Project Report

Kinematics of the Stewart Platform

Team Members:

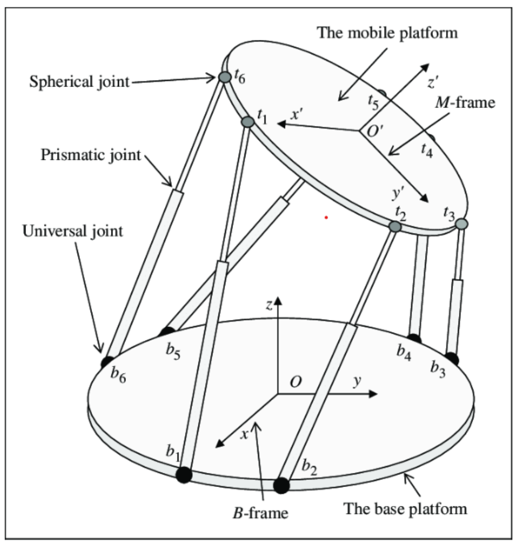
- **Ankit Kumar Aggarwal**
- **Avinash Rajaraman Swaminathan**
- **Sayed Raheel Hussain**

Contents

Stewart Platform

A Stewart platform consists of six variable length struts, or prismatic joints, supporting a payload. Prismatic joints operate by changing the length of the strut, usually pneumatically or hydraulically. As a six-degree-of-freedom robot, the Stewart platform can be placed at any point and inclination in three-dimensional space that is within its reach.

Stewart platforms are known by various other names. In many applications, including in flight simulators, it is commonly referred to as a motion base. It is sometimes called a six-axis platform or 6-DoF platform because of its possible motions and, because the motions are produced by a combination of movements of multiple actuators, it may be referred to as a synergistic motion platform, due to the synergy (mutual interaction) between the way that the actuators are programmed. Because the device has six actuators, it is often called a hexapod (six legs) in common usage.



An example of six-DOF Stewart platform. DOF: degree of freedom.

Project Problem Statement

To simplify matters, the project concerns a two-dimensional version of the Stewart platform. It will model a manipulator composed of a triangular platform in a fixed plane controlled by three struts, as shown in Figure 1.14. The inner triangle represents

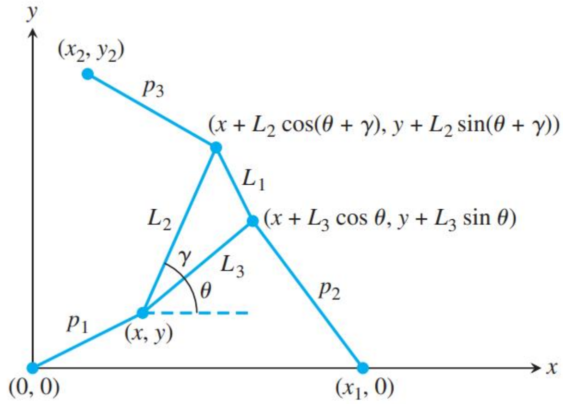


Figure 1.14 Schematic of planar Stewart platform. The forward kinematics problem is to use the lengths p_1, p_2, p_3 to determine the unknowns x, y, θ.

the planar Stewart platform whose dimensions are defined by the three lengths L1, L2, and L3. Let γ denote the angle across from side L1. The position of the platform is controlled by the three numbers p1, p2, and p3, the variable lengths of the three struts.

Finding the position of the platform, given the three strut lengths, is called the forward, or direct, kinematics problem for this manipulator. Namely, the problem is to compute (x, y) and θ for each given p1, p2, p3. Since there are three degrees of freedom, it is natural to expect three numbers to specify the position. For motion planning, it is important to solve this problem as fast as possible, often in real time. Unfortunately, no closed-form solution of the planar Stewart platform forward kinematics problem is known.

The best current methods involve reducing the geometry of Figure 1.14 to a single equation and solving it by using one of the solvers explained in this chapter. Your job is to complete the derivation of this equation and write code to carry out its solution. Simple trigonometry applied to Figure 1.14 implies the following three equations:

$$
\begin{aligned}
p_1^2 &= x^2 + y^2 \\
p_2^2 &= (x + A_2)^2 + (y + B_2)^2 \\
p_3^2 &= (x + A_3)^2 + (y + B_3)^2. \qquad (1.38)
\end{aligned}
$$

In these equations,

$$A_2 = L_3 \cos\theta - x_1$$
$$B_2 = L_3 \sin\theta$$
$$A_3 = L_2 \cos(\theta + \gamma) - x_2 = L_2[\cos\theta \cos\gamma - \sin\theta \sin\gamma] - x_2$$
$$B_3 = L_2 \sin(\theta + \gamma) - y_2 = L_2[\cos\theta \sin\gamma + \sin\theta \cos\gamma] - y_2.$$

Note that (1.38) solves the inverse kinematics problem of the planar Stewart platform, which is to find p1, p2, p3, given x, y, θ. Your goal is to solve the forward problem, namely, to find x, y, θ, given p1, p2, p3.

