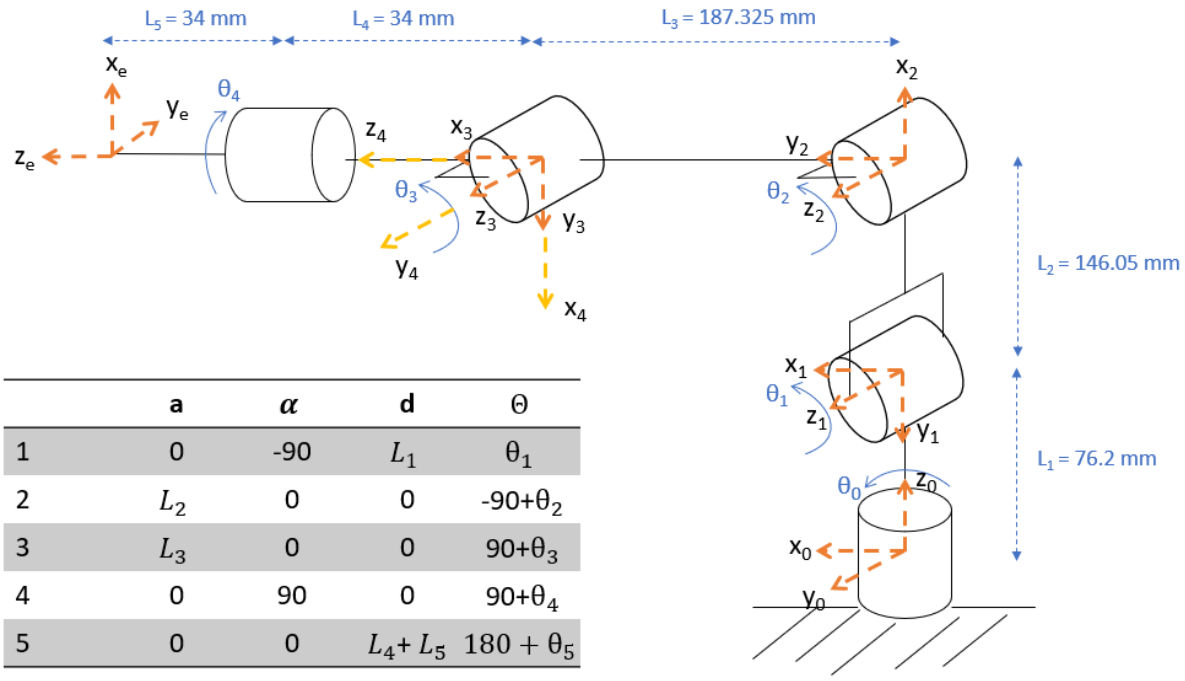
[redacted]

MEAM 520-Lab1

Methods

Robot Arm Configuration:



	a	α	d	Θ
1	0	-90	L_1	θ_1
2	L_2	0	0	-90+θ_2
3	L_3	0	0	90+θ_3
4	0	90	0	90+θ_4
5	0	0	L_4+ L_5	$180 + \theta_5$

Homogenous Transformation matrices:

$$H_n^{n-1} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$H_1^0 = \begin{bmatrix} cos\theta_1 & 0 & -sin\theta_1 & 0 \\ sin\theta_1 & 0 & cos\theta_1 & 0 \\ 0 & -1 & 0 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; H_2^1 = \begin{bmatrix} sin\theta_2 & cos\theta_2 & 0 & L_2 sin\theta_2 \\ -cos\theta_2 & sin\theta_2 & 0 & -L_2 cos\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$H_3^2 = \begin{bmatrix} -sin\theta_3 & -cos\theta_3 & 0 & -L_3 sin\theta_3 \\ cos\theta_3 & -sin\theta_3 & 0 & L_3 cos\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, H_4^3 = \begin{bmatrix} -sin\theta_4 & 0 & cos\theta_4 & 0 \\ cos\theta_4 & 0 & sin\theta_4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$H_e^4 = \begin{bmatrix} -cos\theta_5 & sin\theta_5 & 0 & 0 \\ -sin\theta_5 & -cos\theta_5 & 0 & 0 \\ 0 & 0 & 1 & L_4 + L_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Position of center of gripper:

$$T_e^0 = H_1^0 H_2^1 H_3^2 H_4^3 H_e^4$$

$$T_e^0[0:3, -1]$$

The configuration of the robot arm is R(5)P but the prismatic joint at the end effector does not impact the position of the center of the joint, so it is not included in the robot configuration drawing. The frames of each joint were drawn using Denavit–Hartenberg (DH) constraints to assign directions of the axes. Then the transformation matrix between each link was found using the DH equation in blue. The final transformation matrix of the end effector is found by post multiplying all the successive link transformation matrices. The position of the end effector is the first three rows of the last column in T_e^0.

