TECHNION
Israel Institute
of Technology

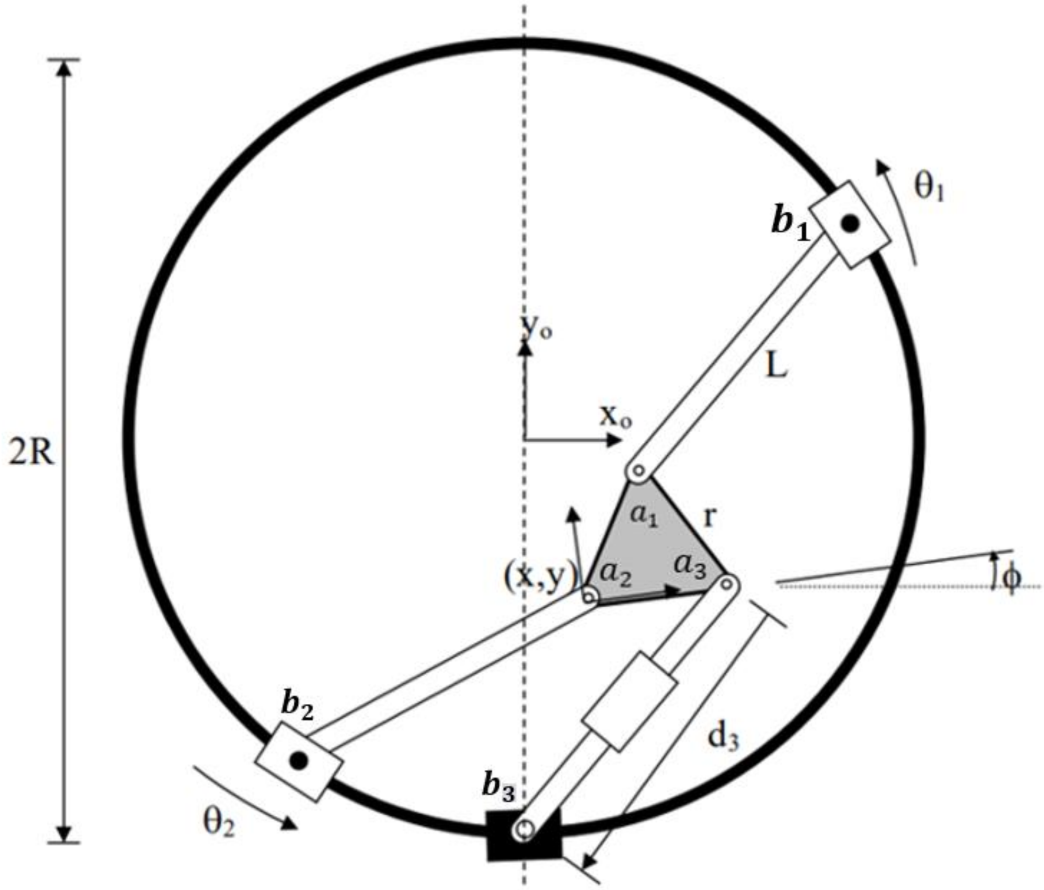
Kinematics, dynamics and control of robots

Homework project 2

Nir Manor 305229627
Sheng Huang 808837678

1.Inverse kinematics

Giving the following parallel robot:



The plate vertices:

$$a_1 = \left[x + r\cos\left(\phi + \frac{\pi}{3}\right), y + r\sin\left(\phi + \frac{\pi}{3}\right)\right]$$
$$a_2 = [x, y]$$
$$a_3 = [x + r\cos(\phi), y + r\sin(\phi)]$$
$$b_1 = [R\cos\theta_1, R\sin\theta_1]$$
$$b_2 = [R\cos\theta_2, R\sin\theta_2]$$
$$b_3 = [0, -R]$$

Calculating for θ_1:

$$\left(a_{1_x} - b_{1_x}\right)^2 + \left(a_{1_y} - b_{1_y}\right)^2 = L^2$$
$$b_{1_x}^2 + b_{1_y}^2 = R^2$$
$$b_{1_x} = \frac{a_{1_x}^2 + a_{1_y}^2 - 2a_{1_y}b_{1_y} + R^2 - L^2}{2a_{1_x}}$$
$$\left[\frac{a_{1_x}^2 + a_{1_y}^2 - 2a_{1_y}b_{1_y} + R^2 - L^2}{2a_{1_x}}\right]^2 + b_{1_y}^2 = R^2 \Rightarrow b_{1_y}$$
$$\theta_1 = atan2\left(b_{1_y}, b_{1_x}\right)$$