Multiplying out the last two equations of (1.38) and using the first yields

$$p_2^2 = x^2 + y^2 + 2A_2x + 2B_2y + A_2^2 + B_2^2 = p_1^2 + 2A_2x + 2B_2y + A_2^2 + B_2^2$$
$$p_3^2 = x^2 + y^2 + 2A_3x + 2B_3y + A_3^2 + B_3^2 = p_1^2 + 2A_3x + 2B_3y + A_3^2 + B_3^2,$$

which can be solved for x and y as

$$x = \frac{N_1}{D} = \frac{B_3(p_2^2 - p_1^2 - A_2^2 - B_2^2) - B_2(p_3^2 - p_1^2 - A_3^2 - B_3^2)}{2(A_2B_3 - B_2A_3)}$$
$$y = \frac{N_2}{D} = \frac{-A_3(p_2^2 - p_1^2 - A_2^2 - B_2^2) + A_2(p_3^2 - p_1^2 - A_3^2 - B_3^2)}{2(A_2B_3 - B_2A_3)}, \qquad (1.39)$$

as long as $D = 2(A_2B_3 - B_2A_3) \neq 0$.

Substituting these expressions for x and y into the first equation of (1.38), and multiplying through by D2, yields one equation, namely,

$$f = N_1^2 + N_2^2 - p_1^2 D^2 = 0 \qquad (1.40)$$

in the single unknown θ. (Recall that p1, p2, p3, L1, L2, L3,γ, x1, x2, y2 are known.) If the roots of f (θ) can be found, the corresponding x- and y- values follow immediately from (1.39).

Note that f (θ) is a polynomial in sinθ and cos θ, so, given any root θ, there are other roots θ + 2πk that are equivalent for the platform. For that reason, we can restrict attention to θ in [−π,π]. It can be shown that f (θ) has at most six roots in that interval.

Tools and Technology

- Python
- Jupiter Notebook

Activity 1:

Write a function file for f (θ). The parameters L1, L2, L3, γ, x1, x2, y2 are fixed constants, and the strut lengths p1, p2, p3 will be known for a given pose. Here, for free, are the first and last lines:

function out=f(theta)
:
:
out=N1^2+N2^2-p1^2*D^2;
To test your code, set the parameters L1 = 2, L2 = L3 = √ 2, γ = π/2, p1 = p2 = p3 = √ 5 from Figure 1.15. Then, substituting θ = −π/4 or θ = π/4, corresponding to Figures 1.15(a, b), respectively, should make f (θ) = 0.

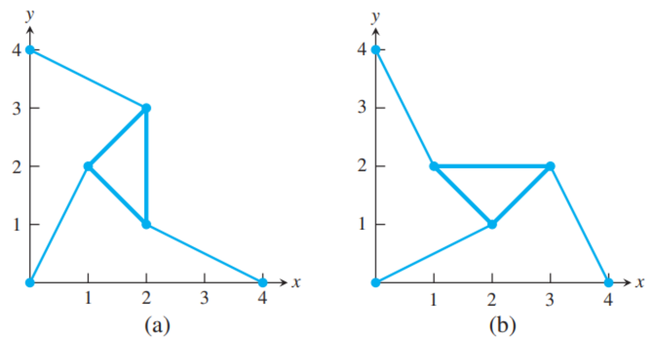


Figure 1.15 Two poses of the planar Stewart platform with identical arm lengths. Each pose corresponds to a solution of (1.38) with strut lengths $p_1 = p_2 = p_3 = \sqrt{5}$. The shape of the triangle is defined by $L_1 = 2$, $L_2 = L_3 = \sqrt{2}$, $\gamma = \pi/2$.

Solution: In order to test this, first we created a function "**stewart_theta**", which will accept the value of "**theta**" as an input argument and return an output "**out**" which is our function equation. This function will basically calculate the value of different parameters of the function equation i.e., N1, N2 and D which are indirectly based on other constants like L1, L2, L3, gamma, p1, p2, p3, A2, B2, A3, B3 etc. of Stewart platform.

Now, in order for this this function to work properly, it should return approx. zero once we substitute the given values of **"theta"** i.e., pi/4 and -pi/4.

Here are the results of substitution which indicates that the **"out"** for the function is very less approx. to zero for the provided values of **"theta"**. It indicates that the function is working properly and returning expected results.

```
#Substitute theta = "pi/4" and execute the function
stewart_theta(np.pi/4)
```

```
(2.0, 1.0, -4.547473508864641e-13)
```

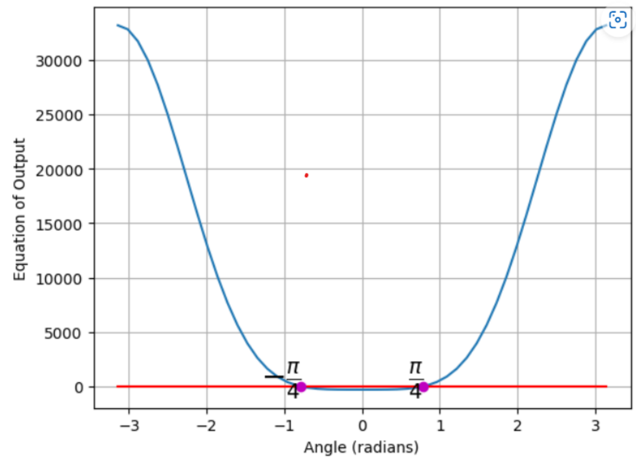
```
#Substitute theta = "-pi/4" and execute the function
stewart_theta(-np.pi/4)
```

```
(1.0, 2.0, -4.547473508864641e-13)
```

Activity 2:

Plot f (θ) on [−π, π]. You may use the @ symbol as described in Appendix B.5 to assign a function handle to your function file in the plotting command. You may also need to precede arithmetic operations with the "." character to vectorize the operations, as explained in Appendix B.2. As a check of your work, there should be roots at ± π/4.