When drawing the configuration, we assumed that the joints in their zero configurations were perfectly aligned in the L shape as drawn above. There is most likely some offset in the physical robot between the joints in the zero configuration that wouldn't make them perfectly aligned. A limitation of using the DH method was that the joint 4 frames could not be located on the actual center of the joint because of the DH constraints. Having the frame not be attached to the joint position did not affect the end effector position calculation but required an intermediate transformation to translate the transformation matrix for T_4^0 by L_4 in the z direction to get the position of frame 4 back onto joint 4. The intermediate transformation to get the position of joint 4 was done with the expression below.

$$T_{4'}^0 = T_4^0 T_{4'}^4 \text{ where } T_{4'}^4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Then the position of joint 4 was the first three rows of the last column.

Code description

This lab has two files, calculateFK.py and computeWorkspace.py.

The first one asks for the six joints' input (distance(mm) or degrees (Rad)) of the gazebo robot arm. It returns all the x,y,z coordinates of each joint's center in mm, and a 4 by 4 homogeneous

transformation matrix representing the end effector frame expressed in the base frame. The function does the following:

1. Generate all the homogeneous transformation matrices for each adjacent joint using input data.
2. Multiply each matrix consecutively starting from the joint 1 expressed in base frame.
3. While multiplying, store the first three numbers of the last column in the matrix from each multiplication. These are the coordinates at each joint respectively.
4. The final result of the multiplication is the desired homogeneous transformation matrix (T_e^0).

The second code generates the reachable workspace of the gazebo robot arm.

1. Generate 10 angles uniformly within the limitation for each joint
2. Uses 5 nested for loops to compute all the gripper locations for each point set combination using calculateFK. Appends the 10^5 points generated in the vector named pos
3. Plot 3D graphs for all the gripper locations, which would represent the reachable workspace of this robot arm.

Evaluation

Homogeneous transformation for the zero pose:

$$T_e^0 = \begin{bmatrix} 0 & 0 & 1 & 255.325 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 222.25 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Inputting 0's for all joint parameters into calculateFK resulted in the transformation matrix above. This matrix matched the expected transformation matrix from the pre-lab.

Given Joint angles (1) $\theta_1 = \pi/4, \theta_2 = \theta_3 = \theta_4 = \theta_5 = 0$ and (2) $\theta_1 = -\pi/2, \theta_2 = 0, \theta_3 = \pi/4, \theta_4 = 0, \theta_5 = \pi/2$:

(1)

$$T_e^0 = \begin{bmatrix} 0 & 0.707 & 0.707 & 180.54 \\ 0 & -0.707 & 0.707 & 180.54 \\ 1 & 0 & 0 & 222.25 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(2)

$$T_e^0 = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 0.707 & -0.707 & -180.54 \\ 0 & -0.707 & -0.707 & 41.71 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Inputting the above joint angles to calculateFK for all joint parameters resulted in the transformation matrices above. Both results matched the expected transformation matrices from the pre-lab. This was good confirmation that the code developed was working properly, but we tested the functionality of the code with more test cases to understand the limitations of the code when compared to the robot.

Other Test Cases:

Test case 1- Making the robot point vertically:

$$\theta_1 = 0, \theta_2 = 0, \theta_3 = -\pi/2\ , \theta_4 = \ 0, \theta_5 = 0$$

This test configuration was chosen because if the robot is vertical it is easy to know by observation that the end effector should be a distance L1+L2+L3+L4+L5 in the z direction and 0 in the x and y direction. Similarly, the rotation of the axis is simple to determine by inspection because the x and y axis of the end effector are in the opposite direction and the z axis is the same direction. It is also easy to tell if the robot in the simulation is in a straight vertical orientation.

