Introduction to Robotics
Lab 4 - Forward Kinematics

Names: Hamad Abdul Razzaq & Muhammad Ali Siddiqui
IDs: hr06899 & ms06875

Lab Report

Task 4.1

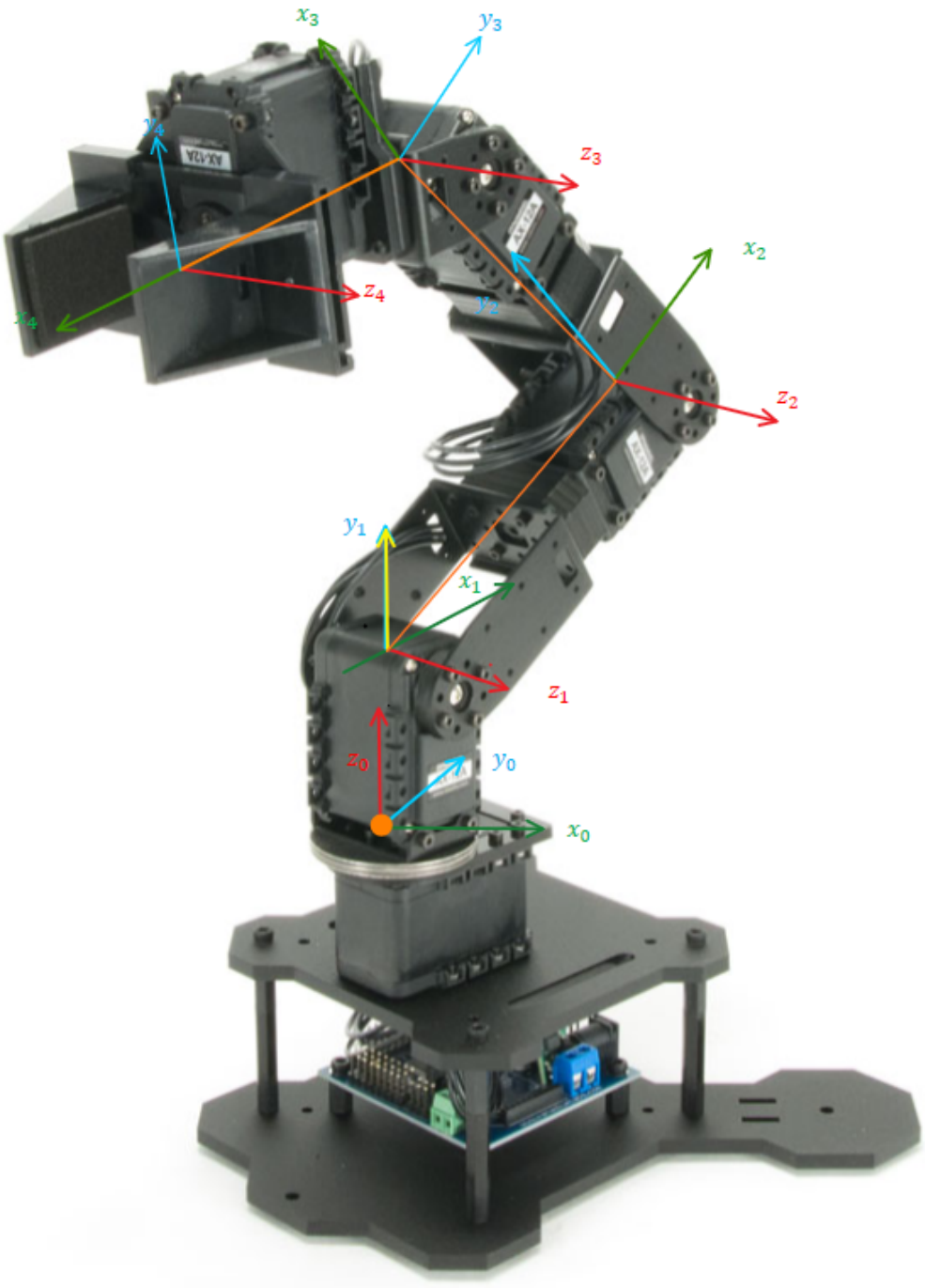

Figure 1: Frame Assignment Based on DH convention

Task 4.2

Link No.	a_i (mm)	α_i (radians)	d_i (mm)	θ_i (radians)
1	0	$\frac{\pi}{2}$	54	θ_1
2	108	0	0	θ_2
3	108	0	0	θ_3
4	76	0	0	θ_4

Table 1: Table of DH parameters

Task 4.3

Part (a)