Solution: Please find below image **2(a)** to refer the plots of f (θ).



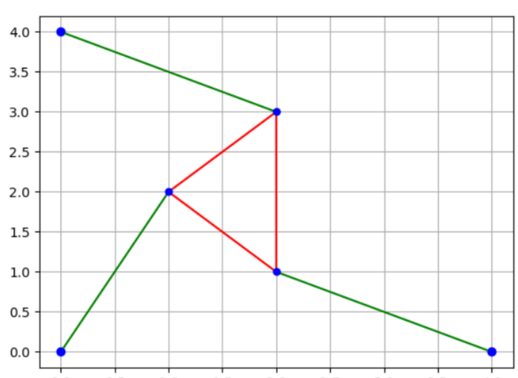
2(a)

In order to verify the roots, we make use of the "**stewart_theta**" function defined in Activity 1 to calculate and plot the graph of different output values with the specified x-axis limits i.e., [-π. π]. The image above clearly shows that the function is approximately equal to zero at [-π/4, π/4]. Hence, we can conclude these values as the roots of the given function.

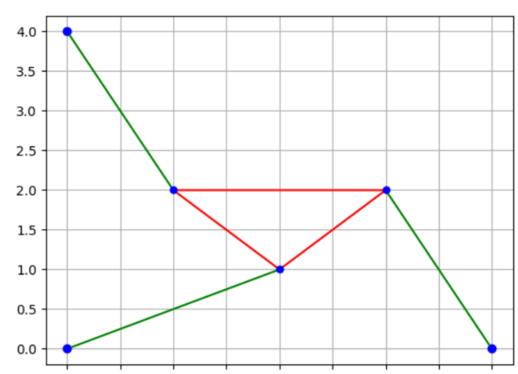
Activity 3:

Reproduce Figure 1.15. Plot a red triangle with vertices (u1, v1), (u2, v2), (u3, v3) and place small circles at the strut anchor points (0, 0), (x1, 0), (x2, y2). In addition, draw the struts.

Solution: For this we referred Figure 1.15, to identify the coordinates of the platform and used them inside the modified Stewart_theta function to reproduce identical output images 3(a) and 3(b) below.



3(a)



3(b)

Activity 4:

Solve the forward kinematics problem for the planar Stewart platform specified by x1 = 5, (x2, y2) = (0,6), L1 = L3 = 3, L2 = 3 √ 2, γ = π/4, p1 = p2 = 5, p3 = 3. Begin by plotting f (θ). Use an equation solver to find all four poses, and plot them. Check your answers by verifying that p1, p2, p3 are the lengths of the struts in your plot.

Solution:

Which equation solver did you use, and why?

we use bisection method to resolve kinematics of Stewart platform as this method guarantees to converge to the roots irrespective of the initial guess. Secondly, bisection method is comparatively simpler and more robust as compared to other equation solving methods. Additionally, from the problem statement, we used the provided intervals [-**π**, **π**] to plot a graph between input and output of the function through which we were able to successfully identified the four roots and their approximate intervals of the root through graph.

Secondly, when we compare the findings between Newton's and bisection method, we found that even though the roots from Newton's method are comparatively closer as compared to the roots from bisection method but the relative error in Newton's method is higher w.r.t. to the Bisection method.

Finally, we did not prefer fixed point Iteration (FPI) method because the function consists of multiple trigonometric functions which make is quite complex to get original Stewart platform equation in form of **f(θ) = θ**, as required by fixed point Iteration. Additionally, due to the complexity of the equation, solving **f(θ) = θ** would be more resource intensive and time-consuming process.

How did you initialize your solver, and why?

To apply the bisection method, it is necessary to first establish an interval [a, b] in which the function f(a) and f(b) have opposite signs, indicating that 'f(a) * f(b) < 0'. This condition is required because the intermediate value theorem guarantees that a root of the function exists within that interval. Bolzano's theorem was used to plot the range of the function "forward_kinematics(theta)" along with its input and output ranges from [-π, π], allowing us to identify the approximate locations of the roots on the graph. We then utilized this information to set the initial interval for the bisection method and chose [-π, π] as the interval for plotting the function.

What stopping condition did you use in your solver? How accurate is the obtained solution (root)?

