Introduction to Robotics
Lab 5 - Inverse Kinematics

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

Lab Report

Task 5.1

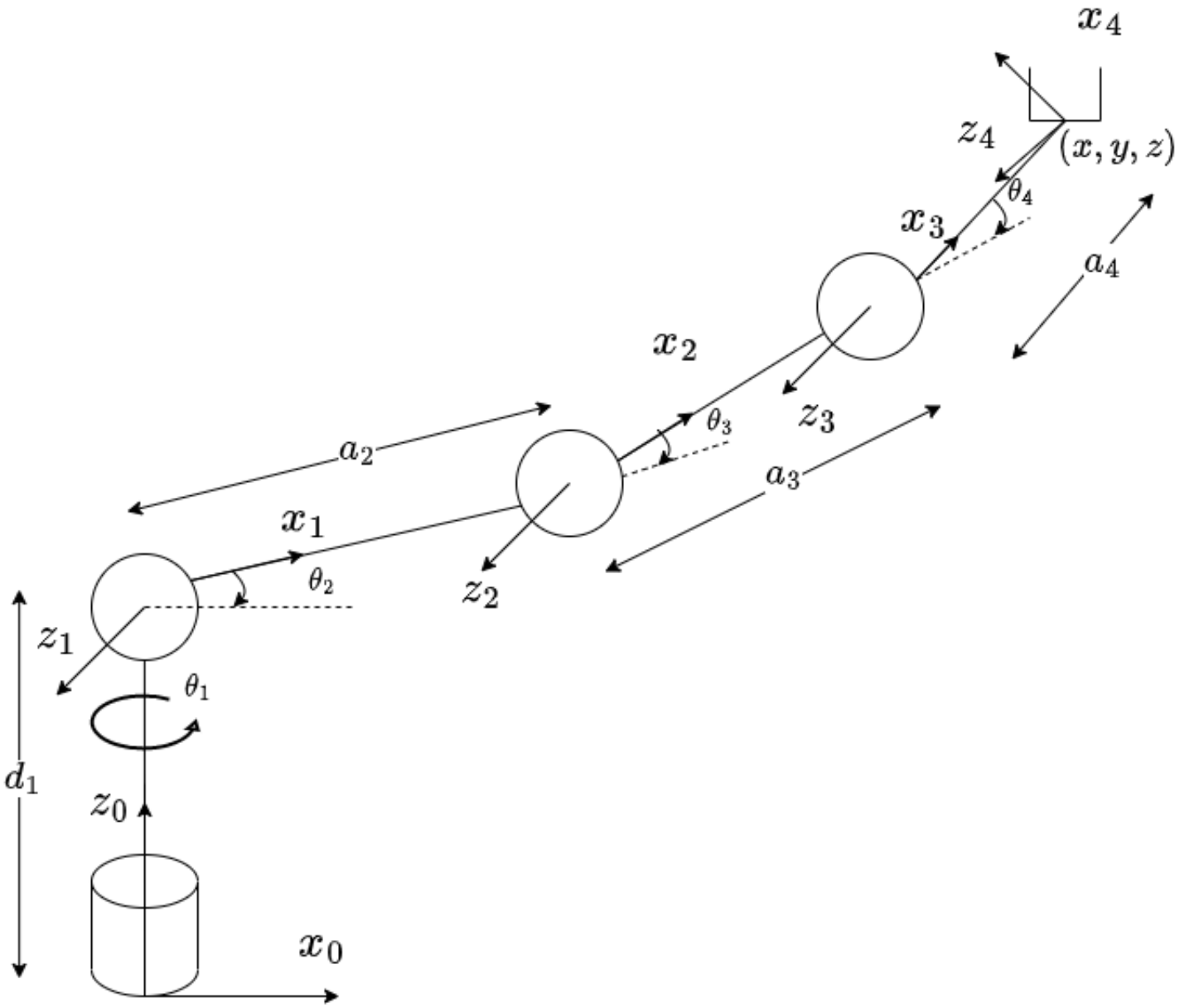

Figure 1: Schematic for PhantomX Arm Pincher

Projecting the end-effector point (x, y, z) onto $x_0 - y_0$ plane, we get:

$$\tan\theta_1 = \frac{y}{x}$$

$$\boxed{\theta_1 = \arctan\left(\frac{y}{x}\right),\ \theta_1 = \pi + \arctan\left(\frac{y}{x}\right)}$$