The following code generates the intermediate homogenous transformations ${}^0T_1$, ${}^1T_2$, ${}^2T_3$ and ${}^3T_4$.

```
syms('theta_1', 'theta_2', 'theta_3', 'theta_4', 'alpha_1', 'alpha_2', 'alpha_3', ...
    'alpha_4', 'a_1', 'a_2', 'a_3', 'a_4', 'd_1', 'd_2', 'd_3', 'd_4');
syms('T_01', 'T_12', 'T_23', 'T_34', 'T_04')
T_01 = [cos(theta_1), -sin(theta_1)*cos(alpha_1), sin(theta_1)*sin(alpha_1), a_1 * ...
    cos(theta_1);
         sin(theta_1), cos(theta_1)*cos(alpha_1), -cos(theta_1)*sin(alpha_1), a_1 * ...
             sin(theta_1);
         0, sin(alpha_1), cos(alpha_1), d_1;
         0 0 0 1]
T_12 = [cos(theta_2), -sin(theta_2)*cos(alpha_2), sin(theta_2)*sin(alpha_2), a_2 * ...
    cos(theta_2);
         sin(theta_2), cos(theta_2)*cos(alpha_2), -cos(theta_2)*sin(alpha_2), a_2 * ...
             sin(theta_2);
         0, sin(alpha_2), cos(alpha_2), d_2;
         0 0 0 1]

T_23 = [cos(theta_3), -sin(theta_3)*cos(alpha_3), sin(theta_3)*sin(alpha_3), a_3 * ...
    cos(theta_3);
         sin(theta_3), cos(theta_3)*cos(alpha_3), -cos(theta_3)*sin(alpha_3), a_3 * ...
             sin(theta_3);
         0, sin(alpha_3), cos(alpha_3), d_3;
         0 0 0 1]
T_34 = [cos(theta_4), -sin(theta_4)*cos(alpha_4), sin(theta_4)*sin(alpha_4), a_4 * ...
    cos(theta_4);
         sin(theta_4), cos(theta_4)*cos(alpha_4), -cos(theta_4)*sin(alpha_4), a_4 * ...
             sin(theta_4);
         0, sin(alpha_4), cos(alpha_4), d_4;
         0 0 0 1]
```

The above code gave the following output:

$$
{}^{0}T_{1} = \begin{pmatrix} \cos(\theta_1) & -\cos(\alpha_1)\,\sin(\theta_1) & \sin(\alpha_1)\,\sin(\theta_1) & a_1\,\cos(\theta_1) \\ \sin(\theta_1) & \cos(\alpha_1)\,\cos(\theta_1) & -\sin(\alpha_1)\,\cos(\theta_1) & a_1\,\sin(\theta_1) \\ 0 & \sin(\alpha_1) & \cos(\alpha_1) & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{1}T_{2} = \begin{pmatrix} \cos(\theta_2) & -\cos(\alpha_2)\,\sin(\theta_2) & \sin(\alpha_2)\,\sin(\theta_2) & a_2\,\cos(\theta_2) \\ \sin(\theta_2) & \cos(\alpha_2)\,\cos(\theta_2) & -\sin(\alpha_2)\,\cos(\theta_2) & a_2\,\sin(\theta_2) \\ 0 & \sin(\alpha_2) & \cos(\alpha_2) & d_2 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{2}T_{3} = \begin{pmatrix} \cos(\theta_1) & -\cos(\alpha_1)\,\sin(\theta_1) & \sin(\alpha_1)\,\sin(\theta_1) & a_1\,\cos(\theta_1) \\ \sin(\theta_1) & \cos(\alpha_1)\,\cos(\theta_1) & -\sin(\alpha_1)\,\cos(\theta_1) & a_1\,\sin(\theta_1) \\ 0 & \sin(\alpha_1) & \cos(\alpha_1) & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{3}T_{4} = \begin{pmatrix} \cos(\theta_4) & -\cos(\alpha_4)\,\sin(\theta_4) & \sin(\alpha_4)\,\sin(\theta_4) & a_4\,\cos(\theta_4) \\ \sin(\theta_4) & \cos(\alpha_4)\,\cos(\theta_4) & -\sin(\alpha_4)\,\cos(\theta_4) & a_4\,\sin(\theta_4) \\ 0 & \sin(\alpha_4) & \cos(\alpha_4) & d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

Part (b)

Finally, we can find the resultant matrix by running the following line of code:

```
T_04 = simplify(T_01*T_12*T_23*T_34)
```

This gives the required transformation ${}^0T_4$ which is:

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

$$A_{11} = -\sin\theta_4\,\sigma_1 - \cos\theta_4\,\sigma_5$$

$$A_{12} = \sin\alpha_4\,\sigma_3 + \cos\alpha_4\,\sin\theta_4\,\sigma_5 - \cos\alpha_4\,\cos\theta_4\,\sigma_1$$

$$A_{13} = \cos\alpha_4\,\sigma_3 - \sin\alpha_4\,\sin\theta_4\,\sigma_5 + \sin\alpha_4\,\cos\theta_4\,\sigma_1$$

$$A_{14} = a_1\,\cos\theta_1 + d_4\,\sigma_3 + d_3\,\sigma_{10} + a_2\,\cos\theta_1\,\cos\theta_2 - a_4\,\sin\theta_4\,\sigma_1 + d_2\,\sin\alpha_1\,\sin\theta_1 - a_3\,\sin\theta_3\,\sigma_{12} - a_4\,\cos\theta_4\,\sigma_5 + a_3\,\cos\theta_3\,\sigma_{13} - a_2\,\cos\alpha_1\,\sin\theta_1\,\sin\theta_2$$

$$A_{21} = -\cos\theta_4\,\sigma_6 - \sin\theta_4\,\sigma_2$$

$$A_{22} = \sin\alpha_4\,\sigma_4 + \cos\alpha_4\,\sin\theta_4\,\sigma_6 - \cos\alpha_4\,\cos\theta_4\,\sigma_2$$

$$A_{23} = \cos\alpha_4\,\sigma_4 - \sin\alpha_4\,\sin\theta_4\,\sigma_6 + \sin\alpha_4\,\cos\theta_4\,\sigma_2$$

$$A_{24} = d_4\,\sigma_4 - d_3\,\sigma_{11} + a_1\,\sin\theta_1 - a_4\,\sin\theta_4\,\sigma_2 - d_2\,\sin\alpha_1\,\cos\theta_1 - a_3\,\sin\theta_3\,\sigma_{14} + a_2\,\cos\theta_2\,\sin\theta_1 - a_4\,\cos\theta_4\,\sigma_6 + a_3\,\cos\theta_3\,\sigma_{15} + a_2\,\cos\alpha_1\,\cos\theta_1\,\sin\theta_2$$

$$A_{31} = \sin\theta_4\,\sigma_7 + \cos\theta_4\,\sigma_9$$

$$A_{32} = \sin\alpha_4\,\sigma_8 + \cos\alpha_4\,\cos\theta_4\,\sigma_7 - \cos\alpha_4\,\sin\theta_4\,\sigma_9$$

$$A_{33} = \cos\alpha_4\,\sigma_8 - \sin\alpha_4\,\cos\theta_4\,\sigma_7 + \sin\alpha_4\,\sin\theta_4\,\sigma_9$$

$$A_{34} = d_1 + d_4\,\sigma_8 + d_2\,\cos\alpha_1 + d_3\,\sigma_{16} + a_4\,\sin\theta_4\,\sigma_7 + a_4\,\cos\theta_4\,\sigma_9 + a_2\,\sin\alpha_1\,\sin\theta_2 + a_3\,\sin\theta_3\,\sigma_{17} + a_3\,\sin\alpha_1\,\cos\theta_3\,\sin\theta_2$$

$$\sigma_1 = \cos\alpha_3\,\sin\theta_3\,\sigma_{13} - \sin\alpha_3\,\sigma_{10} + \cos\alpha_3\,\cos\theta_3\,\sigma_{12}$$

$$\sigma_2 = \sin\alpha_3\,\sigma_{11} + \cos\alpha_3\,\sin\theta_3\,\sigma_{15} + \cos\alpha_3\,\cos\theta_3\,\sigma_{14}$$

$$\sigma_3 = \cos\alpha_3\,\sigma_{10} + \sin\alpha_3\,\sin\theta_3\,\sigma_{13} + \sin\alpha_3\,\cos\theta_3\,\sigma_{12}$$

$$\sigma_4 = \sin\alpha_3\,\sin\theta_3\,\sigma_{15} - \cos\alpha_3\,\sigma_{11} + \sin\alpha_3\,\cos\theta_3\,\sigma_{14}$$

$$\sigma_5 = \sin\theta_3\,\sigma_{12} - \cos\theta_3\,\sigma_{13}$$

$$\sigma_6 = \sin\theta_3\,\sigma_{14} - \cos\theta_3\,\sigma_{15}$$

$$\sigma_7 = \sin\alpha_3\,\sigma_{16} + \cos\alpha_3\,\cos\theta_3\,\sigma_{17} - \cos\alpha_3\,\sin\alpha_1\,\sin\theta_2\,\sin\theta_3$$

$$\sigma_8 = \cos\alpha_3\,\sigma_{16} - \sin\alpha_3\,\cos\theta_3\,\sigma_{17} + \sin\alpha_1\,\sin\alpha_3\,\sin\theta_2\,\sin\theta_3$$

$$\sigma_9 = \sin\theta_3\,\sigma_{17} + \sin\alpha_1\,\cos\theta_3\,\sin\theta_2$$