Calculating for θ_2:

$$\left(a_{2_x} - b_{2_x}\right)^2 + \left(a_{2_y} - b_{2_y}\right)^2 = L^2$$
$$b_{2_x}^2 + b_{2_y}^2 = R^2$$
$$b_{2_x} = \frac{a_{2_x}^2 + a_{2_y}^2 - 2a_{2_y}b_{2_y} + R^2 - L^2}{2a_{2_x}}$$

$$\left[\frac{a_{2_x}^2 + a_{2_y}^2 - 2a_{2_y}b_{2_y} + R^2 - L^2}{2a_{2_x}}\right]^2 + b_{2_y}^2 = R^2 \Rightarrow b_{2_y}$$
$$\theta_2 = atan2\left(b_{2_y}, b_{2_x}\right)$$

Calculating for d_3:

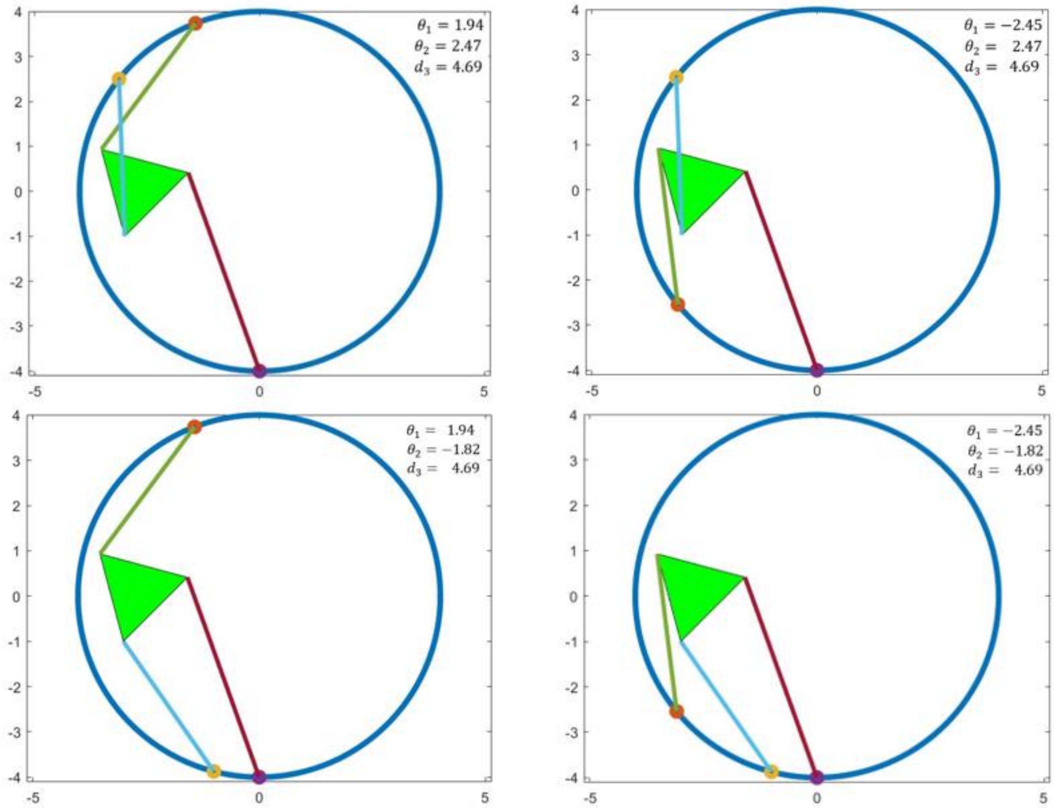
$$\left(a_{3_x}\right)^2 + \left(a_{3_y} + R\right)^2 = d_3^2$$

$$d_3 = \sqrt{\left(a_{3_x}\right)^2 + \left(a_{3_y} - R\right)^2}$$

For each one of the revolute joints, we get 2 solutions and for the prismatic joint we get 1. In total 4 possible solutions for the given position vector.

The script we were using:

- Circle – `this function draws circle with radius R with center at x0,y0.`
- Inverse_kin – `this function calculates the inverse kinematics.`
- Draw_inverse_kin - `this function draws the 4 solutions of the inverse kinematics.`



2.Forward kinematics

We will use the 3 constraints form that connect the plate vertexes with the joints on the slider:

For vertex 1:

1. $(a_{1_x} - b_{1_x})^2 + (a_{1_y} - b_{1_y})^2 = L^2$
2. $a_{1_x}^2 - 2a_{1_x}b_{1_x} + b_{1_x}^2 + a_{1_y}^2 - 2a_{1_y}b_{1_y} + b_{1_y}^2 = L^2$
3. $x^2 + y^2 + r^2 + R^2 + xrcos\phi - \sqrt{3}xrsin\phi + yrsin\phi + \sqrt{3}yrcos\phi - 2Rxc_1$ $-Rrc_1 \cos\phi + \sqrt{3}Rrc_1 \sin\phi - 2Rys_1 - Rrs_1 \sin\phi - \sqrt{3}Rrs_1 \cos\phi = L^2$

Where:
$c_1 = \cos(\theta_1), s_1 = \sin(\theta_1)$

For vertex 2:

4. $(a_{2_x} - b_{2_x})^2 + (a_{2_y} - b_{2_y})^2 = L^2$
5. $a_{2_x}^2 - 2a_{2_x}b_{2_x} + b_{2_x}^2 + a_{2_y}^2 - 2a_{2_y}b_{2_y} + b_{2_y}^2 = L^2$

Isolating for L^2:

6. $x^2 + y^2 + R^2 - 2xRc_2 - 2yRs_2 = L^2$

Isolating for R^2:

7. $L^2 + 2R(xc_1 + ys_1) - x^2 - y^2 = R^2$

Where:
$c_2 = \cos(\theta_2), s_2 = \sin(\theta_2)$

For vertex 3:

8. $(a_3 - b_{3_x})^2 + (a_{3_y} - b_{3_y})^2 = d_3^2$
9. $a_{3_x}^2 - 2a_{3_x}b_{3_x} + b_{3_x}^2 + a_{3_y}^2 - 2a_{3_y}b_{3_y} + b_{3_y}^2 = d_3^2$
10. $R^2 + 2Rrsin(\phi) + 2Ry + r^2 + 2rxcos(\phi) + 2rysin(\phi) + x^2 + y^2 = d_3^2$

Isolating for R^2:

11. $d_3^2 - 2Rrsin(\phi) - 2Ry - r^2 + 2r(xcos(\phi) + ysin(\phi)) + x^2 + y^2 = R^2$

$(Vertex1) - (Vertex2)$:
$Ax + By = E$

$(Vertex2) - (Vertex3)$:
$Cx + Dy = F$

$A = 2R(c_1 - c_2) - rcos(\phi) + \sqrt{3}rsin(\phi)$
$B = 2R(s_2 - s_1) + rsin(\phi) + \sqrt{3}rcos(\phi)$
$E = Rr(cos(\phi)c_1 + sin(\phi)s_1) + \sqrt{3}Rr(s_1cos(\phi) - c_1sin(\phi)) - r^2$
$C = -2Rc_2 - 2rcos(\phi)$
$D = -2R - 2Rs_2 - 2rsin(\phi)$
$F = L^2 - d3^2 + r^2 + 2Rrsin(\phi)$

$Ax + By = E$
$Cx + Dy = F$

$$x = \frac{DE - BF}{AD - BC}, \qquad y = \frac{AF - CE}{AD - BC}$$

By Substitute the solution for $\{x, y\}$ into equation 6, we end up with an equation that only contains $\sin(\phi)$and $\cos(\phi)$ as variables.
Using the tangent half-angle substitution ("Weierstrass Substitution") we can now substitute $cos(\phi)$ and $\sin(\phi)$ with t:

$$\cos\phi = \frac{1-t^2}{1+t^2}, \qquad \sin\phi = \frac{2t}{1+t^2}, \qquad \tan\frac{\phi}{2} = t$$

Once we assigned values to q, r, R, and L, we ended up with an equation that divided 6-degree polynomials:

$$\frac{t^6 + 4.9467t^5 + 2.9034t^4 + 4.8135t^3 - 5.2547t^2 - 0.026t + 0.4194}{t^6 - 3.2103t^5 + 6.2929t^4 - 7.5705t^3 + 7.1375t^2 - 4.3602t + 1.8446} = 0$$

To find the solutions for the inverse kinematics, we utilized the 'roots()' command on the numerator and verified that the denominator remains non-zero.

The roots we got are as follow:

```
           -4.59672
           -0.25833
            0.41421
             0.4472
- 0.47653 - 1.296i
- 0.47653 + 1.296i
```

As you can see, we obtained two complex roots (since the solutions are not physical we ignored them). Then, we used the relationships between the substitution variables to return to the trigonometric form of the solutions:

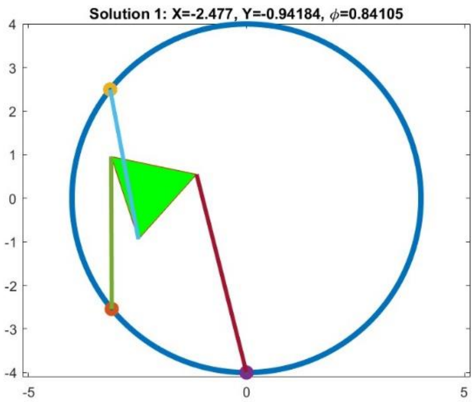
$$\phi = [48.1887^o \quad 45.00^o \quad -28.9696^o \quad -155.4535^o]$$