Now we find the x and y coordinates of frame $\{3\}$. From basic geometry we can see from figure 1 that:

$$x = x_3 + a_4 \cos(\theta_2 + \theta_3 + \theta_4)$$

$$\implies x_3 = x - a_4 \cos\phi$$

Similarly,

$$y_3 = y - a_4 \sin\phi$$

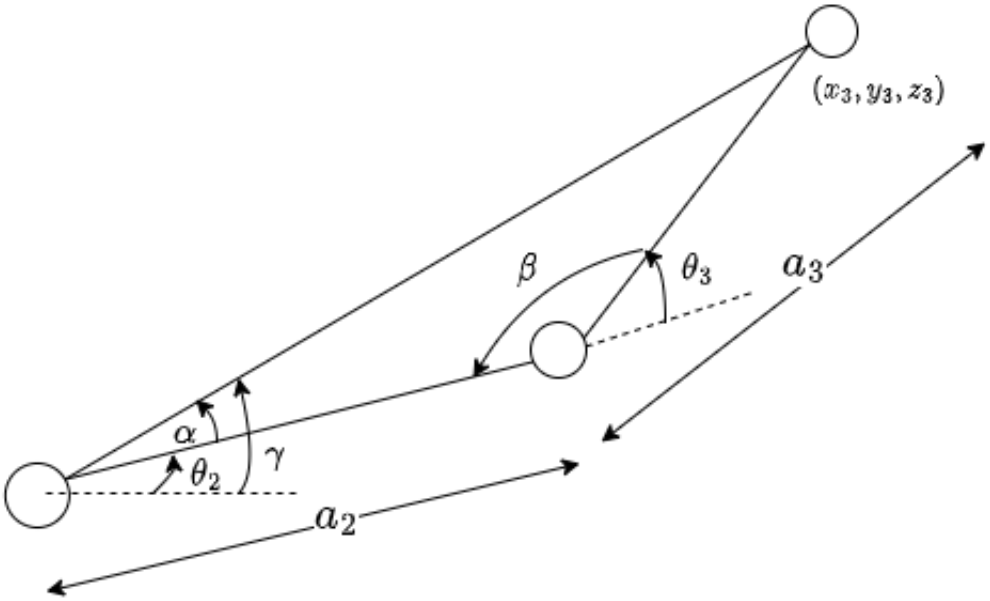

Figure 2: Joints 2, 3 and 4

Now consider figure 2 which shows the Joint 2, 3 and 4 from our manipulator. From cosine law, we know that:

$${}^1x_3^2 + {}^1y_3^2 = a_2^2 + a_3^2 - 2a_2a_3\cos\beta$$

$$\implies \beta = \arccos\left(\frac{a_2^2 + a_3^2 - {}^1x_3^2 - {}^1y_3^2}{2a_2a_3}\right)$$

Similarly, we have:

$$\implies \alpha = \arccos\left(\frac{a_2^2 - a_3^2 + {}^1x_3^2 + {}^1y_3^2}{2a_2\sqrt{{}^1x_3^2 + {}^1y_3^2}}\right)$$

To fid ${}^1x_3, {}^1y_3$, we see from figure 1 that:

$${}^1y = z - d_1$$

$$\implies {}^1y_3 = z - d_1 - a_4\sin\phi$$

We also see that upon projecting the end effector frame onto the $x_0 - y_0$ plane, we have:

$${}^1x = \sqrt{x^2 + y^2}$$

$$\implies {}^1x_3 = \sqrt{x^2 + y^2} - a_4\cos\phi$$

Hence, we have:

$$\theta_2 = \gamma - \alpha,\ \gamma + \alpha$$

$$\boxed{\implies \theta_2 = \arctan\left(\frac{z - d_1 - a_4\sin\phi}{\sqrt{x^2+y^2} - a_4\cos\phi}\right) \pm \arccos\left(\frac{a_2^2 - a_3^2 + \left(\sqrt{x^2+y^2} - a_4\cos\phi\right)^2 + (z - d_1 - a_4\sin\phi)^2}{2a_2\sqrt{\left(\sqrt{x^2+y^2} - a_4\cos\phi\right)^2 + (z - d_1 - a_4\sin\phi)^2}}\right)}$$