$$\sigma_{10} = \cos\alpha_2\,\sin\alpha_1\,\sin\theta_1 + \sin\alpha_2\,\cos\theta_1\,\sin\theta_2 + \cos\alpha_1\,\sin\alpha_2\,\cos\theta_2\,\sin\theta_1$$

$$\sigma_{11} = \cos\alpha_2\,\sin\alpha_1\,\cos\theta_1 - \sin\alpha_2\,\sin\theta_1\,\sin\theta_2 + \cos\alpha_1\,\sin\alpha_2\,\cos\theta_1\,\cos\theta_2$$

$$\sigma_{12} = \cos\alpha_2\,\cos\theta_1\,\sin\theta_2 - \sin\alpha_1\,\sin\alpha_2\,\sin\theta_1 + \cos\alpha_1\,\cos\alpha_2\,\cos\theta_2\,\sin\theta_1$$

$$\sigma_{13} = \cos\theta_1\,\cos\theta_2 - \cos\alpha_1\,\sin\theta_1\,\sin\theta_2$$

$$\sigma_{14} = \sin\alpha_1\,\sin\alpha_2\,\cos\theta_1 + \cos\alpha_2\,\sin\theta_1\,\sin\theta_2 - \cos\alpha_1\,\cos\alpha_2\,\cos\theta_1\,\cos\theta_2$$

$$\sigma_{15} = \cos\theta_2\,\sin\theta_1 + \cos\alpha_1\,\cos\theta_1\,\sin\theta_2$$

$$\sigma_{16} = \cos\alpha_1\,\cos\alpha_2 - \sin\alpha_1\,\sin\alpha_2\,\cos\theta_2$$

$$\sigma_{17} = \cos\alpha_1\,\sin\alpha_2 + \cos\alpha_2\,\sin\alpha_1\,\cos\theta_2$$

Part (c)

$$\text{Position of End-Effector Frame} = {}^0O_4 = \begin{bmatrix} A_{14} \\ A_{24} \\ A_{34} \end{bmatrix}$$

$$\text{Orientation of End-Effector Frame} = {}^0R_4 = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}$$

Where, the entries A_{ij} are as specified in part (b).

Task 4.4

The required MATLAB function for forward kinematics is given below:

```
function [x, y, z, R, theta, phi] = findPincher(jointAngles)
T =  [
    0.5000*cos(jointAngles(1) + jointAngles(2) + jointAngles(3) + jointAngles(4)) + ...
       0.5000*cos(jointAngles(2) - jointAngles(1) + jointAngles(3) + jointAngles(4)), - ...
       0.5000*sin(jointAngles(2) - jointAngles(1) + jointAngles(3) + jointAngles(4)) - ...
       0.5000*sin(jointAngles(1) + jointAngles(2) + jointAngles(3) + jointAngles(4)),  ...
       sin(jointAngles(1)), 5.4000*cos(jointAngles(1) - jointAngles(2)) + ...
       5.4000*cos(jointAngles(1) + jointAngles(2) + jointAngles(3)) + ...
       5.4000*cos(jointAngles(2) - jointAngles(1) + jointAngles(3)) + ...
       3.8000*cos(jointAngles(1) + jointAngles(2) + jointAngles(3) + jointAngles(4)) + ...
       5.4000*cos(jointAngles(1) + jointAngles(2)) + 3.8000*cos(jointAngles(2) - ...
       jointAngles(1) + jointAngles(3) + jointAngles(4));
    0.5000*sin(jointAngles(1) + jointAngles(2) + jointAngles(3) + jointAngles(4)) - ...
       0.5000*sin(jointAngles(2) - jointAngles(1) + jointAngles(3) + jointAngles(4)),   ...
       0.5000*cos(jointAngles(1) + jointAngles(2) + jointAngles(3) + jointAngles(4)) - ...
       0.5000*cos(jointAngles(2) - jointAngles(1) + jointAngles(3) + jointAngles(4)), ...
       -cos(jointAngles(1)), 5.4000*sin(jointAngles(1) - jointAngles(2)) - ...
       3.8000*sin(jointAngles(2) - jointAngles(1) + jointAngles(3) + jointAngles(4)) + ...
       5.4000*sin(jointAngles(1) + jointAngles(2) + jointAngles(3)) - ...
       5.4000*sin(jointAngles(2) - jointAngles(1) + jointAngles(3)) + ...
       3.8000*sin(jointAngles(1) + jointAngles(2) + jointAngles(3) + jointAngles(4)) + ...
       5.4000*sin(jointAngles(1) + jointAngles(2));
    sin(jointAngles(2) + jointAngles(3) + jointAngles(4)), ...
                                                                    ...
       cos(jointAngles(2) + jointAngles(3) + jointAngles(4)),    6.1232e-17, ...

       7.6000*sin(jointAngles(2) + jointAngles(3) + jointAngles(4)) + ...
       10.8000*sin(jointAngles(2) + jointAngles(3)) + 10.8000*sin(jointAngles(2)) + 5.4000;
    0, ...

       0,             0, ...

       1
    ];
x = T(1,4);
y = T(2,4);
z = T(3, 4);
R = T(1:3,1:3);
theta = jointAngles(1);
phi = sum(jointAngles(2:4));
end
```

