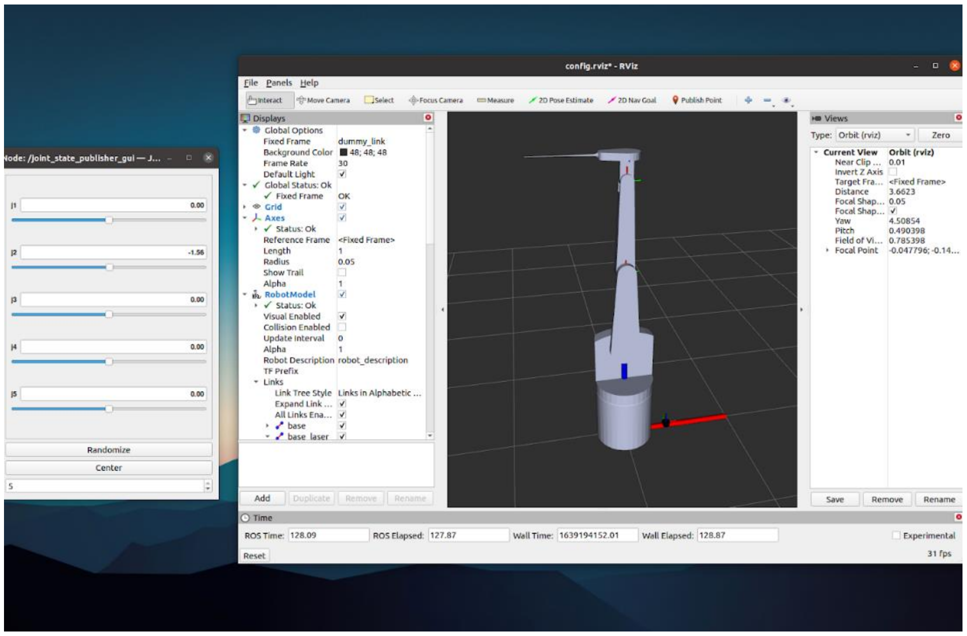


```
Node: /joint_state_publisher_gui — J...
j1  0.00
j2  -0.00
j3  0.00
j4  0.00
j5  0.00
5

Displays
Global Options
  Fixed Frame          dummy_link
  Background Color     48; 48; 48
  Frame Rate           30
  Default Light        ✓
Global Status: Ok
  Fixed Frame          OK
Grid                   ✓
Axes                   ✓
  Status: Ok
  Reference Frame      <Fixed Frame>
  Length               1
  Radius               0.05
  Show Trail
  Alpha                1
RobotModel             ✓
  Status: Ok
  Visual Enabled       ✓
  Collision Enabled
  Update Interval      0
  Alpha                1
  Robot Description    robot_description
  TF Prefix
  Links
    Link Tree Style    Links in Alphabetic ...
    Expand Link ...    ✓
    All Links Ena...   ✓
    base               ✓
    base_laser         ✓

Views
Type: Orbit (rviz)   Zero
Current View     Orbit (rviz)
  Near Clip ...  0.01
  Invert Z Axis
  Target Fra...  <Fixed Frame>
  Distance       2.46649
  Focal Shap...  0.05
  Focal Shap...  ✓
  Yaw            4.63355
  Pitch          0.540398
  Field of Vi... 0.785398
  Focal Point    0.65312; -0.4393...

Time
ROS Time: 89.50   ROS Elapsed: 89.28   Wall Time: 1639194113.39   Wall Elapsed: 90.22   Experimental
31 fps
```

Position 2: (theta = [0,0,0,0,0])



```
sameep@sameep-ROG-Zephyrus-M16-GU603HM-GU603HM:~/catkin_ws/src/finale/src$ pytho
n3 ikt.py
TF = [[ 0.000e+00  0.000e+00  1.000e+00  7.000e+02]
 [ 0.000e+00  1.000e+00  0.000e+00  5.000e+01]
 [-1.000e+00  0.000e+00  0.000e+00 -1.135e+03]
 [ 0.000e+00  0.000e+00  0.000e+00  1.000e+00]]
```



```
Node: /joint_state_publisher_gui — J...
j1  0.00
j2  -1.56
j3  0.00
j4  0.00
j5  0.00
5

Displays
Global Options
  Fixed Frame          dummy_link
  Background Color     48; 48; 48
  Frame Rate           30
  Default Light        ✓
Global Status: Ok
  Fixed Frame          OK
Grid                   ✓
Axes                   ✓
  Status: Ok
  Reference Frame      <Fixed Frame>
  Length               1
  Radius               0.05
  Show Trail
  Alpha                1
RobotModel             ✓
  Status: Ok
  Visual Enabled       ✓
  Collision Enabled
  Update Interval      0
  Alpha                1
  Robot Description    robot_description
  TF Prefix
  Links
    Link Tree Style    Links in Alphabetic ...
    Expand Link ...    ✓
    All Links Ena...   ✓
    base               ✓
    base_laser         ✓

Views
Type: Orbit (rviz)   Zero
Current View     Orbit (rviz)
  Near Clip ...  0.01
  Invert Z Axis
  Target Fra...  <Fixed Frame>
  Distance       3.6623
  Focal Shap...  0.05
  Focal Shap...  ✓
  Yaw            4.50854
  Pitch          0.490398
  Field of Vi... 0.785398
  Focal Point    -0.047796; -0.14...

Time
ROS Time: 128.09   ROS Elapsed: 127.87   Wall Time: 1639194152.01   Wall Elapsed: 128.87   Experimental
31 fps
```