For the bisection method, we have adopted a stopping condition of Tolerance (TOL)<=10^-12. This means that we want the size of the interval to be less than or equal to 10^-12, and we have also set a maximum iteration limit of 1000. For functions that are relatively simple and quickly converge, a very tight tolerance value (e.g., Tol < 10^-12) may suffice as a stopping condition. However, since the "forward_kinematics" function involves complex trigonometric expressions, it would be prudent to set a maximum number of iterations in addition to the tolerance value.

To determine the accuracy of the solution, we evaluated the relative error along with the accuracy of the function at the estimated roots and within the defined tolerance level (i.e., Tol < 10^-12). Here is the relative error and accuracy for each of the roots.

Relative Error of Bisection Method

```
Root -0.3310051842827705 has a relative error of 1.178 %
Root 1.1436855178208596 has a relative error of 1.289 %
Root 2.115909014084604 has a relative error of 0.459 %
```

Relative Error of Newton's Method

```
Root 2.115909014086458 has a relative error of 2.969 %
Root -6.614190491463457 has a relative error of 1.32 %
Root 5.562336102719197 has a relative error of 2.189 %
```

Accuracy

	Root	Accuracy
0	-0.720849	3.731959e-12
1	-0.331005	1.865952e-12
2	1.143686	9.330314e-13
3	2.115909	9.330314e-13

How fast is your solver?

To determine the speed or the iteration of the solution, we evaluated the number of iterations performed by bisection method in order to estimate the roots within the defined tolerance level (i.e., Tol < 10^-12). Here are the number of iterations for each of the roots.

	Root	Iterations
0	-0.720849	20
1	-0.331005	20
2	1.143686	19
3	2.115909	13

Empirically check that your solver achieves the theoretical convergence rate.

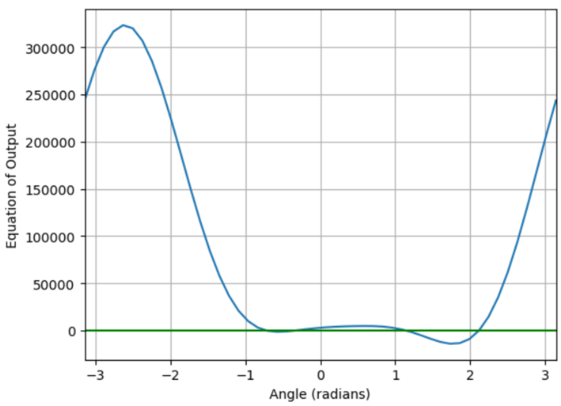
The convergence rate calculation in my code assumes that the bisection method has been run to convergence.

In order to calculate the convergence rate, we calculate the error for each iteration using the absolute value of the difference between the approximate root at each iteration and the true root (where true root is the previous root).

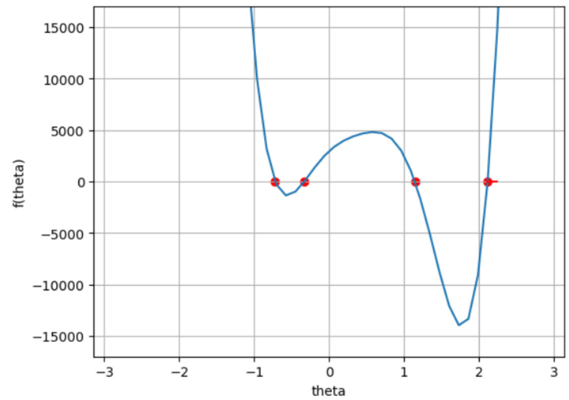
The resulting convergence rate variable should be close to 1 if the bisection method has linear convergence. As we can see the convergence is equal to 1, which signifies that bisection is converging linearly.

	Root	Convergence Rate
0	-0.720849	1.000000
1	-0.331005	1.000000
2	1.143686	0.999914
3	2.115909	1.000000

Plotted Roots



4(a)

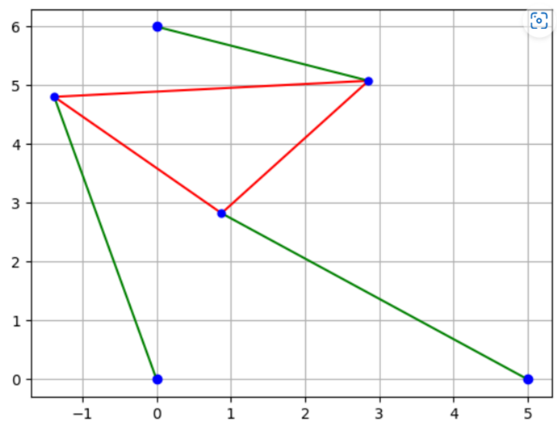


4(b)

Table 1: Containing values of θ, x, y, out.