Test case 1- Output:

$$T_e^0 = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 477.525 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

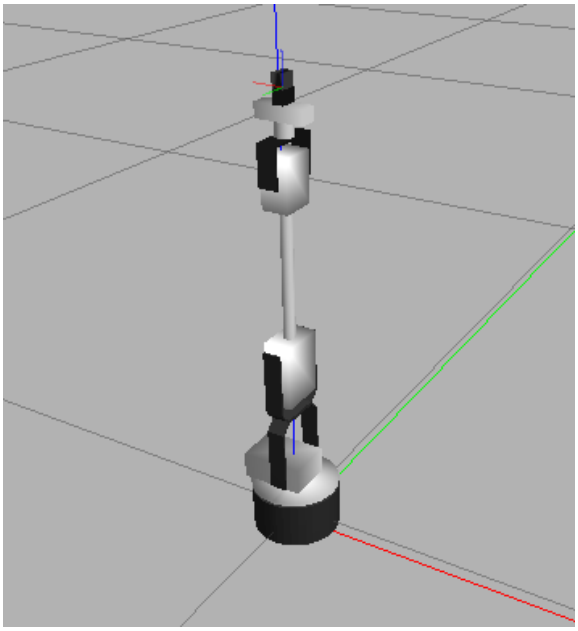


Figure 1: Arm in vertical position

The output of the code was the expected matrix and the robot moved to the expected configuration, but the tip of the robot was swaying back and forth in the x direction slightly. This movement was not expected based on our calculations.

Test case 2- Making the robot point in the opposite direction:

$$\theta_1 = 0, \theta_2 = 0, \theta_3 = -\pi, \theta_4 = 0, \theta_5 = 0$$

This test configuration was chosen because if the robot is pointing in the opposite direction it is easy to know by observation that the end effector should be a distance L3+L4+L5 in the -x direction and L2 +L3 in the z and 0 in the y direction. Similarly, the rotation of the axis is simple to determine by inspection because the x axis of the end effector is in the -z direction, the y axis of the end effector is in the -y direction, and the z axis of the end effector is in the -x direction. It is also easy to tell if the robot in the simulation is pointing in the opposite orientation.

Test case 2- Output:

$$T_e^0 = \begin{bmatrix} 0 & 0 & -1 & -255.325 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 1 & 222.25 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

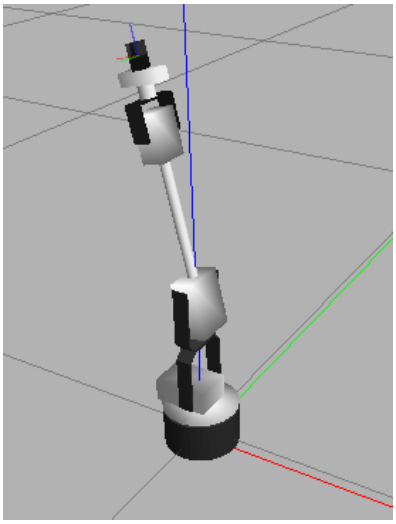


Figure 2: Arm unable to achieve opposite position

The output of the code was the expected matrix, but the robot did not move to the expected configuration. This is because the robot has joint limits that restrict its range of motion. We found that the limits for each revolute joint are -1.4 to 1.4 , -1.2 to 1.4, -1.8 to 1.7, -1.9 to 1.7, -2.0 to 1.5 for $\theta_1, \theta_2, \theta_3$, θ_4 , θ_5 respectively. The gripper does not change the position of the end effector but was shown to have a range of -15 to 30 mm.

Test case 3- Making the robot's tip point straight down:

$$\theta_1 = 0, \theta_2 = 0, \theta_3 = 0\,, \theta_4 = \pi/2\,, \theta_5 = 0$$

This test configuration was chosen because if the robot is pointing straight down it is easy to know by observation that the end effector should be a distance L3 in the x direction and L1 + L2 – L4 – L5 in the z direction and 0 in the y direction. Similarly, the rotation of the axis is simple to determine by inspection because the x axis of the end effector is in the x direction, the y axis of the end effector is in the -y direction, and the z axis of the end effector is in the -z direction. It is also easy to tell if the robot in the simulation is pointing in the downward orientation.