Inverse Kinematics Validation

Inverse Kinematics Equation:

$$x = x_d - 700\cos(\alpha)$$
$$y = y_d - 700\sin(\alpha)$$

$$\theta_1 = \tan^{-1}(y/x)$$

$$d = \sqrt{x^2 + y^2}$$

$$x_3 = d - (L_3 \cos\phi)$$
$$y_3 = z_d - (L_3 \sin\phi)$$

$$\Delta = x_3^2 + y_3^2$$

$$c_3 = (\Delta - L_1^2 - L_2^2) / 2L_1L_2$$

$$s_3 = \sqrt{1 - c_3^2}$$

$$\theta_3 = \tan^{-1}(s_3/c_3)$$

$$c_2 = \frac{((L_1 + l_2 c_3)\, x_3) + (L_2 y_3 s_3)}{\Delta}$$

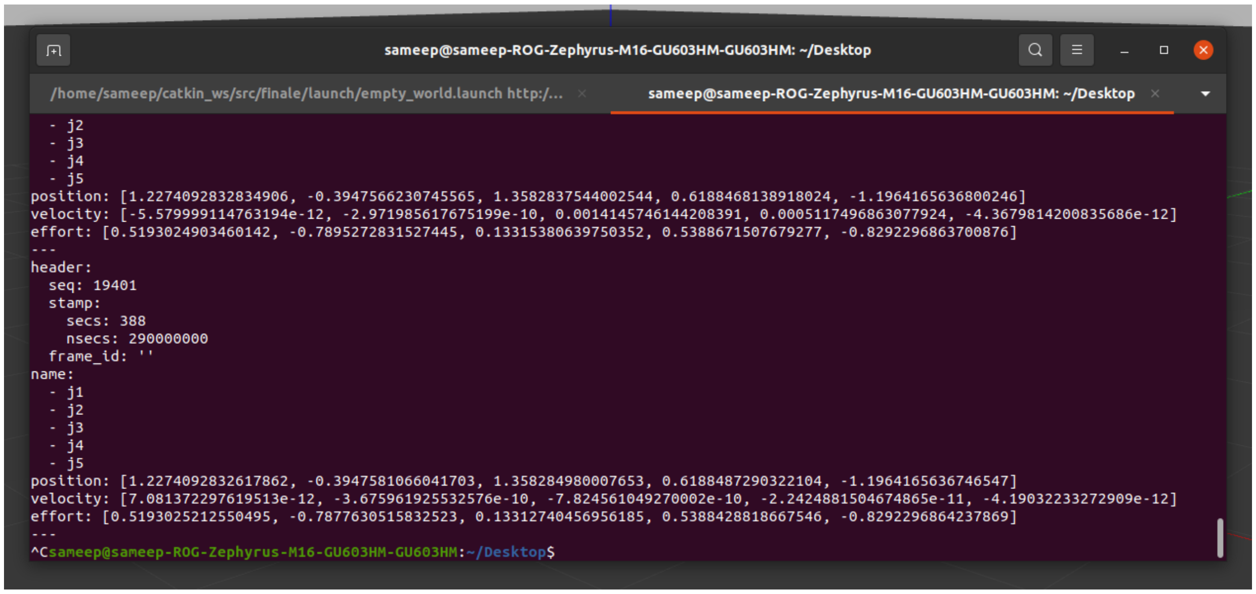
$$s_2 = \frac{((L_1 + l_2 c_3)\, y_3) - (L_2 x_3 s_3)}{\Delta}$$

$$\theta_2 = \tan^{-1}(s_2/c_2)$$

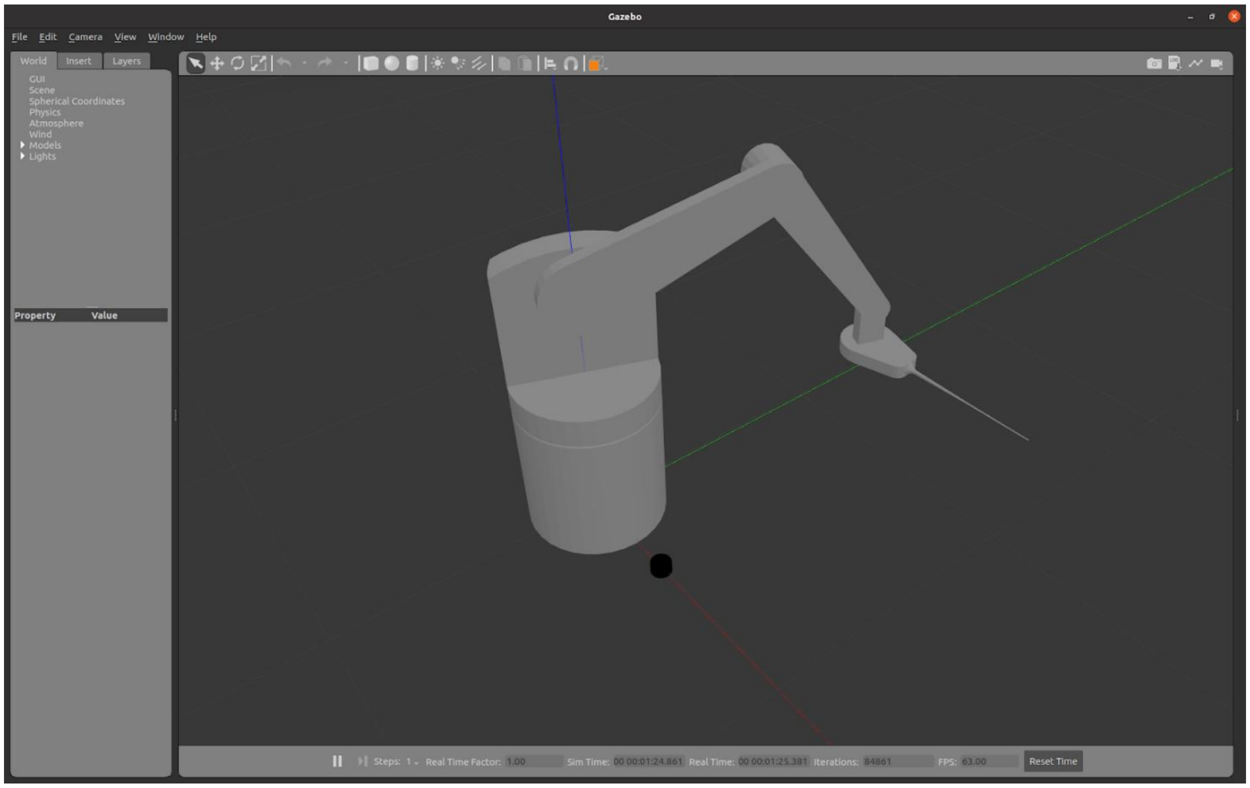
$$\theta_4 = \phi - \theta_2 - \theta_3$$