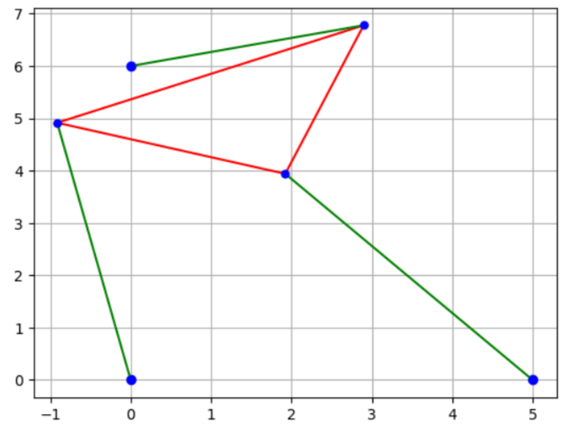
θ	X	Y	Out
−0.7208486822970133	-1.3783796529728198	4.806252733212699	-0.0097575363543001
-0.33100527019002235	-0.9147089624578358	4.915618601429166	-0.0008209647585317725
1.1436863776005795	4.48174823932921	2.2167316910137442	-0.018179710607000743
2.1159093133838343	4.571830313270758	2.0244434405504936	0.02740930297295563

θ = −0.7208486822970133



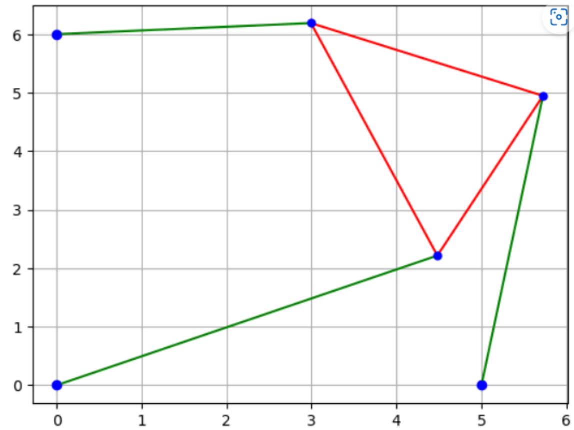
4(c)

θ = −0.33100527019002235



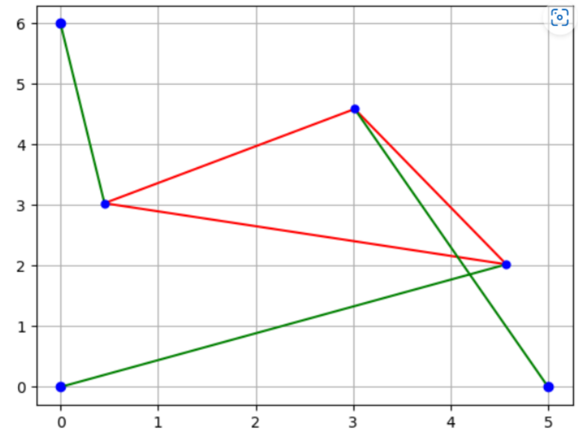
4(d)

θ = 1.1436863776005795



4(e)

θ = 2.1159093133838343

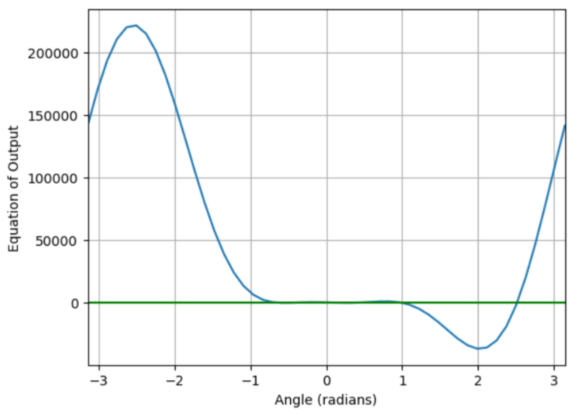


4(f)

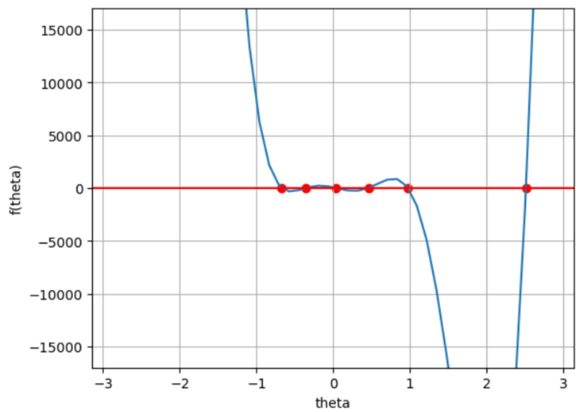
Activity 5:

Change strut length to p2 = 7 and re-solve the problem. For these parameters, there are six poses.

Solution: Once we change the length of p2=7, we are able to identify that the Stewart platform now has six roots i.e., six different poses. Now using bisection, we found the (θ), which further helps us to identify the different positions of the Stewart platform on strut length.



5(a)

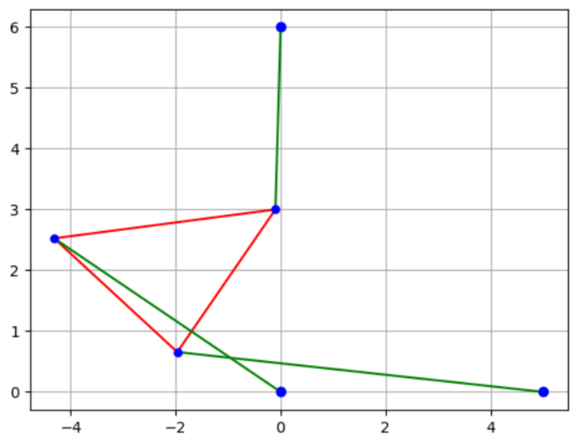


5(b)

Table 2: Containing values of θ, x, y, out.