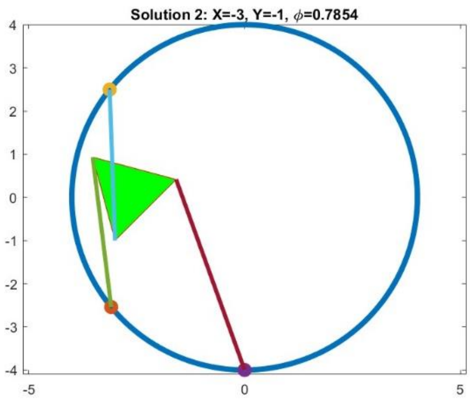
Finally, we got 4 solutions:

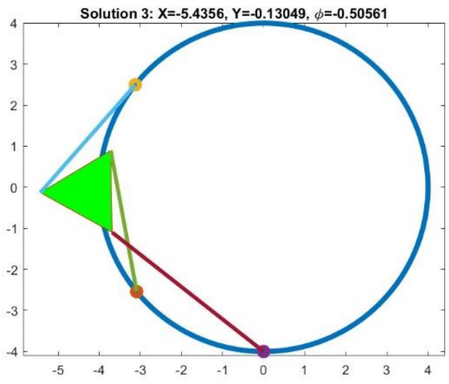
$$S_1 = \begin{bmatrix} -2.4770 \\ -0.9418 \\ 48.1887^o \end{bmatrix}, S_2 = \begin{bmatrix} -3.0000 \\ -1.0000 \\ 45.00^o \end{bmatrix}, S_3 = \begin{bmatrix} -5.4356 \\ -0.1305 \\ -28.9696^o \end{bmatrix}, \qquad S_4 = \begin{bmatrix} 0.1286 \\ 1.2060 \\ -155.4535^o \end{bmatrix}$$

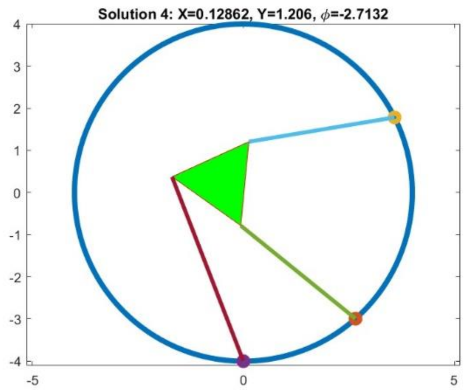
Next, we will utilize inverse kinematics on the solutions and compare them with one of the solutions obtained in question 1, to see if they match.
The joints vector we will compare with:

$$q^* = \begin{bmatrix} -140.547^o \\ 141.358^o \\ 4.690 \end{bmatrix}$$









The error in the joint's values(θ_1, θ_2, d_3):

$$E_{q,i} = q^* - f(S_i)$$

$$E_{q,1} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, E_{q,2} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, E_{q,3} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, E_{q,4} = \begin{bmatrix} -92.14^o \\ 114.89^o \\ 0 \end{bmatrix}$$

The error in the plate position (x, y, ϕ):

$$E_{p,i} = p^* - f(S_i)$$

$$E_{p,1} = \begin{bmatrix} 0.523 \\ 0.058 \\ 3.191° \end{bmatrix}, E_{p,2} = \begin{bmatrix} -0.444 \\ 0 \\ -6.359° \end{bmatrix} * 10^{-15}, E_{p,3} = \begin{bmatrix} -2.436 \\ 0.869 \\ -1.2910 \end{bmatrix}, E_{p,4} = \begin{bmatrix} 3.129 \\ 2.2060 \\ -200.455° \end{bmatrix}$$

The script we were using:

- `Forward_kin` – Function that calculates the forward kinematics.
- `Inverse_kin` – Function that calculates the forward kinematics.
- `Draw_state` – Function that draws the current system state by its tool and joint position.

3.Path Planning

$$X_a = [-3, -2{,}45^o]^T, \qquad X_b = [-2{,}0{,}0^o]^T, \qquad v_{a\to b} = const$$

Plate position:

$$x(t) = x_a + \frac{x_b - x_a}{T} t, \qquad y(t) = y_a + \frac{y_b - y_a}{T} t,$$