Task 4.5

The following code was added below in the given `PincherModel.m` file which generates 5 random configurations for the robot.

```
for i = 1:5
    disp(strcat('Random Configuration Number: ', num2str(i)));
    config = randomConfiguration(robot);
    disp("Our Calculations:");
    config_Arr = [0, 0, 0, 0];
    config_Arr(1) = getfield(config,{1}, 'JointPosition');
```

```
    config_Arr(2) = getfield(config,{2}, 'JointPosition');
    config_Arr(3) = getfield(config,{3}, 'JointPosition');
    config_Arr(4) = getfield(config,{4}, 'JointPosition');
    [x, y, z, R, theta, phi] = findPincher(config_Arr)
    figure;
    show(robot,config);

    % Determine the pose of end-effector in provided configuration
    poseNow = getTransform(robot,config,"body4");

    % Display position and orientation of end-effector

    disp('The position of end-effector is:');
    disp('');
    disp(['X: ', num2str(poseNow(1,4))]);
    disp('');
    disp(['Y: ', num2str(poseNow(2,4))]);
    disp('');
    disp(['Z: ', num2str(poseNow(3,4))]);
    disp(' ');
    disp(['R: ']);
    poseNow(1:3,1:3)
    disp(' ');
    disp('The orientation angle is given with respect to the x-axis of joint 2:');
    disp('');
    poseNow01 = getTransform(robot,config,"body1");
    R14 = poseNow01(1:3,1:3)'*poseNow(1:3,1:3);
    angle = rad2deg(atan2(R14(2,1),R14(1,1)));
    disp(['Angle: ',num2str(angle), ' degrees.']);
end
```

We get the following calculated and obtained values for our 5 different configurations:

No.	Obtained				Calculated			
	x	y	z	R	x	y	z	R
1	23.4303	18.3029	262.3922	$\begin{bmatrix} 0.3047 & 0.7264 & -0.6160 \\ 0.2383 & 0.5681 & 0.7877 \\ 0.9222 & -0.3868 & 0.0 \end{bmatrix}$	23.404	18.303	262.392	$\begin{bmatrix} 0.3047 & 0.7264 & -0.6160 \\ 0.2383 & 0.5681 & 0.7877 \\ 0.9222 & -0.3868 & 0.0 \end{bmatrix}$
2	50.9604	32.7114	70.9767	$\begin{bmatrix} -0.2543 & 0.8022 & -0.5402 \\ -0.1633 & 0.5149 & 0.8415 \\ 0.9532 & 0.3022 & 0.0 \end{bmatrix}$	50.96	32.711	70.977	$\begin{bmatrix} -0.2543 & 0.8022 & -0.5402 \\ -0.1633 & 0.5149 & 0.8415 \\ 0.9532 & 0.3022 & 0.0 \end{bmatrix}$
3	153.9145	-60.8438	120.451	$\begin{bmatrix} -0.5925 & -0.7168 & -0.3676 \\ 0.2342 & 0.2834 & -0.93 \\ 0.7708 & -0.6371 & 0.0 \end{bmatrix}$	153.914	-60.844	120.451	$\begin{bmatrix} -0.5925 & -0.7168 & -0.3676 \\ 0.2342 & 0.2834 & -0.9300 \\ 0.7708 & -0.6371 & 0.0 \end{bmatrix}$
4	-120.9881	-13.7822	57.0917	$\begin{bmatrix} 0.9639 & -0.2412 & 0.1132 \\ 0.1098 & -0.0275 & -0.9936 \\ 0.2428 & 0.9701 & 0.0 \end{bmatrix}$	-120.988	-13.782	57.092	$\begin{bmatrix} 0.9639 & -0.2412 & 0.1132 \\ 0.1098 & -0.0275 & -0.9936 \\ 0.2428 & 0.9701 & 0.0 \end{bmatrix}$
5	-11.8649	-202.9725	18.4770	$\begin{bmatrix} -0.0380 & 0.0443 & -0.9983 \\ -0.6507 & 0.7570 & 0.0584 \\ 0.7583 & 0.6519 & 0.0 \end{bmatrix}$	-11.865	-202.972	184.770	$\begin{bmatrix} -0.0380 & 0.0443 & -0.9983 \\ -0.6507 & 0.7570 & 0.0584 \\ 0.7583 & 0.6519 & 0.0 \end{bmatrix}$

Table 2: Calculated and Obtained $[x, y, z, R]$ for 5 Random Robot Configurations

Where the values of x, y, z are in `mm`. We see that both the Calculated and Obtained values are the same.