$$\theta_5 = \tan^{-1}\left(\frac{y_d - y}{x_d - x}\right) - \theta_1$$

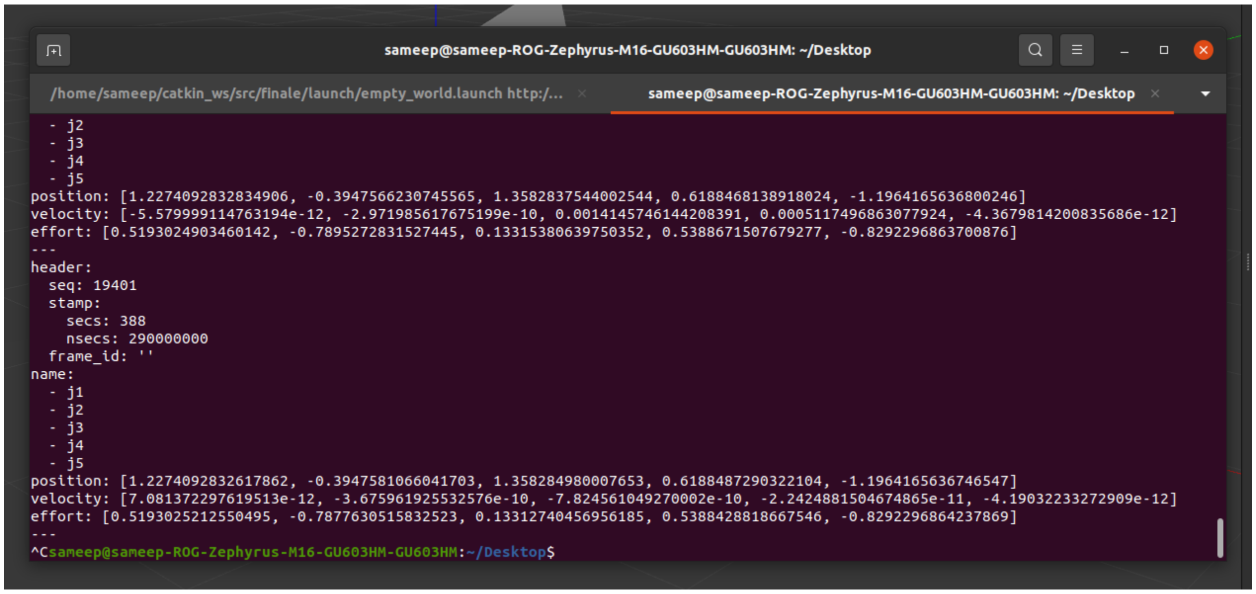
Position 1: (Orientation = 0 degrees)