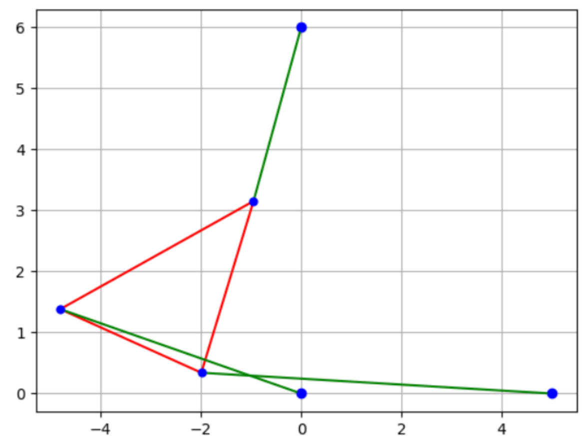
θ	X	Y	Out
-0.673157486370657	-4.314759599570059	2.5264302083997703	-6.344635039567947e-09
-0.3547402704165651	-4.804896519073602	1.383101384928693	-1.9063008949160576e-09
0.03776676057600378	-4.949024616818798	0.7121483989452148	-1.5688783605583012e-10
0.4588781810485133	-0.8198001690738083	4.932334911859019	-1.326725396211259e-09
0.9776728949997805	2.303554099145971	4.43775151538828	6.108166417106986e-09
2.5138527993513557	3.2156960361507183	3.8287464009903434	1.5471596270799637e-07

θ = -0.673157486370657



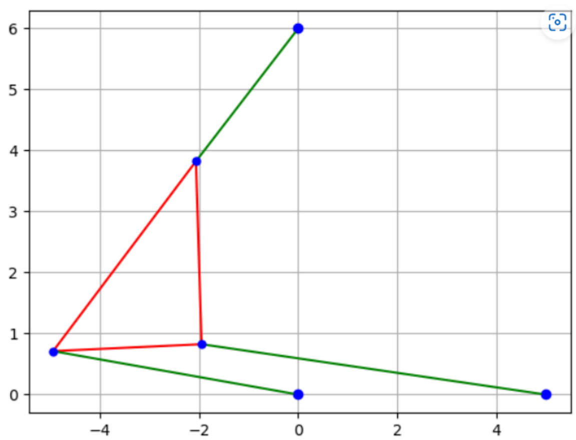
5(c)

θ = -0.3547402704165651



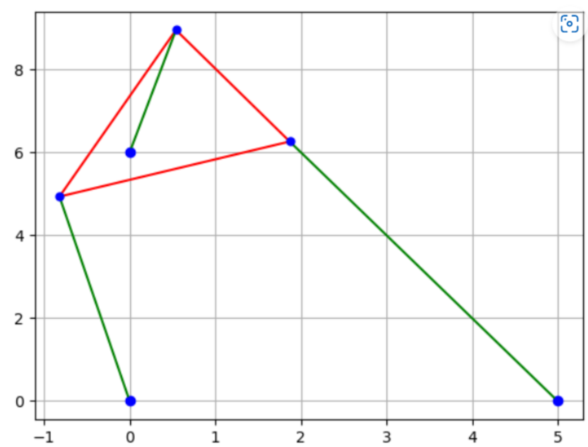
5(d)

θ = 0.03776676057600378



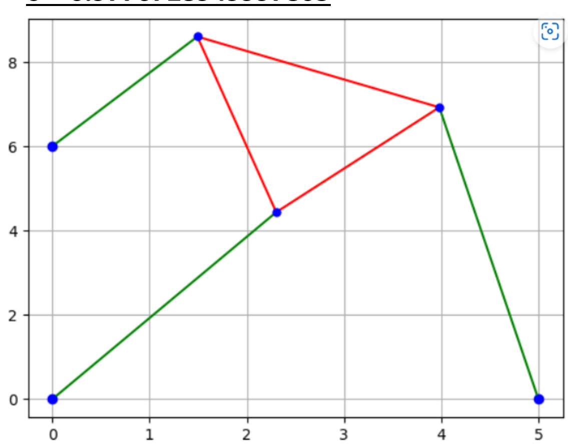
5(e)

θ = 0.4588781810485133



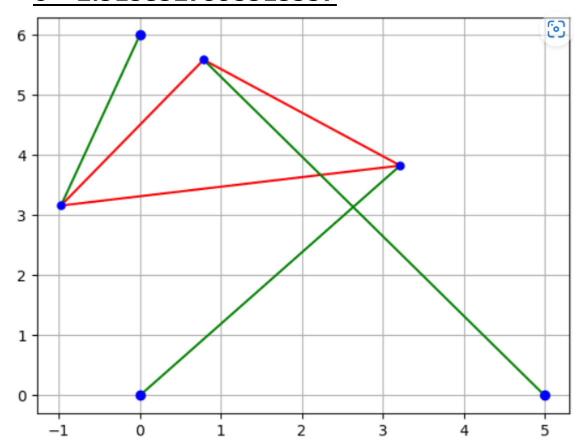
5(f)