Task 4.6

Joint ID	DH Joint Angle (θ_i)	Servo Angle (ψ_i)	Aligned directions of rotation (Yes/No)
1	0^o	-90^o	Yes
2	0^o	-90^o	Yes
3	0^o	0^o	Yes
4	0^o	0^o	Yes

Table 3: Linear mapping between servo angles and DH angles

Joint ID	Minimum Joint Angle		Maximum Joint Angle	
	Servo Angle	DH Joint Angle	Servo Angle	DH Joint Angle
1	-150^o	-60^o	150^o	240^o
2	-150^o	-60^o	150^o	240^o
3	-150^o	-150^o	150^o	150^o
4	-150^o	-150^o	150^o	150^o

Table 4: Joint Limits

Task 4.7

The following code generates the required figures for identifying the reachable workspace.

```
N = 500;
arr = transpose(0:1/(N-1):1);
theta = [ones(1,N^2); ones(1,N^2); ones(1,N^2); ones(1,N^2)];
min_theta = [-60, -60, -150, -150] .* (pi/180);
max_theta = [240, 240, 150, 150] .* (pi/180);
x = ones(1, N^2);
y = ones(1, N^2);
z = ones(1, N^2);
theta(3,:) = min_theta(3) + (max_theta(3)-min_theta(3)) * rand(N^2,1);
theta(4,:) = min_theta(4) + (max_theta(4)-min_theta(4)) * rand(N^2,1);
theta(1,:) = min_theta(1) + (max_theta(1)-min_theta(1)) * rand(N^2,1);
theta(2,:) = min_theta(2) + (max_theta(2)-min_theta(2)) * rand(N^2,1);
for i = 1:N^2
    [x(i), y(i), z(i)] = findPincher(theta(1:4,i));
end
figure;
scatter3(x, y, z, 10, [0 0 1], 'filled');
[X, Y, Z] = sphere(25);
figure;
hold on;
scatter3(x, y, z, 10, [0 0 1], 'filled');
s = surf(X.*((max(x) - min(x))/2 + 1) + (max(x) - min(x))/2 + min(x), Y.*((max(y) - ...
   min(y))/2 + 1) + (max(y) - min(y))/2 + min(y), (Z).*((max(z) - min(z))/2 + 2) + ...
    (max(z) - min(z))/2 + min(z));
s.FaceAlpha = 0.8;
s.FaceColor = [1 0 1];
```

The following Figure is obtained containing randomized points for our reachable workspace.