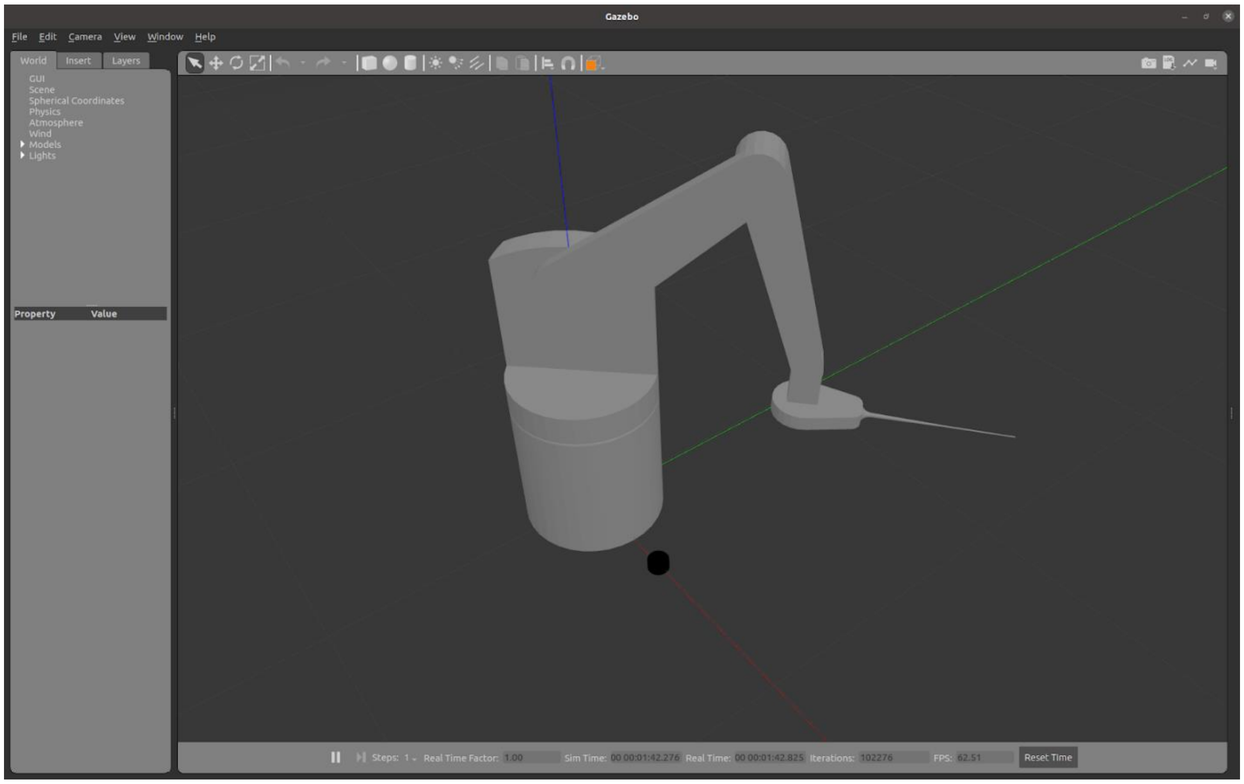
```
  - j2
  - j3
  - j4
  - j5
position: [1.2274092832834906, -0.3947566230745565, 1.3582837544002544, 0.6188468138918024, -1.1964165636800246]
velocity: [-5.579999114763194e-12, -2.971985617675199e-10, 0.0014145746144208391, 0.0005117496863077924, -4.3679814200835686e-12]
effort: [0.5193024903460142, -0.7895272831527445, 0.13315380639750352, 0.5388671507679277, -0.8292296863700876]
---
header:
  seq: 19401
  stamp:
    secs: 388
    nsecs: 290000000
  frame_id: ''
name:
  - j1
  - j2
  - j3
  - j4
  - j5
position: [1.2274092832617862, -0.3947581066041703, 1.358284980007653, 0.6188487290322104, -1.1964165636746547]
velocity: [7.081372297619513e-12, -3.675961925532576e-10, -7.824561049270002e-10, -2.2424881504674865e-11, -4.19032233272909e-12]
effort: [0.5193025212550495, -0.7877630515832523, 0.13312740456956185, 0.5388428818667546, -0.8292296864237869]
---
^Csameep@sameep-ROG-Zephyrus-M16-GU603HM-GU603HM:~/Desktop$
```


Gazebo
File Edit Camera View Window Help
World Insert Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
Lights
Property Value
Steps: 1 Real Time Factor: 1.00 Sim Time: 00 00:01:24.861 Real Time: 00 00:01:25.381 Iterations: 84861 FPS: 63.00 Reset Time

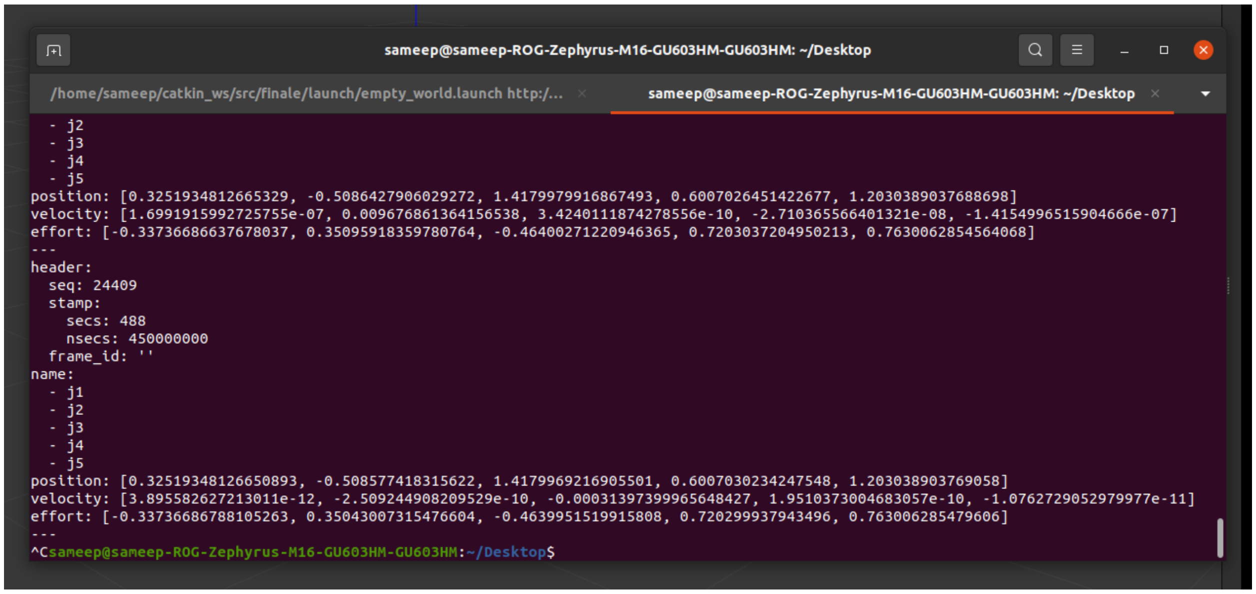
Position 2: (Orientation = 45 degrees)



```
  - j2
  - j3
  - j4
  - j5
position: [1.2274092832834906, -0.3947566230745565, 1.3582837544002544, 0.6188468138918024, -1.1964165636800246]
velocity: [-5.579999114763194e-12, -2.971985617675199e-10, 0.0014145746144208391, 0.0005117496863077924, -4.3679814200835686e-12]
effort: [0.5193024903460142, -0.7895272831527445, 0.13315380639750352, 0.5388671507679277, -0.8292296863700876]
---
header:
  seq: 19401
  stamp:
    secs: 388
    nsecs: 290000000
  frame_id: ''
name:
  - j1
  - j2
  - j3
  - j4
  - j5
position: [1.2274092832617862, -0.3947581066041703, 1.358284980007653, 0.6188487290322104, -1.1964165636746547]
velocity: [7.081372297619513e-12, -3.675961925532576e-10, -7.824561049270002e-10, -2.2424881504674865e-11, -4.19032233272909e-12]
effort: [0.5193025212550495, -0.7877630515832523, 0.13312740456956185, 0.5388428818667546, -0.8292296864237869]
---
^Csameep@sameep-ROG-Zephyrus-M16-GU603HM-GU603HM:~/Desktop$
```