Test case 3- Output:

$$T_e^0 = \begin{bmatrix} 1 & 0 & 0 & 187.325 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 154.25 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

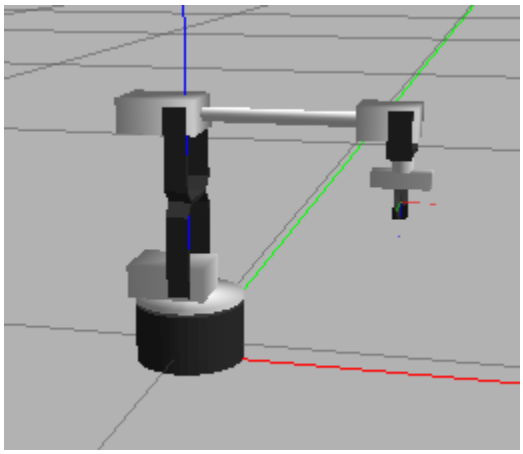


Figure 3: Arm in downward pointing position

The output of the code was the expected matrix and the robot moved to the expected configuration.

Reachable Workspace strategy:

Workspace of a robot arm is the entire set of points reachable by the manipulator, which is the center of the gripper in this case. The strategy was to compute all the joint angle combinations within their limitations with 5 nested for loops and store each position coordinate of the end effector. The 3D plot of the stored coordinates would represent the reachable workspace for the robot simulation. First, 10 uniform angles were generated for each joint within its respective range, then 'calculateFK' function was used to compute all the end effector positions for every combination of joint angles. Once the data was collected, mathplotlib toolkit was used to plot the points and approximate the entire reachable workspace.