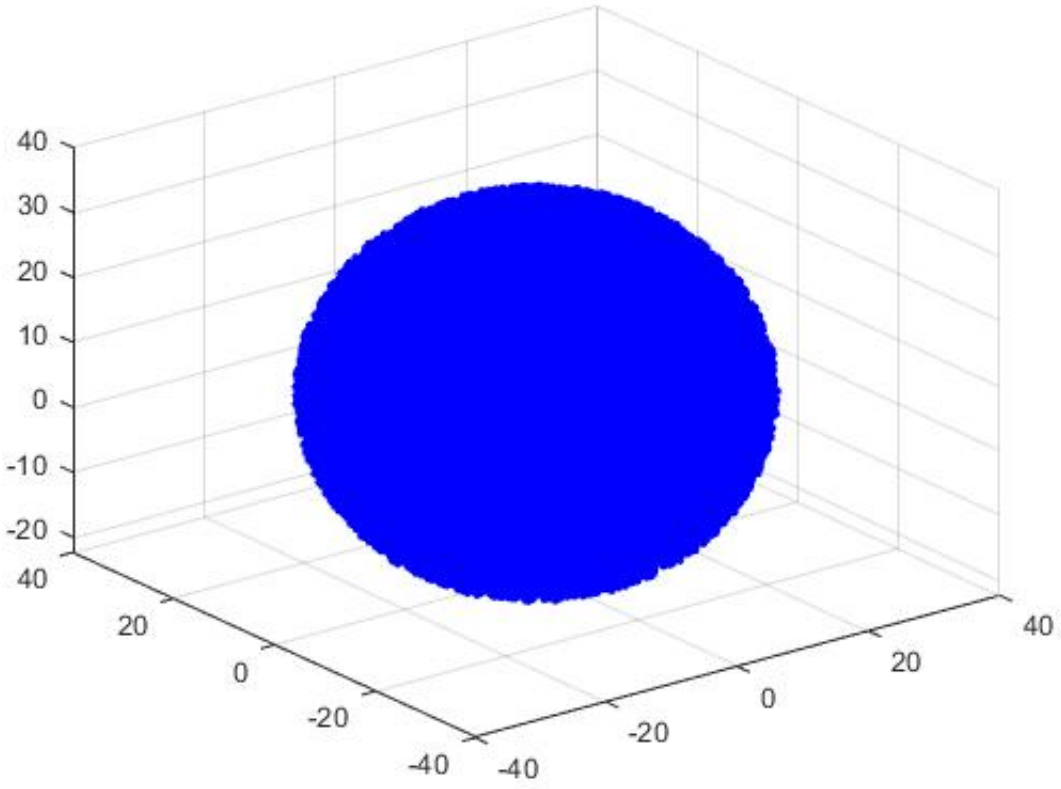

Figure 2: Reachable Workspace

We can see that the reachable workspace estiamtes a sphere. So our workspace can be expressed as a sphere that encloses all the points at which our Robot can move, Such a sphere is shown in the following figure, where a circle encloses all the plotted points.

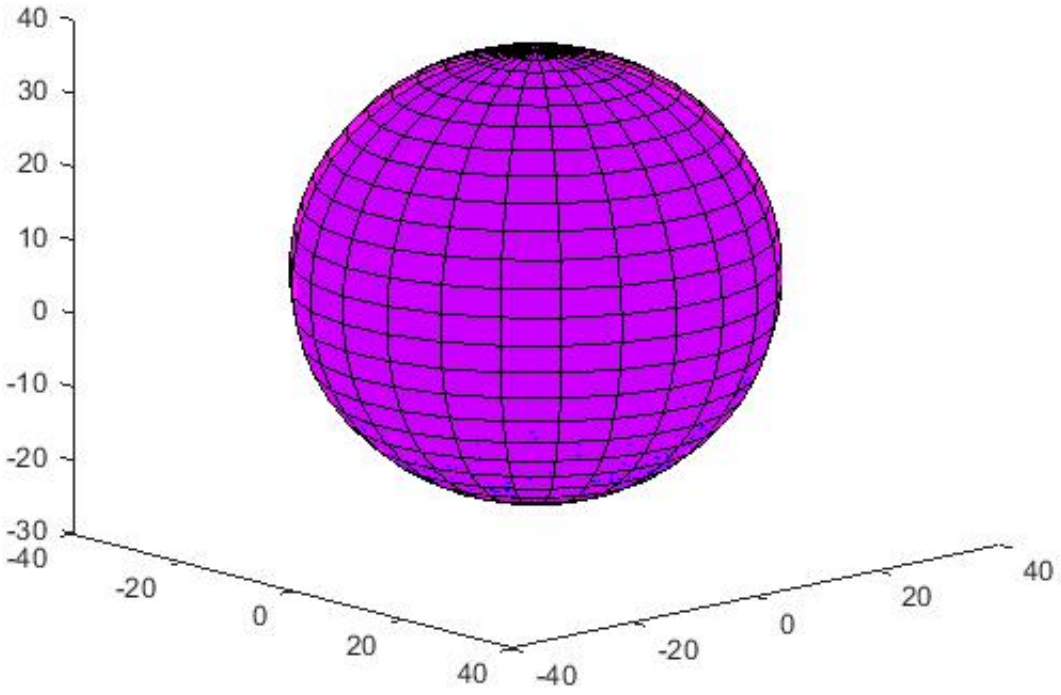

Figure 3: Reachable Workspace (Approximated as a Sphere)

The Projection of the point onto the $X - Y$ plane can be seen in the following figure.