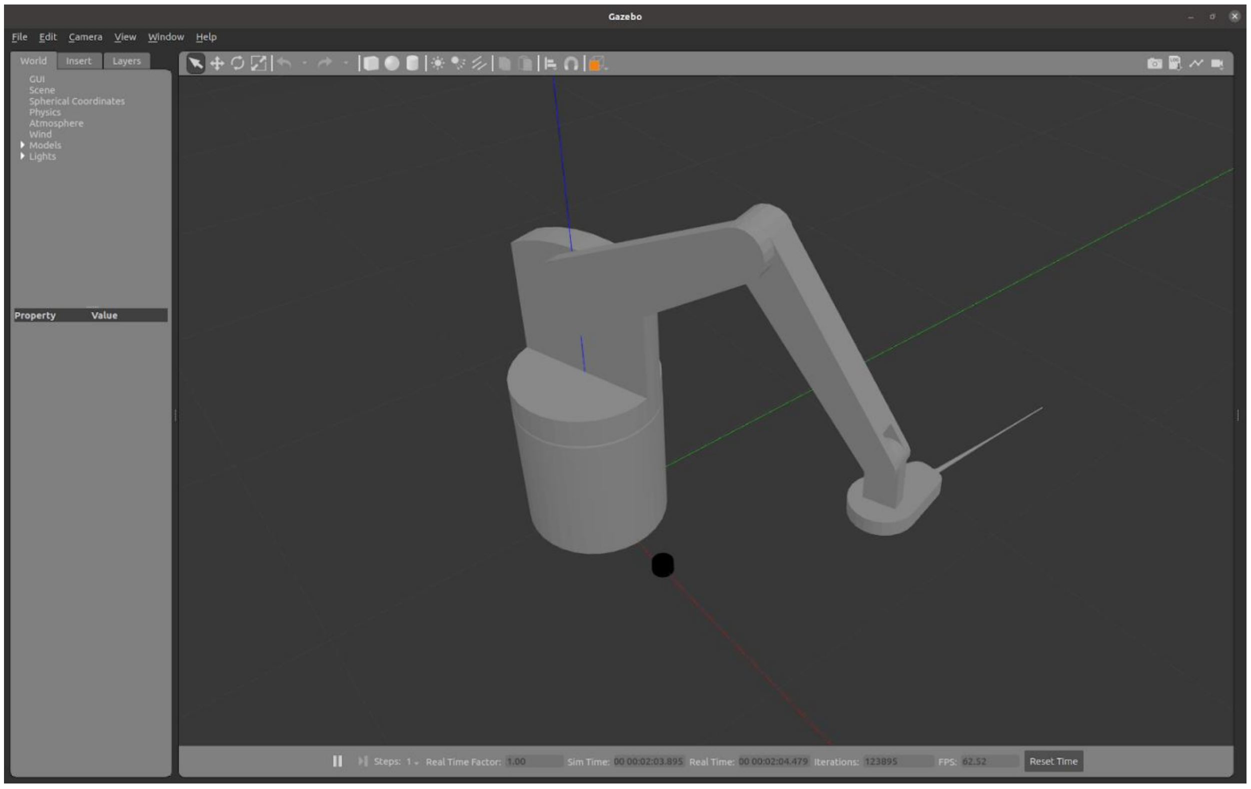


Position 3: (Orientation = 90 degrees)



sameep@sameep-ROG-Zephyrus-M16-GU603HM-GU603HM: ~/Desktop
/home/sameep/catkin_ws/src/finale/launch/empty_world.launch http:/...
sameep@sameep-ROG-Zephyrus-M16-GU603HM-GU603HM: ~/Desktop