$$\phi(t) = \phi_a + \frac{\phi_b - \phi_a}{T} t$$

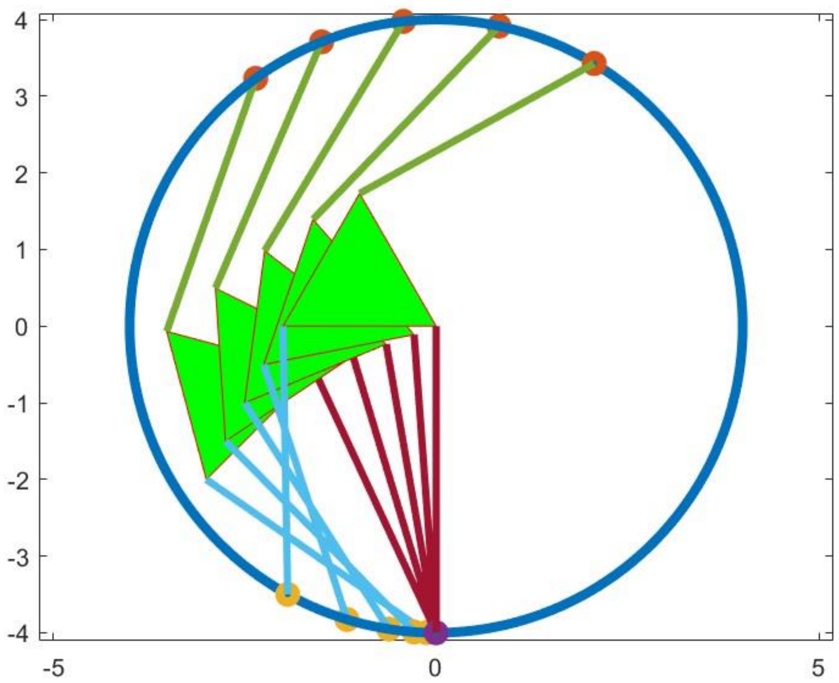
Plate velocity:

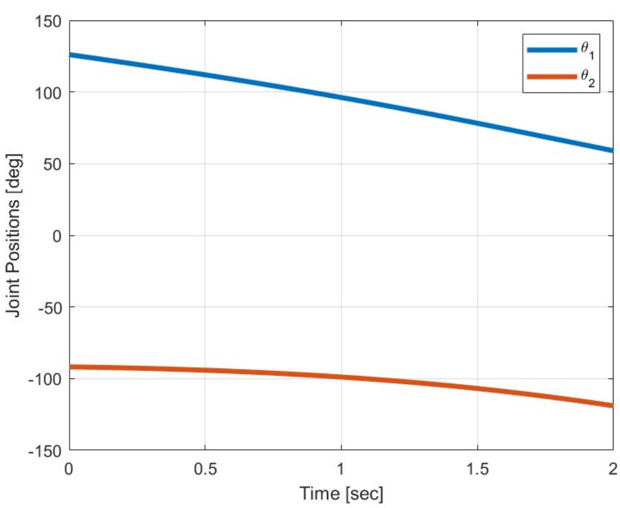
$$v = \left[\frac{x_b - x_a}{T}, \frac{y_b - y_a}{T}, \frac{\phi_b - \phi_a}{T}\right]^T$$

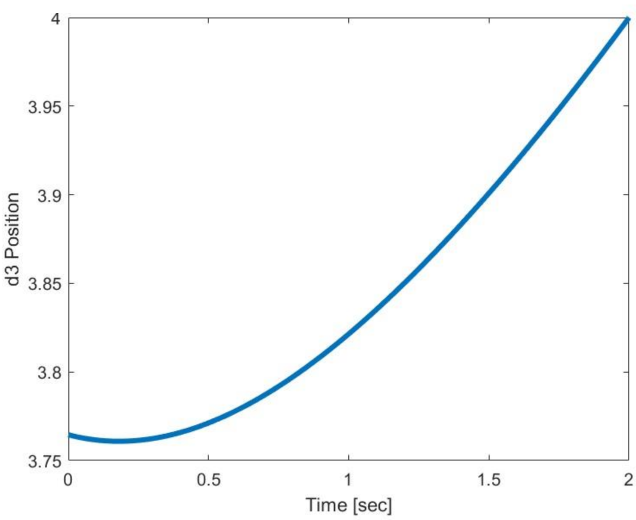
Plate acceleration:

$$a = [0,0,0]^T$$

At this stage, we have generated a position vector in MATLAB that represents the tool's movement, $X(t_0 \rightarrow T)$. By applying inverse kinematics to the points on the position vector, we derived 4 possible trajectories of actuator values that can produce the desired path, based on the orientation of the upper/lower elbows. However, some of these trajectories led to singularity regions, which caused discontinuities in the joints' positions, and others caused link intersections, creating collisions. To resolve this issue, we iterate through all the elbow options for each step and ensure that there are no sudden changes in joint values and that there is no intersection between the links.







The scripts and functions used in this section to perform the calculations and drawings are:

- **Main**: Calculates Q2 forward kinematics.
- **Pos plan**: This function computes the end effector's position as a time function.
- **Joint plan**: This function computes the joint positions as a function of time while under to the mechanical constraints.
- **"polyxpoly"**: A function from "Mapping Toolbox" library that we used in the Joint_plan function, to checks intersection points for lines or polygon edges.

