Simulation of 6DOF LWA4P Schunk Robot Arm

University of Maryland-College Park

Introduction:

This report examines and simulates a 6 Degrees of freedom Robotics arm made by Schunk. The arm is mounted on a base which is fixed. We worked on construction DH parameters of the said robot arm and found the forward and inverse kinematic equations with help of sympy library in python. A CAD model was created and exported to gazebo using which position and trajectory control of the end effector are visually demonstrated. This is verified by comparing results from mathematical equations and simulations.

Lwa4p Powerball:

The LWA4P is a 6 DoF (Degrees of Freedom) Robotic Arm made by Schunk Robotics under the modular Robotics platform.

As of 2023, it has it ROS packages developed for ROS1.X (latest melodic)

Figure 1: Schunk LWA4P Powerball Arm with PG70 End Effector (Najjar, 2016)

Joint Assignments:

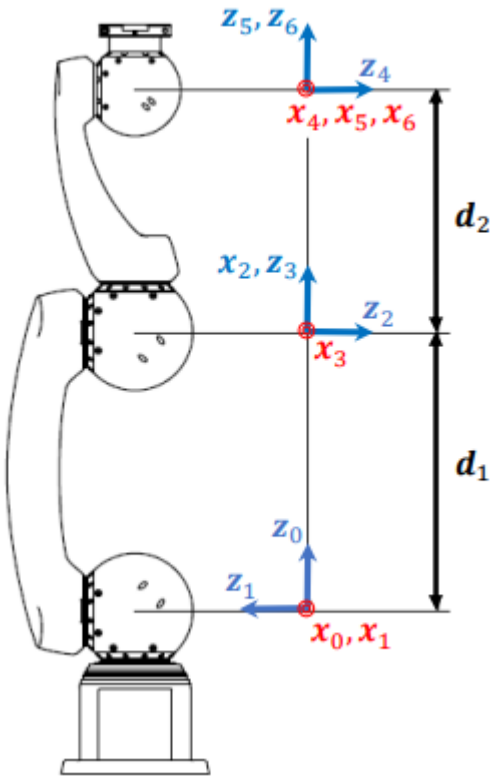


Figure 2: Joint Assignments for LWA4P (Memar and Esfahani, 2015)

Applications:

The robot can be used in the medical industry as a helping hand (with computer vision) to assist in surgeries**.** It can also be used in the manufacturing industry to perform assembly, pick and place tasks among others.

Robot Type: Manipulator

Degrees of Freedom: 6

DH Parameters:

link	d	Theta	a	alpha
Joint 1	0	q1	0	pi/2
Joint 2	0	q2+(pi/2)	0.350	pi
Joint 3	0	q3+(pi/2)	0	pi/2
Joint 4	0.350	q4	0	-pi/2
Joint 5	0	q5	0	pi/2
Joint 6	0	q6	0	0

(Bradley, 2014)

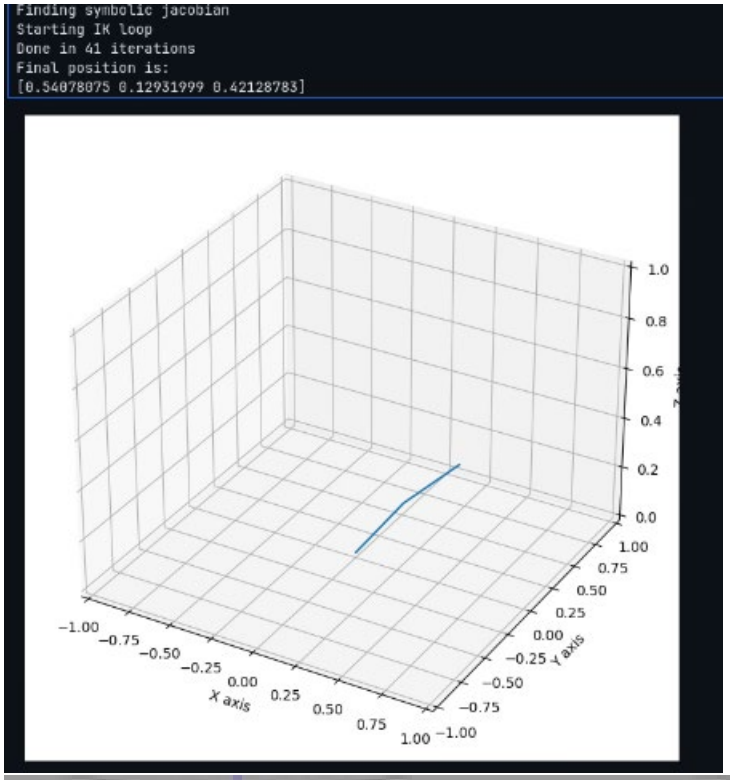
Forward Kinematics:

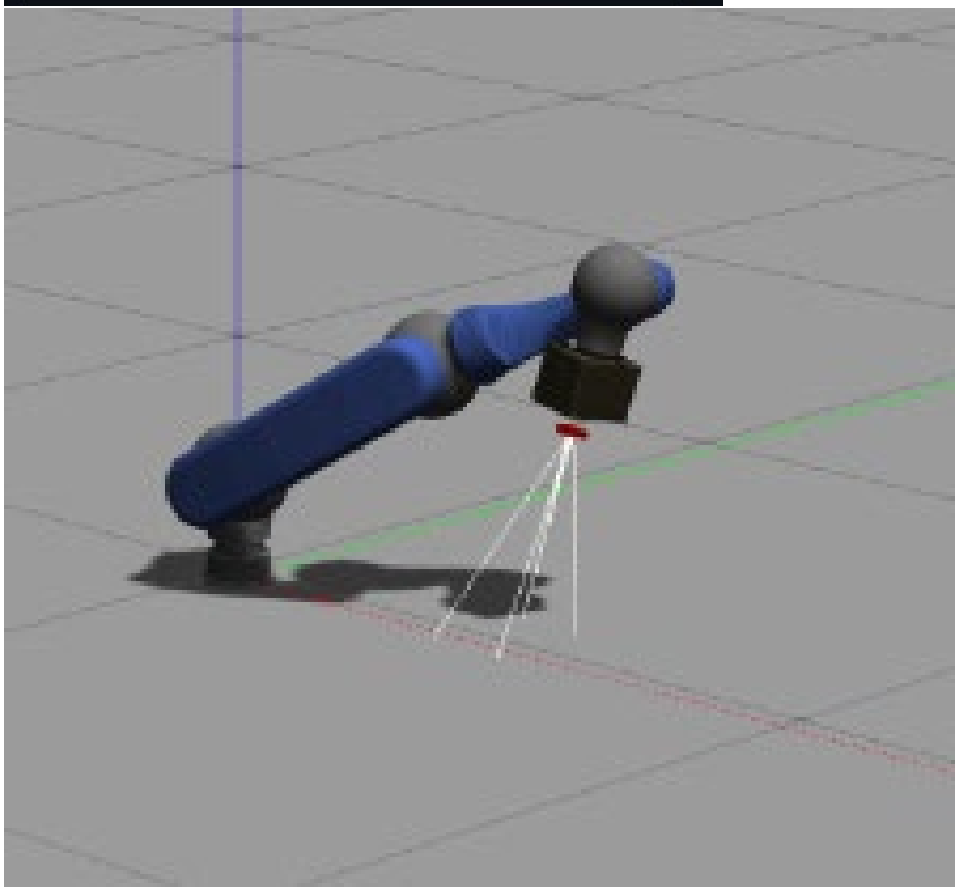
```
Final position is:
[0.55212852 0.13933223 0.40561359]

The final transformation matrix is:

 [[ 0.24719362]
  [-0.90197673]
  [ 0.09976528]
  [ 3.13973622]
  [-1.99911384]
  [ 0.25538125]]
```

Inverse Kinematics:

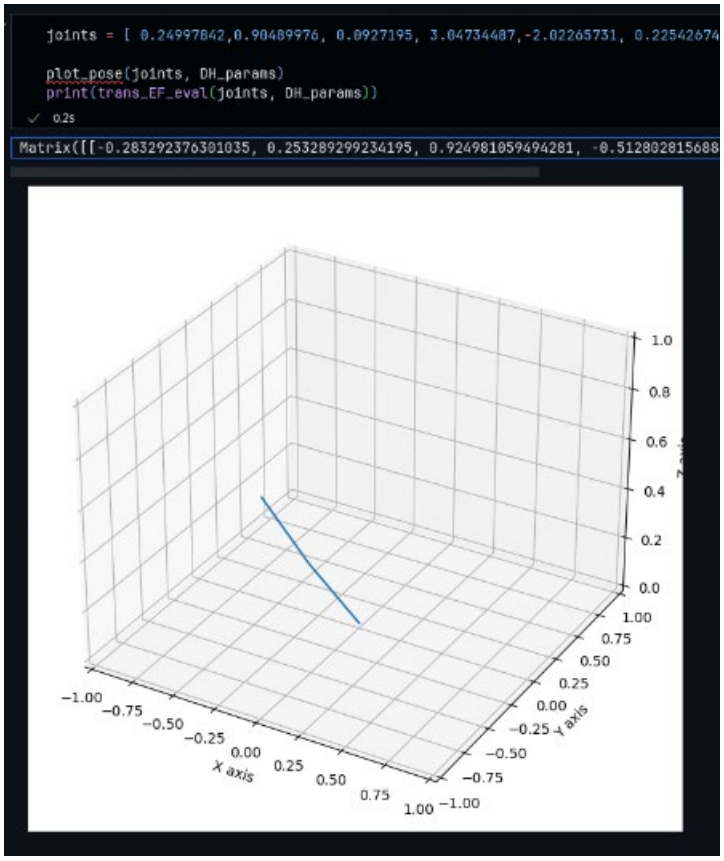
```
jonathan@jonathan-virtual-machine:~/enpm662p2_ws$ /bin/python3 /home/jonathan/enpm662p2_ws/src/

 The Jacobian matrix is:

⎡ 0.990374787299706   -0.0251716261145088   0.136103526478241   0.551596093601579⎤
⎢                                                                               ⎥
⎢-0.0181441903497499  -0.998449266452176   -0.0526293708651702  0.140832926805873⎥
⎢                                                                               ⎥
⎢ 0.137217233019814   0.0496533136846168   -0.989295698667756   0.40602397657514 ⎥
⎢                                                                               ⎥
⎣         0                   0                   0                   1        ⎦
```

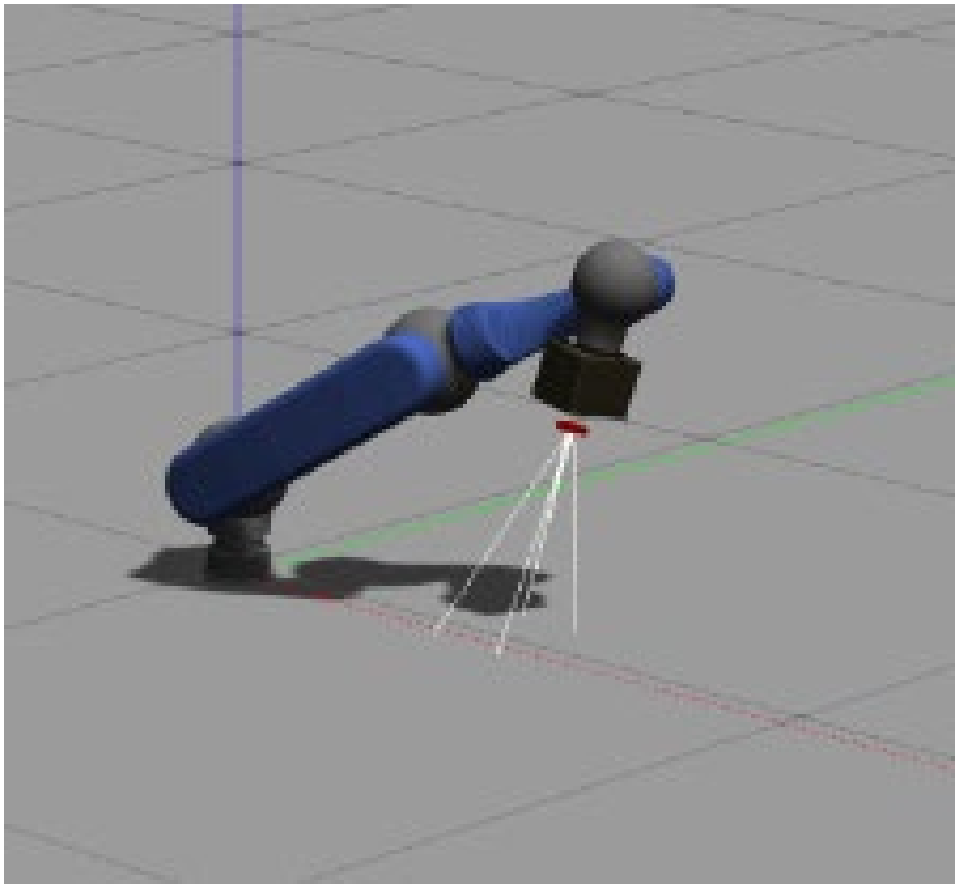
Forward Kinematics Validation:





Inverse Kinematics Validation:





Control Method:

We have used open loop control for our project.