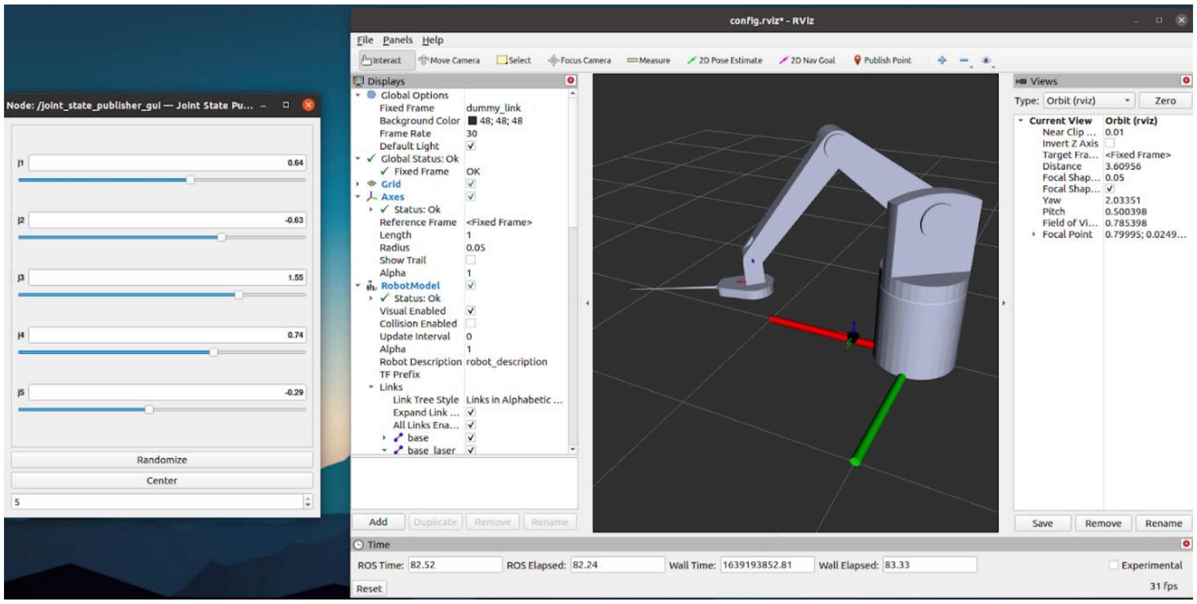
```
  - j2
  - j3
  - j4
  - j5
position: [0.3251934812665329, -0.5086427906029272, 1.4179979916867493, 0.6007026451422677, 1.2030389037688698]
velocity: [1.6991915992725755e-07, 0.009676861364156538, 3.4240111874278556e-10, -2.710365566401321e-08, -1.4154996515904666e-07]
effort: [-0.33736686637678037, 0.35095918359780764, -0.46400271220946365, 0.7203037204950213, 0.7630062854564068]
---
header:
  seq: 24409
  stamp:
    secs: 488
    nsecs: 450000000
  frame_id: ''
name:
  - j1
  - j2
  - j3
  - j4
  - j5
position: [0.32519348126650893, -0.508577418315622, 1.4179969216905501, 0.6007030234247548, 1.203038903769058]
velocity: [3.895582627213011e-12, -2.509244908209529e-10, -0.00031397399965648427, 1.9510373004683057e-10, -1.0762729052979977e-11]
effort: [-0.33736686788105263, 0.35043007315476604, -0.4639951519915808, 0.720299937943496, 0.763006285479606]
---
^Csameep@sameep-ROG-Zephyrus-M16-GU603HM-GU603HM:~/Desktop$
```



Gazebo
File Edit Camera View Window Help
World Insert Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
Lights
Property Value
Steps: 1 Real Time Factor: 1.00 Sim Time: 00 00:02:03.895 Real Time: 00 00:02:04.479 Iterations: 123895 FPS: 62.52 Reset Time

Control Method

Control Methods used in our project for controlling all the joins of the CyberKnife are Joint Position Controller from effort controllers and Joint State Controller. The Joint Position Controller commands a desired position to the joint. The position is controlled using PID control to specify the effort to the joint. When started, the controller will command to the current position. The joint state controller is used to simulate the prototype in Rviz using the joint state publisher GUI. Using it makes studying the workspace of the robot easy as we can manually control individual joints through a simple slider GUI. For autonomous working of the robot, we use the Joint Position controller as its command sends the respective joints at desired position and stabilizes over there using the built-in PID controller in the effort controller.



j1	0.64
j2	-0.63
j3	1.55
j4	0.74
j5	-0.29
	5

Global Options	
Fixed Frame	dummy_link
Background Color	■ 48; 48; 48
Frame Rate	30
Default Light	✓
✓ Global Status: Ok	
✓ Fixed Frame	OK
Grid	✓
Axes	✓
✓ Status: Ok	
Reference Frame	<Fixed Frame>
Length	1
Radius	0.05
Show Trail	☐
Alpha	1
RobotModel	✓
✓ Status: Ok	
Visual Enabled	✓
Collision Enabled	☐
Update Interval	0
Alpha	1
Robot Description	robot_description
TF Prefix	
Links	
Link Tree Style	Links in Alphabetic ...
Expand Link ...	✓
All Links Ena...	✓
base	✓
base_laser	✓

Type: Orbit (rviz)