4. Calculate the Jacobian

$$x = \begin{bmatrix} x \\ y \\ \phi \end{bmatrix}, q = \begin{bmatrix} \theta_1 \\ \theta_2 \\ d_3 \end{bmatrix}$$

$$a_1 = \left[x + r\,cos\left(\phi + \frac{\pi}{3}\right), y + r\,sin\left(\phi + \frac{\pi}{3}\right)\right]$$

$$a_2 = [x, y]$$

$$a_3 = [x + r\,cos(\phi), y + r\,sin(\phi)]$$

$$b_1 = [R\,cos\,\theta_1, R\,sin\,\theta_1]$$

$$b_2 = [R\,cos\,\theta_2, R\,sin\,\theta_2]$$

$$b_3 = [0, -R]$$

$$F = \begin{bmatrix} \left[x + r\,cos\left(\phi + \frac{\pi}{3}\right) - R\,cos\,\theta_1\right]^2 + \left[y + r\,sin\left(\phi + \frac{\pi}{3}\right) - R\,sin\,\theta_1\right]^2 - L^2 \\ [x - R\,cos\,\theta_2]^2 + [y - R\,sin\,\theta_2]^2 - L^2 \\ [x + r\,cos(\phi)]^2 + [y + r\,sin(\phi) - R]^2 - d_3^2 \end{bmatrix}$$

$$J_x = \frac{\partial \bar{F}}{\partial \bar{x}}$$

$$J_x(1,1) = 2x + 2r\,cos\left(\phi + \frac{\pi}{3}\right) - 2R\,cos\,\theta_1$$

$$J_x(1,2) = 2y + 2r\,sin\left(\phi + \frac{\pi}{3}\right) - 2R\,sin\,\theta_1$$

$$J_x(1,3) = 2r\left[sin\left(\phi + \frac{\pi}{3}\right)[R\,cos\,\theta_1 - x] + cos\left(\phi + \frac{\pi}{3}\right)[-R\,sin\,\theta_1 + y]\right]$$

$$J_x(2,1) = 2x - 2R\,cos\,\theta_2$$

$$J_x(2,2) = 2y - 2R\,sin\,\theta_2$$

$$J_x(2,3) = 0$$

$$J_x(3,1) = 2x + 2r\,cos\,\phi$$

$$J_x(3,2) = 2y + 2r\,sin(\phi) - 2R$$

$$J_x(3,3) = -2r[Rcos(\phi) - ycos(\phi) + xsin(\phi)]$$

$$J_q = \frac{\partial \bar{F}}{\partial \bar{q}}$$

$$J_q(1,1) = 2R\left[x \sin\theta_1 + r\cos\left(\phi + \frac{\pi}{3}\right)\sin\theta_1 - y\cos\theta_1 - r\sin\left(\phi + \frac{\pi}{3}\right)\cos\theta_1\right]$$

$$J_q(1,2) = 0$$

$$J_q(1,3) = 0$$

$$J_q(2,1) = 0$$

$$J_q(2,2) = 2R(x\sin\theta_2 - y\cos\theta_2)$$

$$J_q(2,3) = 0$$

$$J_q(3,1) = 0$$

$$J_q(3,2) = 0$$

$$J_q(3,3) = -2d_3$$

The script we were using:

- `Jacobian_calculation-` This function calculate parametrically the Jacobian for the tool space J_x and for the joint space J_q

5. Singularity Points

To achieve our goal of identifying singular positions and determining the direction of free motion, we focused on the tool's position within the domain specified by $\phi = 10^o$, $y = 0$ and $0 < x < 2$. In order to accomplish this, we use the $J_x(x,q)$ form previous question and checked the component values at each point in the domain using inverse kinematics for each one of the possible 4 solutions. We discovered that there is no solution for the inverse kinematics between $x = 0$ and $x = 0.5$ due to the circle's radius being 4 and the link's length being 3.5, resulting in an area ($r = 0.5$) inside the circle center, where there is no solution for x. Finally, we calculated the determinant for each point and plotted the determinant -$det(J_x)$ as a function of x to determine precisely when it equals zero within the domain.
To determine the direction of free motion, we computed the Eigenvalues and Eigenvectors of J_x, and identified the Eigenvectors that corresponded to Eigenvalues of 0.

These are the X location we have found:

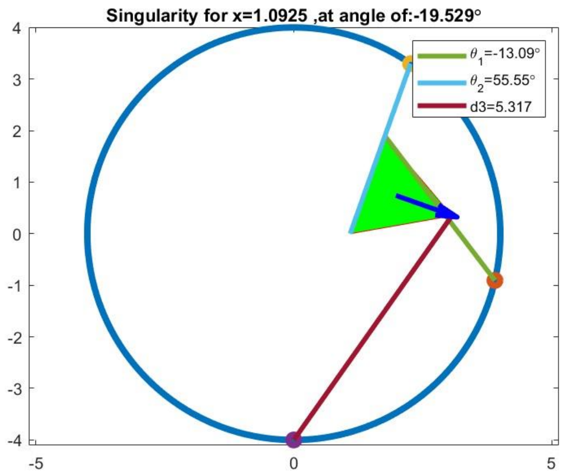
1) $X = 1.0925, \beta = -19.529°$ [1 1 1]