θ = 0.9776728949997805



5(g)

θ = 2.5138527993513557



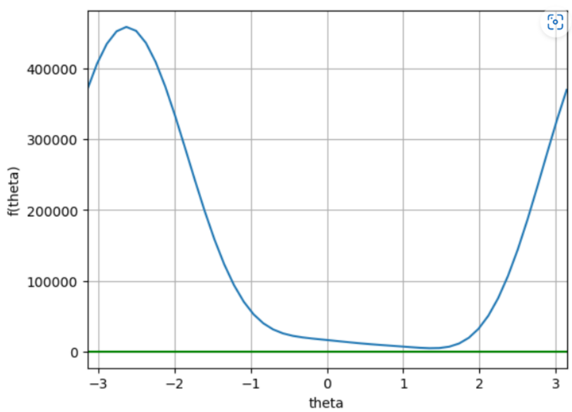
5(h)

Activity 6:

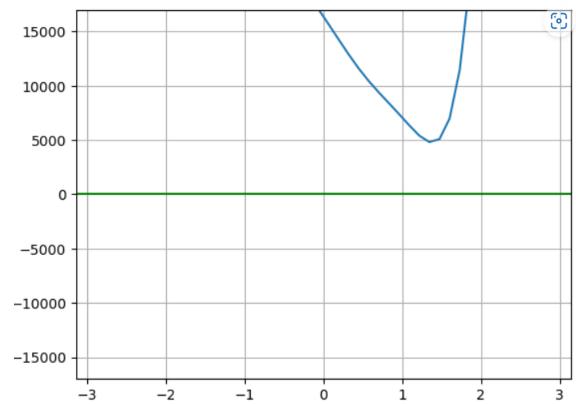
Find a strut length p2, with the rest of the parameters as in Step 4, for which there are only two poses.

Solution: In order to find a strut length p2, with only two poses, for given parameters in Step 4, we tried to calculate the no. of roots for different values of p2 i.e., p2=3 and p2=4. We found that for p2=3 there are no roots (refer image 6(a) and 6(b)). However, as soon as we change p2=4, it returns two roots (refer image 6(c) and 6(d) below).

For p2=3: No Roots

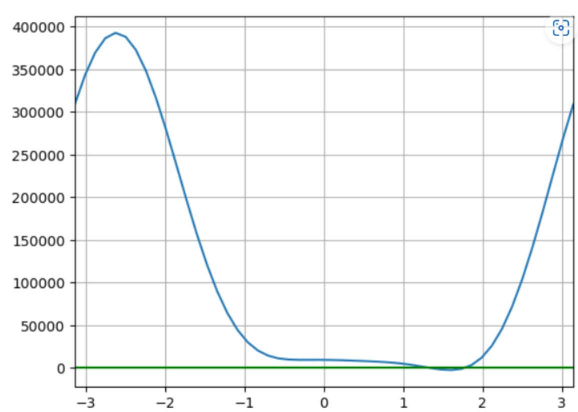


6(a)

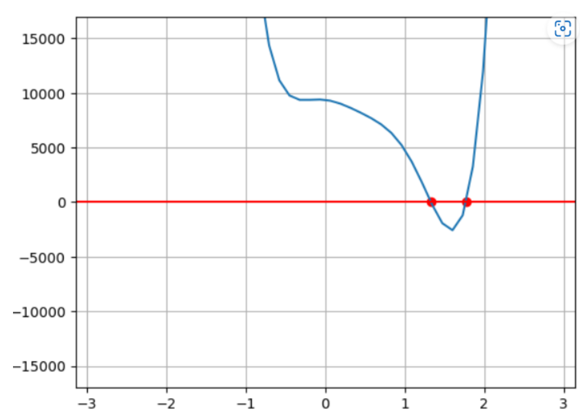


6(b)

For p2=4: Two Roots



6(c)

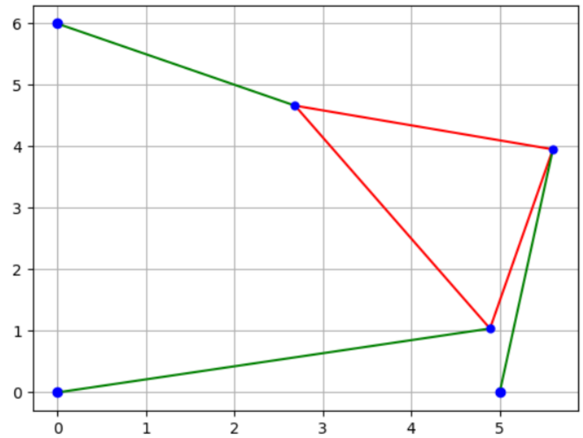


6(d)

Table 2: Containing values of θ, x, y, out.