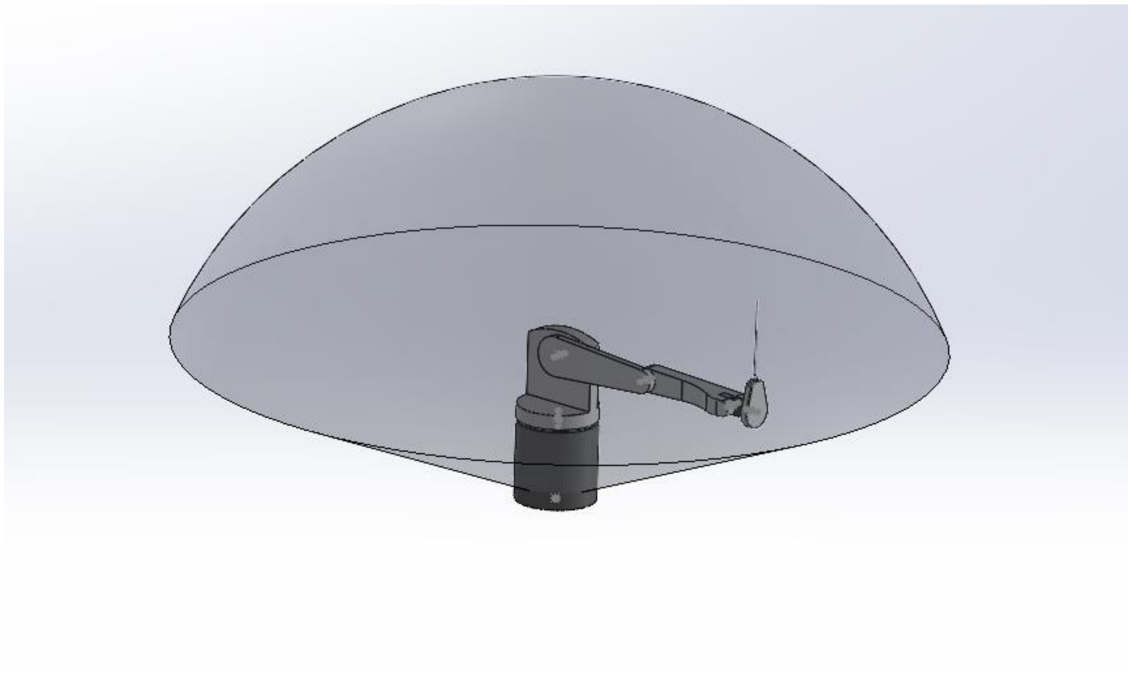
Current View	Orbit (rviz)
Near Clip ...	0.01
Invert Z Axis	☐
Target Fra...	<Fixed Frame>
Distance	3.60956
Focal Shap...	0.05
Focal Shap...	✓
Yaw	2.03351
Pitch	0.500398
Field of Vi...	0.785398
Focal Point	0.79995; 0.0249...

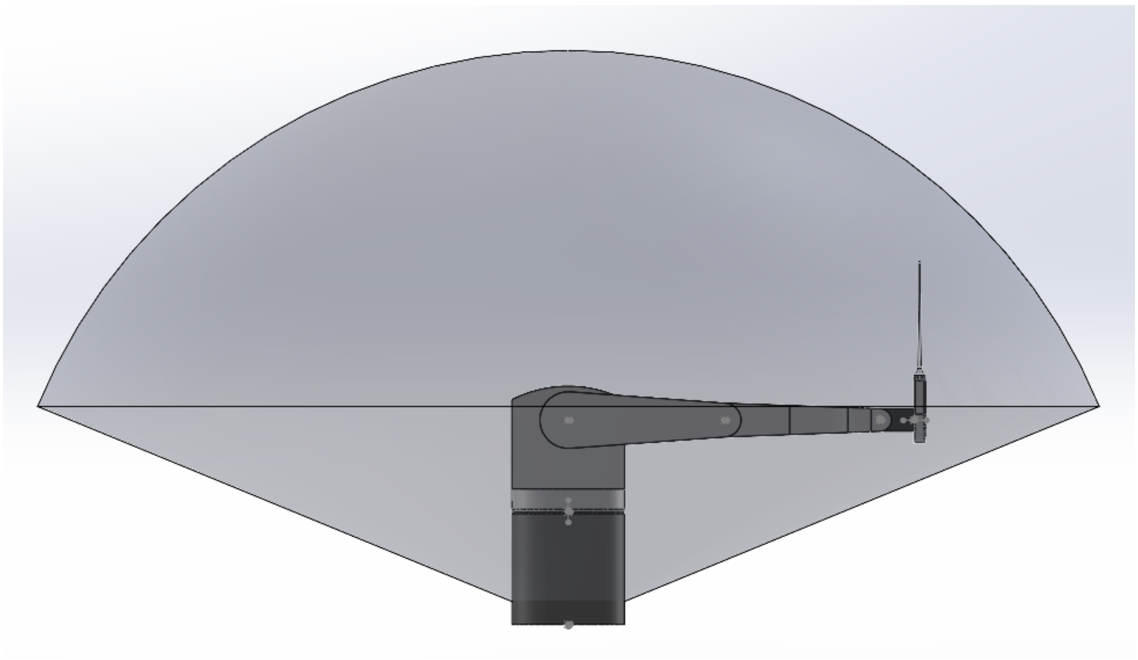
ROS Time: 82.52 ROS Elapsed: 82.24 Wall Time: 1639193852.81 Wall Elapsed: 83.33