Isaac Sim:

Isaac Sim is a new Physics simulator developed by Nvidia. It is a part of Nvidia's Omniverse Platform and is great alternative to existing ROS Simulation platforms like Gazebo and Rviz.

as of 2023, Isaac Sim is capable of Assembling Robot Models, Making Worlds, Importing URDFs, and adding realistic textures which comply with technologies like Real Time Ray Tracing, DLSS with Physics.

The Native ROS Publishers and Subscribers can be created by means of either an Action Graph or by Scripting it in Python.

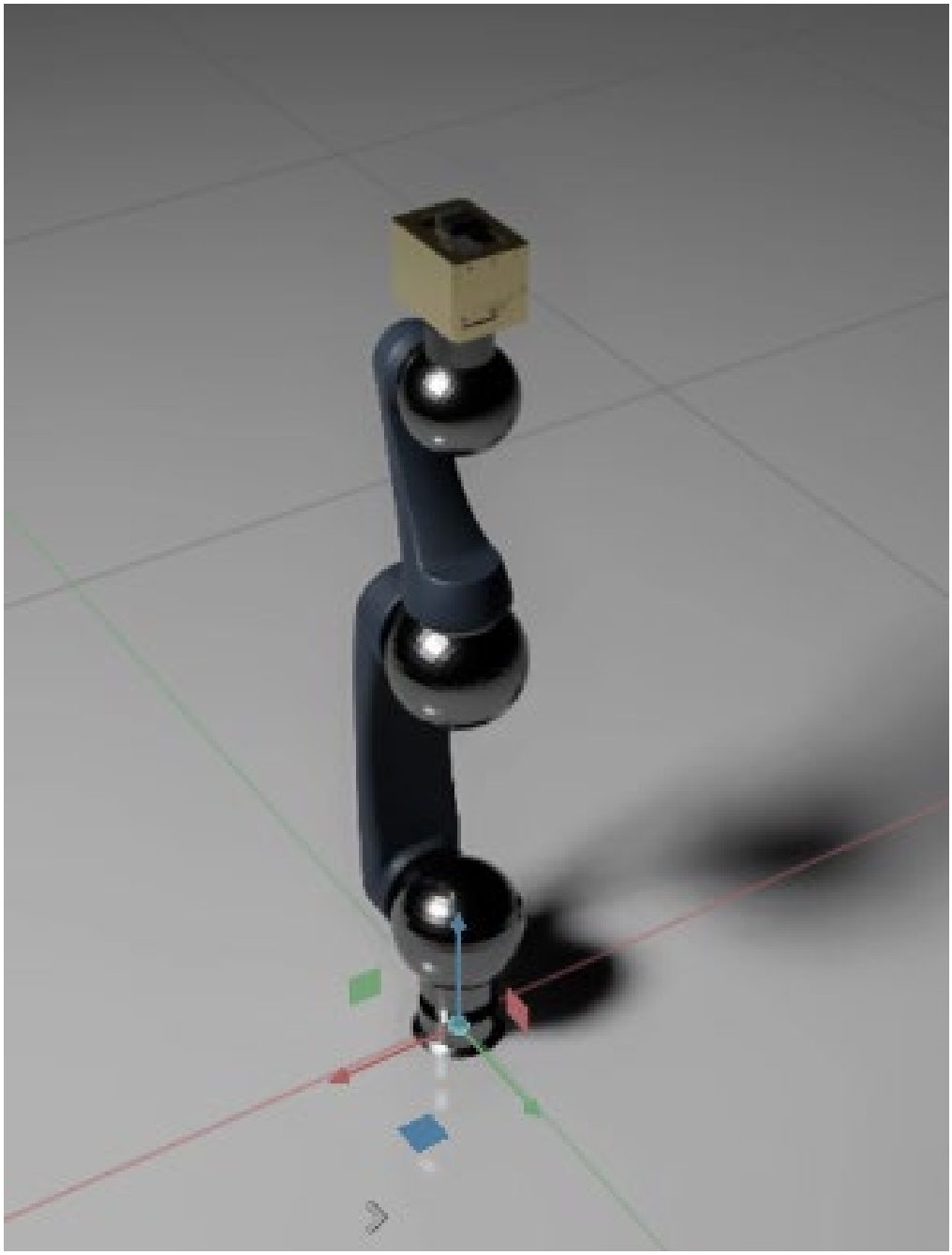


Figure 3: LWA4P + PG70 in Isaac Sim

Publishers and Subscribers in Isaac Sim:

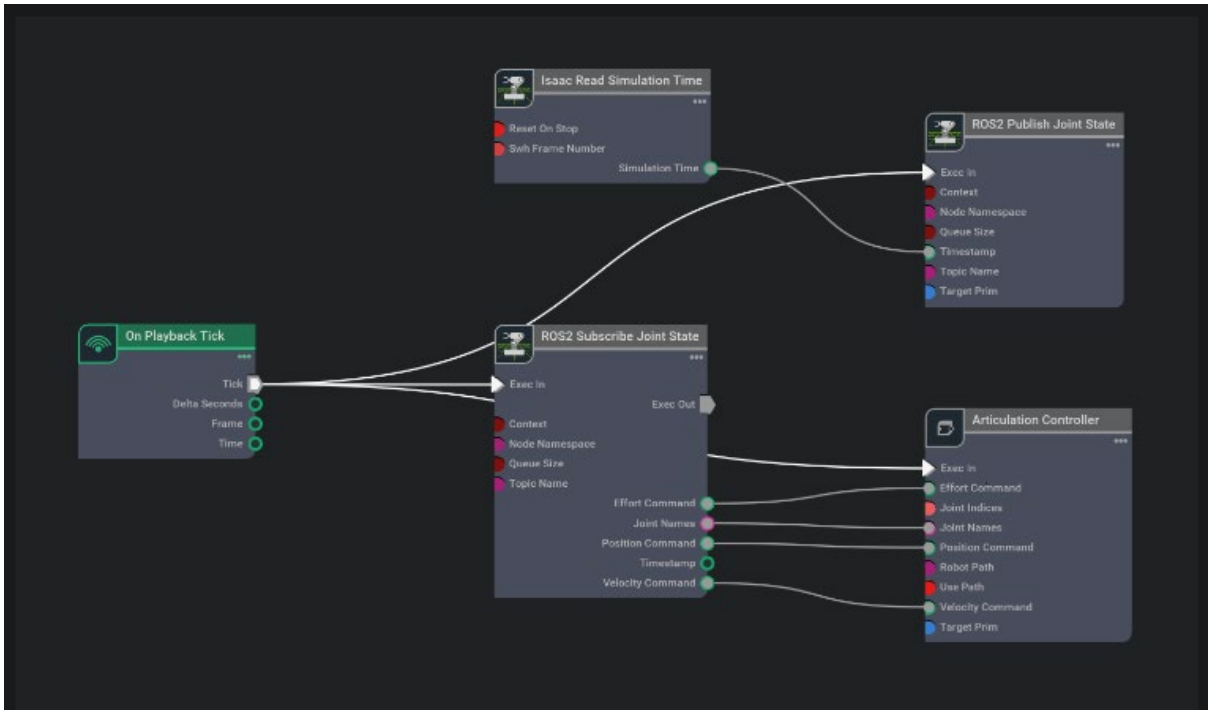


Figure 4: Action Node Implementation

Node Structure:

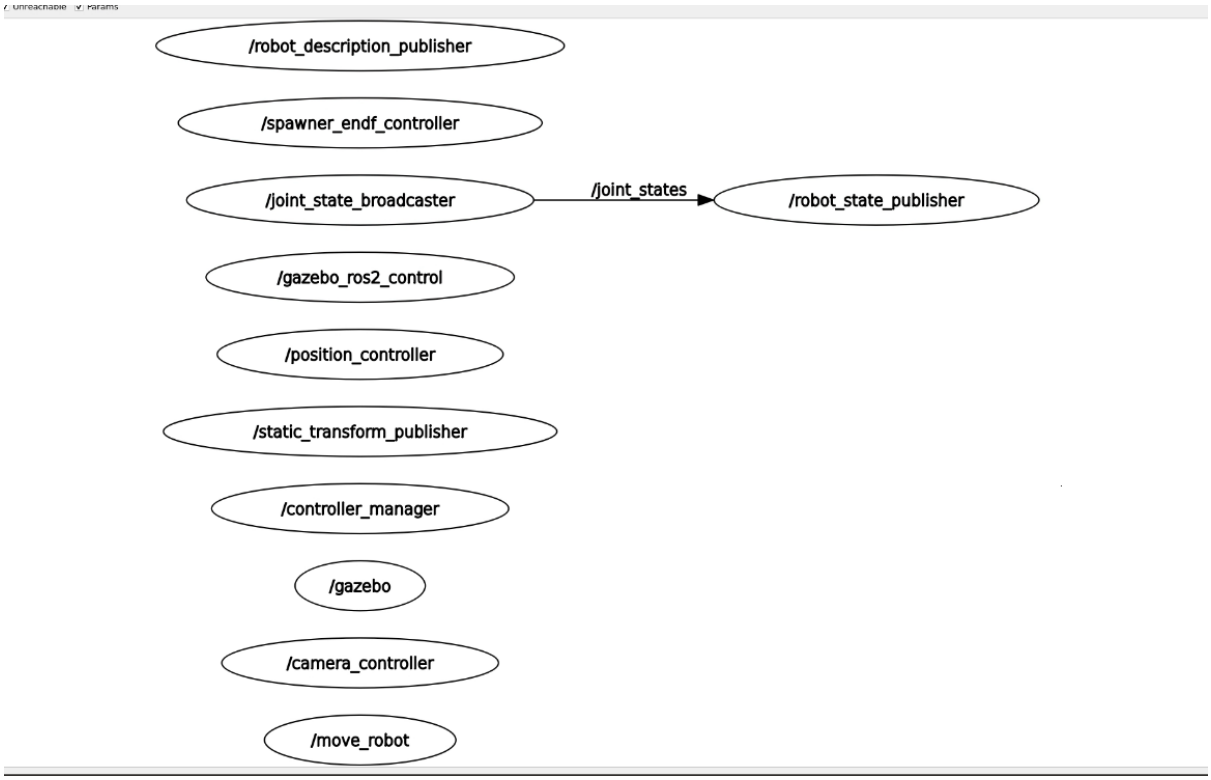


Figure 5: RQT_Graph Output

Problems Faced:

1. Iterating the URDF through SolidWorks or Fusion 360 often requires the need to reboot to Windows 11 from Ubuntu 22.04 repeatedly.

2. The Draw Circle node, although publishing correctly, does not complete the current circle (radius = 10) due limits in the joint angles. Although it might work with lesser values (not tested)

Lessons Learned:

1. Package Versioning to make ROS nodes to work across versions of ROS2.X
2. Recreating a Robot in Fusion 360
3. Making URDF from scratch
4. Simulated a Robot in Gazebo, Rviz and Isaac Sim

Future Scope:

1. As the IK Node takes too long to run, efforts have been made to push the IK kinematics to the GPU Pipeline using CuPy instead of python and will be added in the future iterations of the repository. With sufficient bandwidth on the PCIe lanes, should be able to integrate OpenCV for near Real Time Inverse Kinematics for cases like Hand Gesture tracking.
2. The Publishing and subscribing nodes in Isaac Sim are currently set up using Action Graphs, the script-based implementation will be done in future iterations of the repository.

Github Links:

Current Project :

https://github.com/1412kauti/lwa4p_pg70/tree/humble

Noetic Node with OpenCV + Isaac Sim:
https://github.com/1412kauti/lwa4p_pg70_moveit_isaac/tree/noetic

Reference list

Bradley, C. (2014). *Robotic Arm Calibration and Control 6-DOF Powerball LWA 4P*. [online] Available at: https://foswiki.cs.rpi.edu/foswiki/pub/RoboticsWeb/LabPublications/bradley_ROBOTIC_ARM_CALIBRATION__CTRL_report.pdf [Accessed 10 Dec. 2023].

Memar, A.H. and Esfahani, E.T. (2015). Modeling and Dynamic Parameter Identification of the SCHUNK Powerball Robotic Arm. *Volume 5C: 39th Mechanisms and Robotics Conference*. doi:https://doi.org/10.1115/detc2015-47703.

Najjar, A. (2016). *Lwa4p Robot Arm Setup*. [online] ammarnajjar.github.io. Available at: https://ammarnajjar.github.io/development/2016/06/22/lwa4p-robot/ [Accessed 10 Dec. 2023].