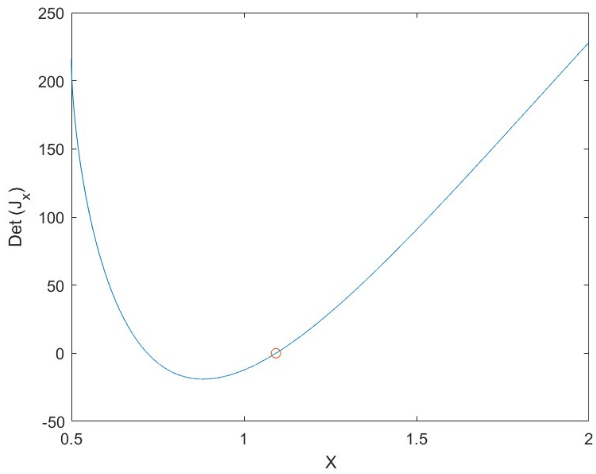
The Eigenvectors matrix:

$$\begin{bmatrix} 0.5740 & 0.8439 & 0.2911 \\ 0.1007 & -0.2993 & 0.8321 \\ -0.8127 & 0.4453 & -0.4721 \end{bmatrix}$$

The Eigenvector that corresponds to Eigenvalue of 0:

$$\begin{bmatrix} 0.8439 \\ -0.2993 \\ 0.4453 \end{bmatrix}$$





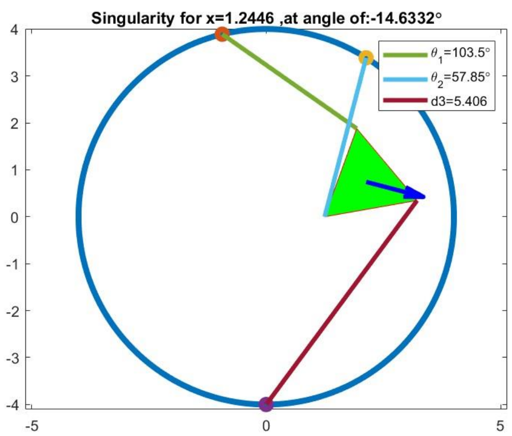
2) $X = 1.2446, \beta = -14.6332°$ `[-1 1 1]`

The Eigenvectors matrix:

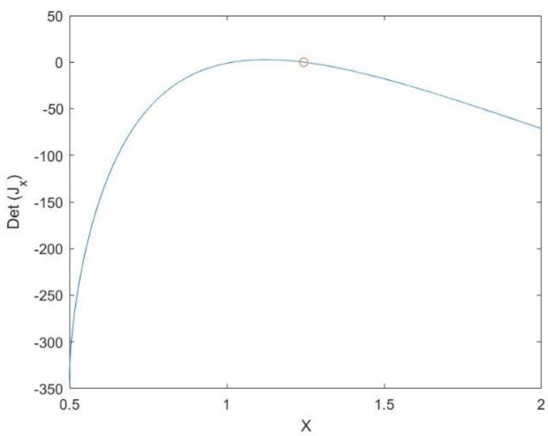
$$\begin{bmatrix} -0.5946 & 0.8705 & -0.6334 \\ -0.1619 & -0.2273 & 0.4707 \\ -0.7875 & 0.4366 & -0.6142 \end{bmatrix}$$

The Eigenvector that corresponds to Eigenvalue of 0:

$$\begin{bmatrix} 0.8705 \\ -0.2273 \\ 0.4366 \end{bmatrix}$$



1]

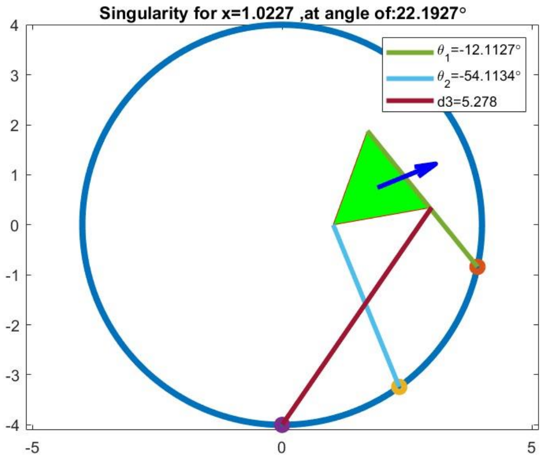


The Eigenvectors matrix:

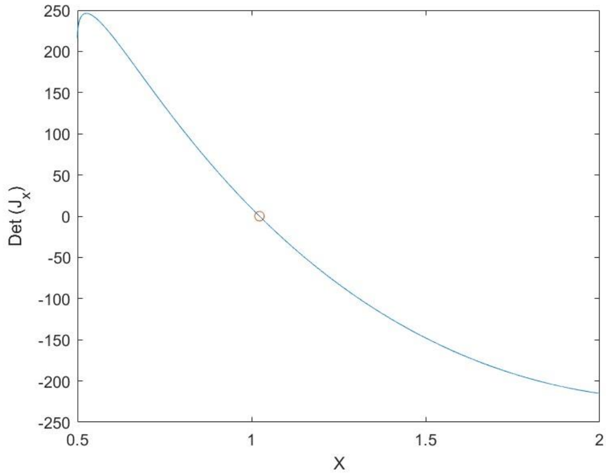
$$\begin{bmatrix} 0.5887 & 0.9130 & 0.2020 \\ 0.0581 & 0.3724 & 0.9457 \\ -0.8062 & 0.1666 & -0.2547 \end{bmatrix}$$

The Eigenvector that corresponds to Eigenvalue of 0:

$$\begin{bmatrix} 0.9130 \\ 0.3724 \\ 0.1666 \end{bmatrix}$$



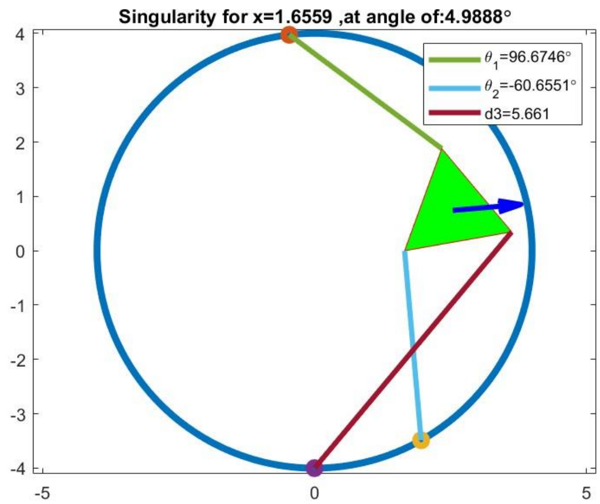
1]

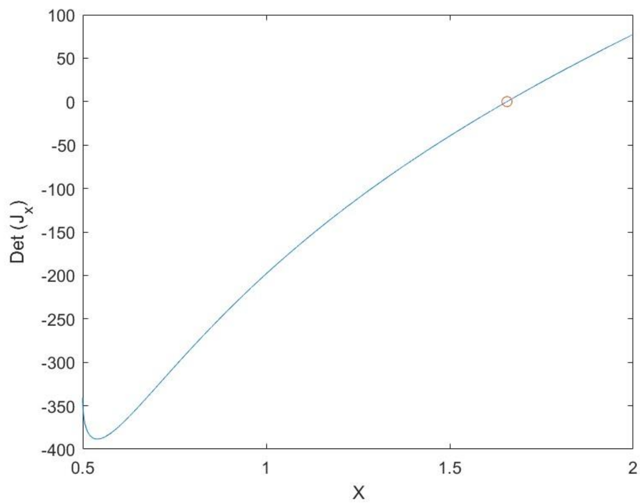


$$\begin{bmatrix} -0.6241 & 0.9282 & -0.0153 \\ -0.0208 & 0.0810 & 0.9549 \\ -0.7810 & 0.3631 & -0.2966 \end{bmatrix}$$

The Eigenvector that corresponds to Eigenvalue of 0:

$$\begin{bmatrix} 0.9282 \\ 0.0810 \\ 0.3631 \end{bmatrix}$$





- `Jacobian_x`– Function that calculates the Jacobian Jx according to the tool position X and the actuators position Q.
- `Draw_Singular_state`- Function that draws the current system state by its tool and joint position and also an arrow from the center of the plate that pointing the direction of the free motion.

6. Summary

In this homework exercise, we experimented with the analysis of parallel robots. We examined the direct and inverse kinematics of the robot, computed the corresponding Jacobians for the joint and tool spaces, explored path planning for the robot, and analyzed its singular points. Unlike serial robots, parallel robots have geometric constraints that affect their calculations and analysis. The inverse kinematics in parallel robots are generally simpler and can be analytically derived based on the geometric constraints that define the robot. On the other hand, the Forward kinematics in parallel robots are often more complex, and sometimes they can only be represented by implicit equations.

Additionally, we discovered that the Jacobian in parallel robots can be divided into two parts: one related to the joint space and the other related to the tool space. For each Jacobian, we can find singular points, where each type represents a different form of singularity. J_x allows us to find singular points where the robot gains degrees of freedom, meaning that for certain configurations of the joints, the tool's position is unattainable. J_q enables us to find singular points where the robot loses degrees of freedom, similar to the singularities we encountered in serial robots.

In general, parallel robots offer several advantages. These include higher precision, which allows for more accurate positioning and control, and a high load-to-weight ratio, meaning they can handle heavier loads relative to their own weight. However, despite these advantages, analyzing and controlling parallel robots can still be more challenging compared to serial robots, and each unique geometric system may require a slightly different analysis method, potentially involving numerical iterations. Therefore, if the requirements are met, utilizing a serial robot may be a simpler option.