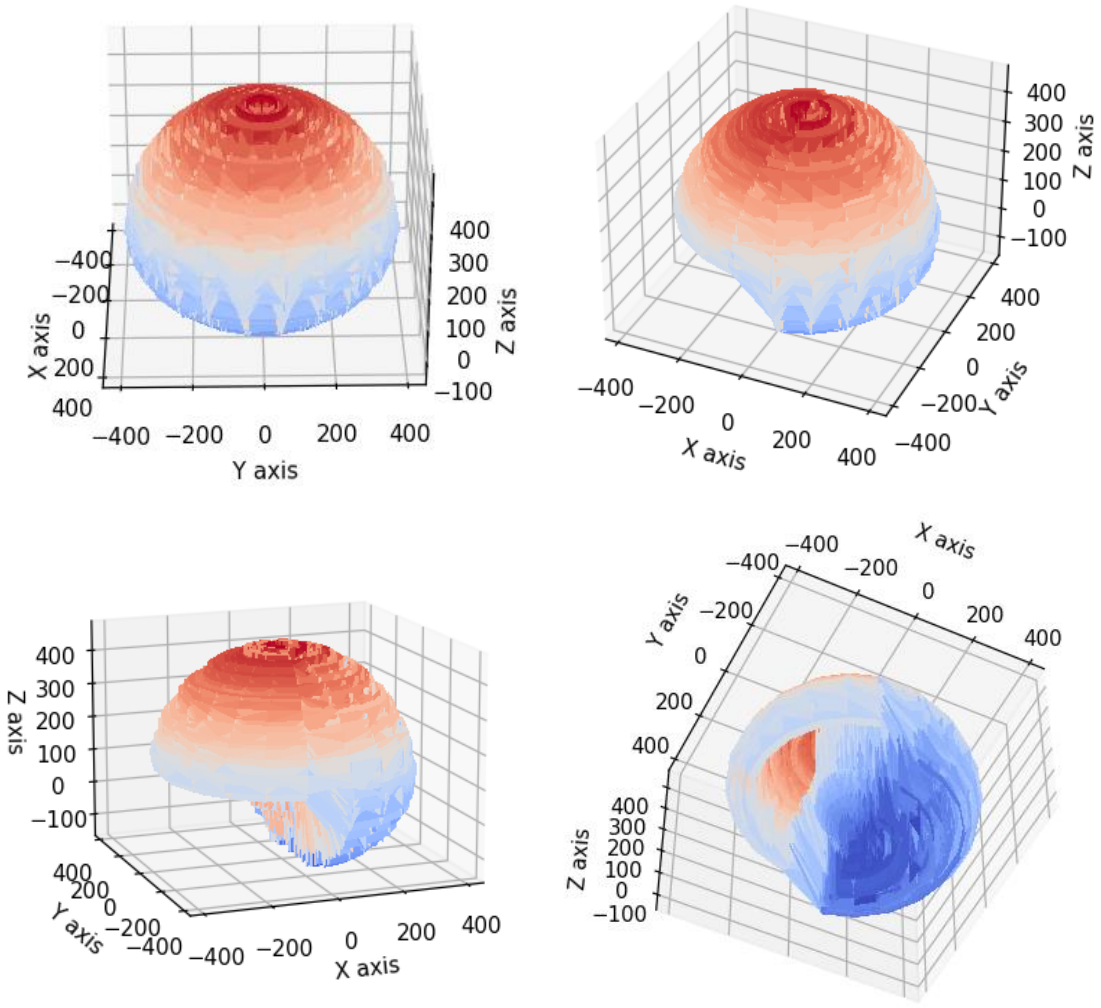


Figure 4: 3D workspace of robot simulation

Results compared to TestFK_Sim output:

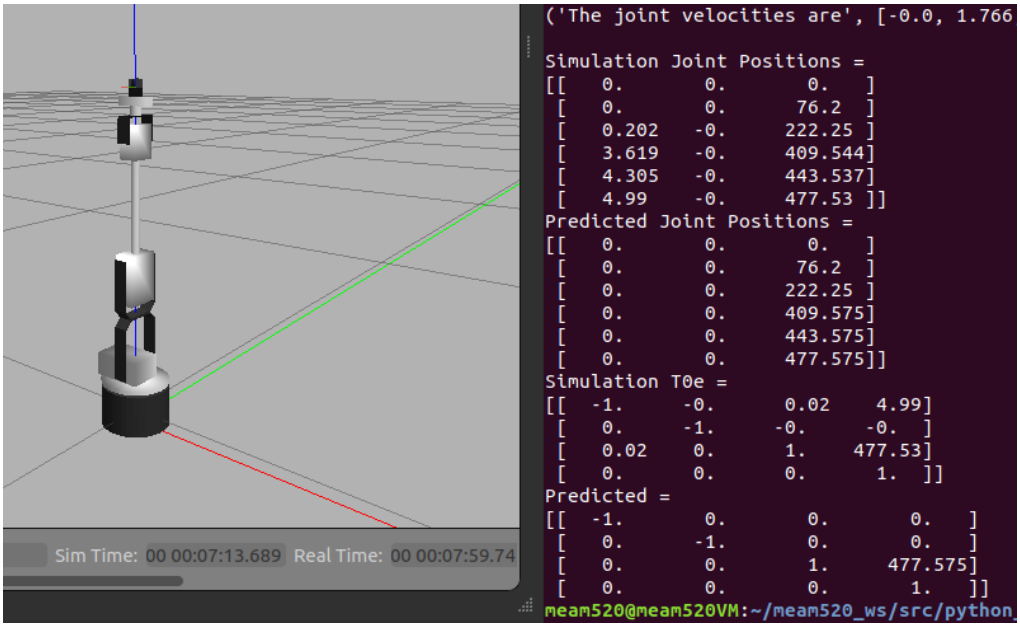


Figure 5: TestFK_sim output for test case 1

The TestFK_sim output for test case 1 revealed that the position of the robot was not actually perfectly vertical. This is consistant with the unstable swaying that was oberved when looking at the robot in our initial testing.