31 fps

Workspace

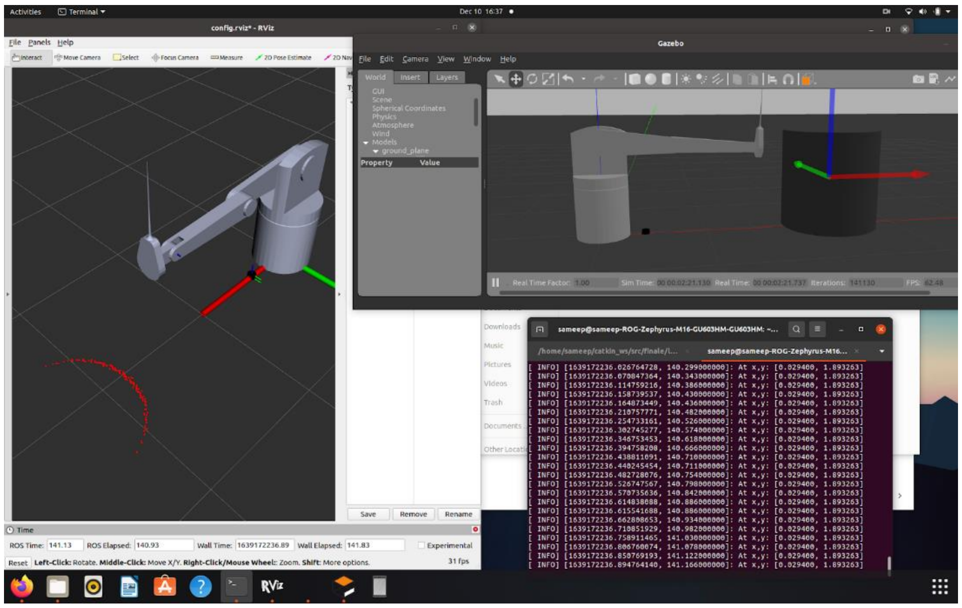
Robot working space is an important kinematic indicator. Exact computation of the boundary shape and volume or area of the manipulator workspace is very important for its optimum design and application. The Workspace of our Robot is approximated as shown in the figures below. The Workspace illustrated is an approximation and illustrated in SolidWorks





Gazebo and Rviz Visualization

The following Images represent Rviz and Gazebo Visualization using LiDAR. The red line in Rviz represents that the LiDAR has successfully detected an object in front of the arm and the node terminal on the right shows the exact location of the detected object.



```
[ INFO] [1639172236.026764728, 140.299000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.070847364, 140.343000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.114759216, 140.386000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.158739537, 140.430000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.164873449, 140.436000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.210757771, 140.482000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.254733161, 140.526000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.302745277, 140.574000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.346753453, 140.618000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.394758208, 140.666000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.438811091, 140.710000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.440245454, 140.711000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.482728076, 140.754000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.526747567, 140.798000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.570735636, 140.842000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.614838088, 140.886000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.615541688, 140.886000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.662808653, 140.934000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.710851929, 140.982000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.758911465, 141.030000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.806760074, 141.078000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.850769193, 141.122000000]: At x,y: [0.029400, 1.893263]
[ INFO] [1639172236.894764140, 141.166000000]: At x,y: [0.029400, 1.893263]
```

Problems Faced and Mitigation

The following are the problems faced by us during the execution of the given project:

1. While launching the URDF file in Gazebo world we observed that at a default inertia and friction without the joint state controller the arm used to collapse.
2. The YD LiDAR used was a 3D LiDAR hence gave multiple distance values and not the center of the object
3. Problems faced launching Joint State Publisher GUI

Lessons Learned

The following are the lessons learned by us during the execution of the given project:

1. Before launching the file, we should first set controllers and give desired friction, damping and inertia in the URDF file.
2. Learned how to use YD LiDAR to get required points
3. Joint State Publisher GUI has been separated into a new node itself hence calling it using Joint State Publisher was not working

Conclusions

In the end we would like to conclude that the CyberKnife was successfully simulated in the Gazebo World and visualized in Rviz. There were few problems faced while doing so but were resolved in the process with the help of TA’s and Professor. Through this project we were able to learn about various ROS controllers, RViz and Gazebo worlds and make our SolidWorks Skills more proficient.

References

- M. Spong, S. Hutchinson, and M. Vidyasagar, Robot Modeling and Control. New York: Wiley, 2006.
- Sun Young Noh, Kyungmin Jeong, Yong-chil Seo, Chang-hoi Kim, Jong won Park, Yoo Rark Choi, Sung Uk Lee, Yeong - Goel Bae, Seung Kim, "Development of a Prototype Robotic System for Radiosurgery with Upper Hemispherical Workspace", *Journal of Healthcare Engineering*, vol. 2017, Article ID 4264356, 9 pages, 2017. https://doi.org/10.1155/2017/4264356
- https://www.accuray.com/cyberknife/
- https://wiki.ros.org/

Drive Link for Video and PPT: Drive Link (PPT and Video): https://drive.google.com/drive/folders/1exBz_4H13SFFSS5AO3FQjCWqYOE54ETW?usp=sharing

Github Profile for Package: GitHub Link for Package: https://github.com/Atharva-Paralikar/cyber_knife_ros.git

Instructions to run Package:

1. Open teminal.
2. Type roslaunch finale empty_world.launch
3. Navigate to finale/src and open a new terminal and run cik.py
4. Follow the instructions on the terminal. Enjoy!