$$\theta_3 = \pi - \beta,\ \beta + \pi$$

$$\boxed{\theta_3 = \pi - \arccos\left(\frac{a_2^2 + a_3^2 - \left(\sqrt{x^2+y^2} - a_4\cos\phi\right)^2 - (z - d_1 - a_4\sin\phi)^2}{2a_2a_3}\right)}$$

$$\boxed{\theta_3 = \arccos\left(\frac{a_2^2 + a_3^2 - \left(\sqrt{x^2+y^2} - a_4\cos\phi\right)^2 - (z - d_1 - a_4\sin\phi)^2}{2a_2a_3}\right) - \pi}$$

Now that we know θ_2 & θ_3, we can find θ_4 using:

$$\boxed{\theta_4 = \phi - \theta_2 - \theta_3}$$

From the equations, we see that the possible solutions for the Inverse Kinematics comes out to be 4. From simplicity, the equations are written in terms of β, γ, α. These are:

Solution 1

$$\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\\ \theta_4\end{bmatrix} = \begin{bmatrix}\arctan\left(\frac{y}{x}\right)\\ \pi - \beta\\ \gamma - \alpha\\ \phi - \theta_2 - \theta_3\end{bmatrix}$$

Solution 2

$$\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\\ \theta_4\end{bmatrix} = \begin{bmatrix}\arctan\left(\frac{y}{x}\right)\\ \beta - \pi\\ \gamma + \alpha\\ \phi - \theta_2 - \theta_3\end{bmatrix}$$

Solution 3

$$\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\\ \theta_4\end{bmatrix} = \begin{bmatrix}\pi + \arctan\left(\frac{y}{x}\right)\\ -(\pi - \beta)\\ \pi - (\gamma - \alpha)\\ \phi - \theta_2 - \theta_3\end{bmatrix}$$

Solution 4

$$\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\\ \theta_4\end{bmatrix} = \begin{bmatrix}\pi + \arctan\left(\frac{y}{x}\right)\\ -(\beta - \pi)\\ \pi - (\gamma + \alpha)\\ \phi - \theta_2 - \theta_3\end{bmatrix}$$

Task 5.2

The required code is:

```
function q = findJointAngles(x,y,z,phi)
    q = zeros(4,4);
    a4 = 7.6; d1 = 5.4; a2 = 10.8; a3 = 10.8;
    x13 = a4*cos(phi) - sqrt(x^2 + y^2);
    y13 = a4*sin(phi) - z + d1;
    r = sqrt(x13^2 + y13^2);
    alpha = acos((a2^2 + (r^2-a3^2))/(2*a2*r));
    beta = acos((a2^2 + a3^2 - r^2)/(2*a2*a3));

    % Solution Set 1
    theta1 = atan(y/x); % Theta 1 (1)
    theta3 = pi - beta;
    gamma = atan2(z - d1 -a4*sin(phi),( sqrt(x^2 + y^2)) -a4*cos(phi));
    theta2 = gamma - alpha;
    theta4 = phi - theta2 - theta3;
    q(1,:) = [theta1, theta2, theta3, theta4];
    % Solution Set 2
    theta1 = atan(y/x); % Theta 1 (1)
    theta3 = -(pi - beta);
    gamma = atan(y13/x13);
    theta2 = gamma + alpha;
    theta4 = phi - theta2 - theta3;
    q(2,:) = [theta1, theta2, theta3, theta4];
    % Solution Set 3
    theta1 = atan(y/x) + pi; % Theta 1 (2)
    theta3 = -(pi - beta);
    gamma = atan(y13/x13);
    theta2 = pi - (gamma - alpha);
    theta4 = phi - theta2 - theta3;
    q(3,:) = [theta1, theta2, theta3, theta4];
    % Solution Set 4
    theta1 = atan(y/x) + pi; % Theta 1 (2)
    theta3 = pi - beta;
    gamma = atan(y13/x13);
    theta2 = pi - (gamma + alpha);
    theta4 = phi - theta2 - theta3;
    q(4,:) = [theta1, theta2, theta3, theta4];

    end
```