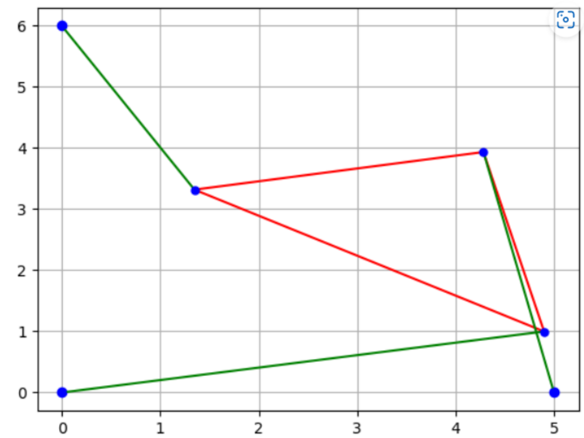
θ	X	Y	Out
1.3316422033434971	4.890658973004355	1.039930194658232	-1.1124939192086458e-08
1.7775135743995185	4.899151197793317	0.9991584164473836	-1.0317307896912098e-08

θ = 1.3316422033434971



6(e)

θ = 1.7775135743995185



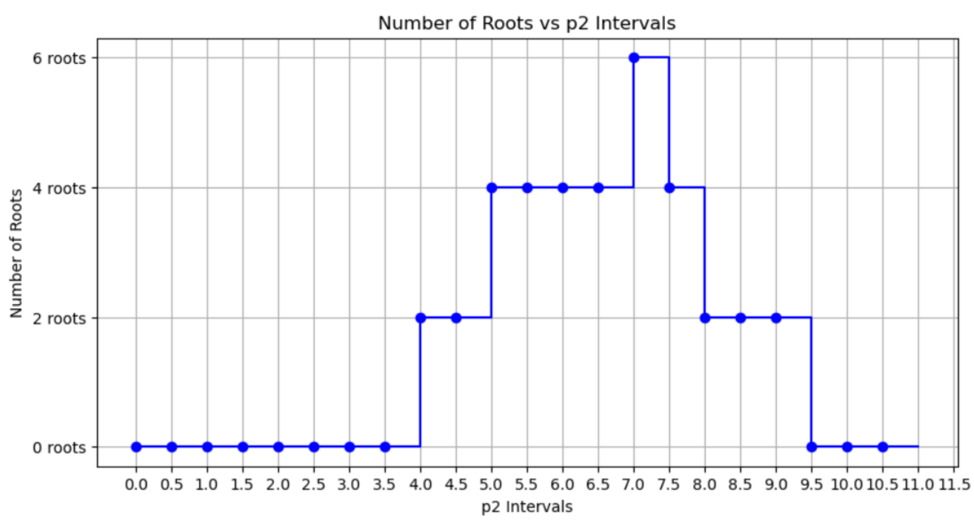
6(f)

Activity 7:

Calculate the intervals in p2, with the rest of the parameters as in Step 4, for which there are 0,2,4, and 6 poses, respectively.

Solution: Here are the approximate intervals of p2 for which there are 0,2,4 and 6 poses (i.e., # of roots) are available. Also find below the plot of p2 intervals vs number of roots which does verify the identified interval values.

Number of roots	Start_Interval	End_Interval
0	0.0000	3.5000
0	9.5000	10.5000
2	4.0000	4.5000
2	8.0000	9.0000
4	5.0000	6.5000
4	7.5000	7.5000
6	7.0000	7.0000



7(a)

Conclusion

As a part of this project concerns a two-dimensional version of the Stewart platform, that composed of a triangular platform in a fixed plane controlled by three struts. In this we have successfully located the position of the platform, given the three strut lengths i.e., the forward kinematics of the platform. In this, we have calculated the impact on the position of the platform by changing the length of strut.

To achieve this,

- The function has been created based on the Stewart platform that returns approximately zero when the input angle "theta" is the root, indicating a valid platform position for given parameters.
- Afterwards, we generated a plot that shows the relationship between the input and output of the function, using the initial interval i.e., [-pi to pi] from Activity 2. This allowed us to determine the approximate roots within the specified intervals.
- After determining the interval and number of roots within it, we proceeded to apply the bisection method to estimate the approximate locations of these roots.
- Following that, we utilized the roots to determine the coordinates of the platform, and then plotted the platform using the given strut length and other parameters.
- Finally, we examined how varying the strut length affects the number of possible poses of the Stewart platform.

Citations

- Stewart Platform Image:
    - https://www.researchgate.net/figure/An-example-of-six-DOF-Stewart-platform-DOF-degree-of-freedom_fig1_258177248
- What is Stewart Platform
    - Textbook: Timothy-Sauer-Numerical-Analysis-Pearson-2017 (3rd Edition)
    - https://en.wikipedia.org/wiki/Stewart_platform
- BI-Section Function Code
    - Textbook: Timothy-Sauer-Numerical-Analysis-Pearson-2017 (3rd Edition)
- Code References (with respect to Syntax's and library functions)
    - Google Search
    - Python-for-Data-Analysis-2nd-Edition.pdf (AIM-5001 Course Textbook)
- Convergence of Bisection Method
    - https://math.stackexchange.com/questions/2513852/finding-convergence-rate-for-bisection-newton-secant-methods