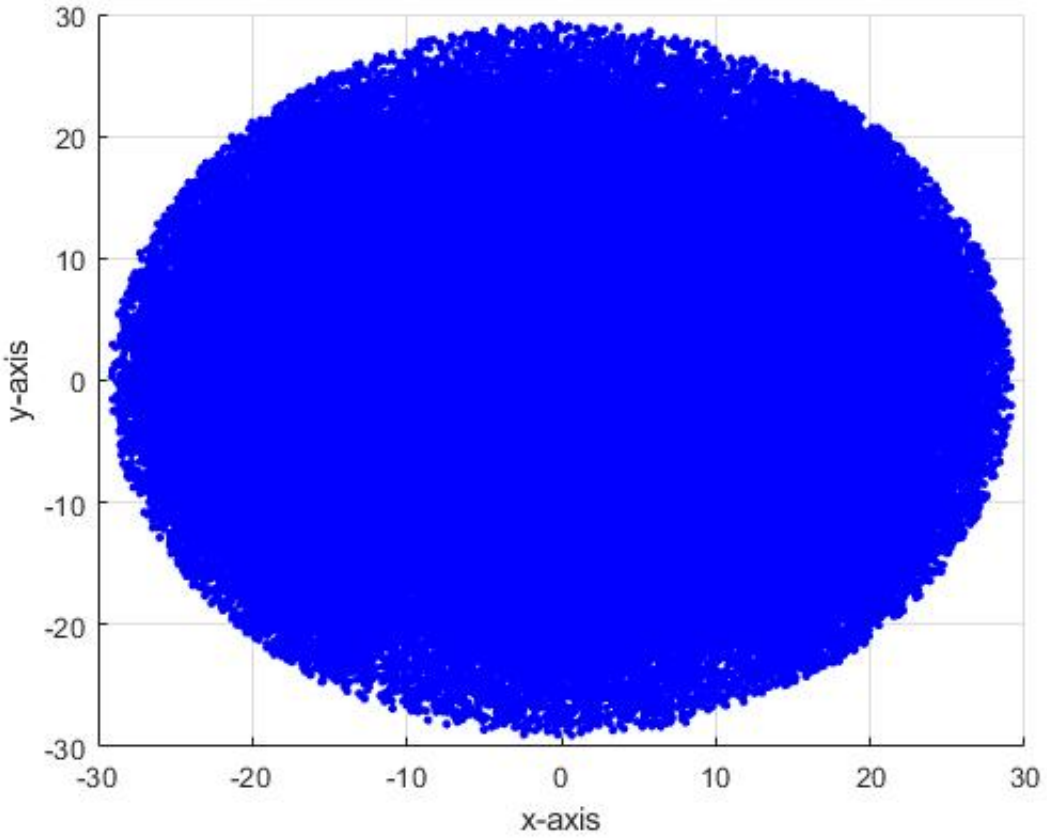

Figure 4: Reachable Workspace (Project on $X - Y$ plane)

The maximum horizontal reach according to our identified workspace is $[-30, 30]$

Task 4.8

```
function errorCode = setPosition(jointAngles)
    new_theta = zeros(1,4); % Array for JointAngles Mapped to [-pi, pi]
    errorCode = 0; % 0 -> No error yet
    for i=1:4
        % Mapping each jointangle to [-pi,pi]
        new_theta(i) = mod(jointAngles(i)+pi, 2*pi) - pi;
        % Condition for invalid input angle
        if ~(new_theta(i) <= pi && new_theta(i) >= -pi)
            disp(strcat('Angle ', num2str(i), ' Out of range'));
            % Error code = i -> ith Joint is out of range
            errorCode = i;
        end
    end
    % Passing the Joint Angles to Robot is no Error Occured
    if errorCode == 0
        % Mapping DH angles to Servo Angles
        map_theta = zeros(1,5);
        map_theta(1) = jointAngles(1) - pi/2;
```

```
        map_theta(2) = jointAngles(2) - pi/2;
        map_theta(3) = jointAngles(3);
        map_theta(4) = jointAngles(4);
        % Connecting to Robot and passing the theta information
        % to Robot for execution with a certain speed (64 for every joint
        % in this case
        arb = Arbotix('port', 'COM3', 'nservos', 5)
        arb.setpos(map_theta, [64, 64, 64, 64, 64]);

    end

end
```

Task 4.9

The following code command the robot to move to the 5 selected joint configurations. It also gives the values of $[x, y, z, R]$.

```
    thetas = zeros(5,4);
    thetas(1,:) = [pi/2, pi/2, pi/2, pi/2];
    thetas(2,:) = [-pi/4, pi/4, 0, 0];
    thetas(3,:) = [-pi/4, pi/4, pi/2, -pi/2];
    thetas(4,:) = [-pi/4, pi/4, -pi/2, pi/2];
    thetas(5,:) = [pi/4, 5*pi/6, 0, 0];
    for i=5:5
        [x, y, z, R, theta, phi] = findPincher(thetas(i,:))
        setPosition(thetas(i,:));
        pause(5);
    end
```

No.	Calculated (cm)			Measured (cm)			Error (cm)
	x	y	z	x	y	z	d
1	0	-10.8	8.6	0	-10.4	8.4	0.447
2	14.6	-14.6	26.04	15	-15	24.8	0.947
3	3.8	-3.8	26.04	3.5	3.4	25.6	0.751
4	14.6	-14.6	10.77	15	-15	10.2	0.803
5	-17.88	-17.88	20	18	-18	19.6	0.434

Table 5: Calculated & Measured End-Effector Positions (with Error)

There are indeed errors in the measurement. The most prominent of them is the human error in measuring the positions because of limitations of instruments and ways to measure the position. Secondly, the torque of the motors also come into play. There are also uncertainties related to Servo Motor Angles which can cause the error.