Task 5.3

Several Functions were made for this task. Below are the codes for them.

```
function lst = getCurrentPose()
    arb = Arbotix('port', 'COM3', 'nservos', 5);
    lst = arb.getpos();
    lst = lst(1,1:4);
end
```

```
function out = isValid(jointAngles)
    out = 0;
    if isreal(jointAngles(1)) && isreal(jointAngles(2)) && isreal(jointAngles(3)) && ...
       isreal(jointAngles(4))
        out = 1;
    end
end
```

```
function out = checkJointLimits(jointAngles)
    new_theta = zeros(1,4); % Array for JointAngles Mapped to [-pi, pi]
    offset = [-pi/2 -pi/2 0 0];
    out = 1;
    for i=1:4
        new_theta(i) = jointAngles(i) + offset(i);
        new_theta(i) = mod(new_theta(i)+pi, 2*pi) - pi;
        % Condition for invalid input angle
        if ~(new_theta(i) < 5*pi/6 && new_theta(i) > -5*pi/6)
            out = 0;
            break;
        end
    end
end
```

```
function lst = findOptimalSolution(x, y, z, phi)
    current_angles = getCurrentPose();
%     current_angles = [-pi/3 -pi/4 pi/4 pi/4];
    current_angles(1) = current_angles(1) + pi/2;
    current_angles(2) = current_angles(2) + pi/2;
    current_angles = current_angles(:,1:4);
    IK_sol = findJointAngles(x, y, z, phi)
    IK_1 = IK_sol(1,:); % Inverse Kinematic Solution 1
    IK_2 = IK_sol(2,:); % Inverse Kinematic Solution 2
    IK_3 = IK_sol(3,:); % Inverse Kinematic Solution 3
    IK_4 = IK_sol(4,:); % Inverse Kinematic Solution 4
    d1 = sum(abs(current_angles - IK_1)); % Delta 1
    d2 = sum(abs(current_angles - IK_2)); % Delta 2
    d3 = sum(abs(current_angles - IK_3)); % Delta 3
    d4 = sum(abs(current_angles - IK_4)); % Delta 4
    d = min([d1, d2, d3, d4]); % Extract Min
    if d == d1 && checkJointLimits(IK_1) && isValid(IK_1)
        lst = IK_1;
    elseif d == d2 && checkJointLimits(IK_2) && isValid(IK_2)
        lst = IK_2;
    elseif d == d3 && checkJointLimits(IK_3) && isValid(IK_3)
        lst = IK_3;
    elseif d == d4 && checkJointLimits(IK_4) && isValid(IK_4)
        lst = IK_4;
    end
end
```

Task 5.4

Part (a)

Following are the 5 choosen points and their measured value after executing the angles as given by `findOptimalSolution.m`

No.	Actual Values (cm)				Measured Values (cm)			Error (cm)
	x	y	z	ϕ	x	y	z	d
1	16	-16	3	$\pi/4$	16.3	-16.1	3	0.3162
2	13	10	5	$\pi/6$	12.8	9.8	5.1	0.3
3	-3	4	8	$\pi/2$	-3.1	4.2	8	0.2236
4	12	5	2	$3\pi/4$	12	5	1.95	0.05
5	-5	-6	2	$5\pi/6$	-5.1	-5.8	1.8	0.3

Table 1: Calculated & Measured End-Effector Positions (with Error)

Part (b)

We see from the above tables that the accuracy has indeed improved by observing the error in the above table as compared to Lab 4. This is because of the fact that we are minimizing the change in joint angles with the help of our `findOptimalSolution.m` function. Since our result is optimal, therefore the error is also reduced too.

Part (c)

Yes, there are many such points whose all four solutions are realizable. However such points should not cause collisions between any links, should not be boundary points and all those points where singularity occurs. One such point is $[x, y, z] = [16, -16, 3]$. All the four Robot Configurations corresponding to the Inverse Kinematic Solutions for this point are realizable.