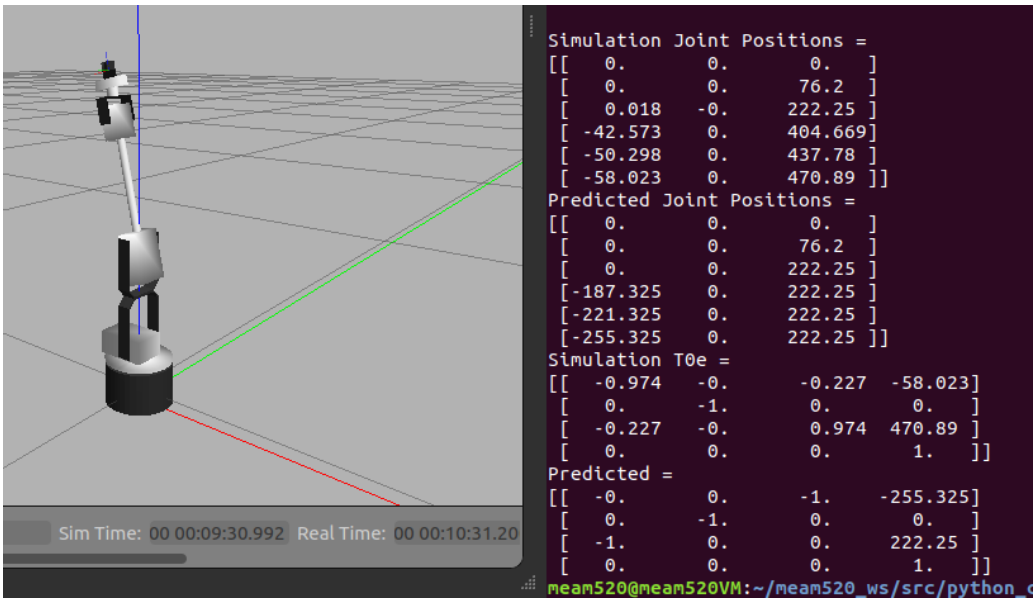


Figure 6: TestFK_sim output for test case 2

The TestFK_sim output for test case 2 revealed that the coordinates of the joints were not at all in the desired positions. This was expected because the robot was not able to reach the configeration in the previous testing because of the joint limits.

Figure 7: TestFK_sim output for test case 3

The TestFK_sim output for test case 3 revealed that the position of the robot was in the exact positon that was desired. This is consistant with the testing which showed that the robot was in the expected position.

2.5 Analysis

1. The computeFK function output for the first test case was the expected transformation matrix for the configuration, and the robot simulation looked like it was in the correct configuration. However, we noticed that the tip of the robot was shaking back and forth in the simulation which was unexpected. The swaying of the arm resulted in a position that was slightly off from the expected position of the robot end effector. This test case revealed that the robot simulation considers acceleration and velocity of the linkages while the computeFK function does not.

The computeFK function output for the second test case was the expected transformation matrix for the configuration, but the robot simulation was unable to reach this configuration. because it was out of joint 3's range of motion. The computeFK code outputs the final matrix and position even if the input is outside the arms joint limit. The decision to allow joint inputs that are outside of the joint's limit was based on the given pre-lab inputs which were not all

inside the joint limits of the simulated robot. This test case brought the joint limits of the simulated robot into closer attention when considering the motion of the robot.

The computeFK function output for the third test case was the expected transformation matrix for the configuration, and the robot simulation looked like it was in the correct configuration. The output position from the simulation was the same as the expected position. This test case further validated our model for the robot positions.

The workspace generated by the computeWorkspace function was able to better depict the workspace of the robot than the sketches initially done in the prelab. The 3D plot is similar to what was expected based on our sketches, but the shape of workspace is better visualized. It is clearer that the workspace has a large gap in the negative xyz region.

2. There is a point ($\theta_1 = -\pi/2, \theta_2 = 0, \theta_3 = \pi/4, \theta_4 = 0, \theta_5 = \pi/2$) in pre-lab that the team did not expect the simulated robot to reach, but it did. The limit of joint 1 was between [-1.4,1,4] (Radians) and the input $\theta_1$is out of its range. Taking this point and the test case 1 which bounced back and forth into consideration, the team guesses it might be because of the joint acceleration. When the robot arm reached test case 1, the acceleration became 0, but it still had velocity to keep it moving forward. This phenomenon was enlarged by the length of the arm as it was rotating around joint 2. The longer the arm, the bigger velocity it would be, which made it visually obvious in the simulation. And the team hypothesizes that it is the acceleration that helped the simulated to reach the pre-lab position as the difference between them were not significant (0.17 Radians). In order to predict the difference, the team wanted to acknowledge the acceleration of each joint to better predict this difference.

There are also a lot of points that were not in the plot of our reachable workspace that were reachable in the simulation. This was due to the fact that there are an infinite number of points in the workspace, but the team only calculated 100,000 end effector positions. Any point that isn't in the calculated set is not shown in the graph. The team tried to represent the workspace more closely by generating more angles with less interval between them within the joints' limitations, but the process time was exponentially longer. If more the reachable gripper positions were provided, the team could plot a smoother plot without dealing with interminable process times.