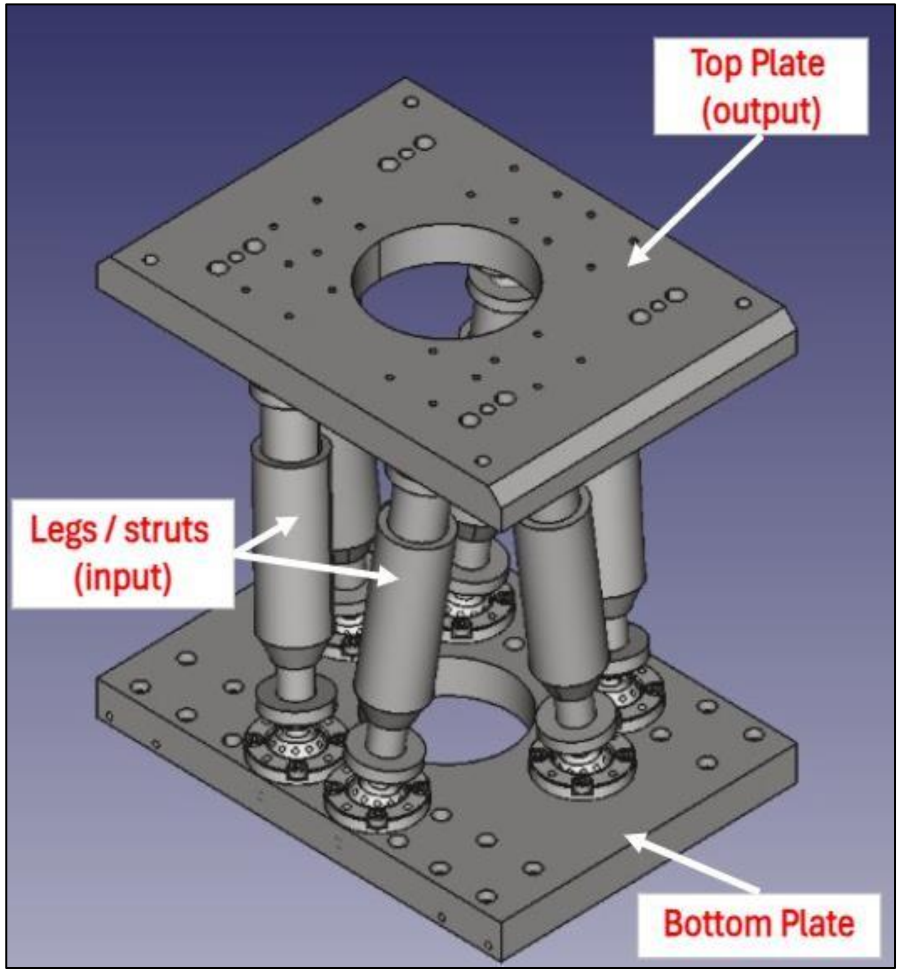
STEWART PLATFORM

by

KODANDA CHALLA

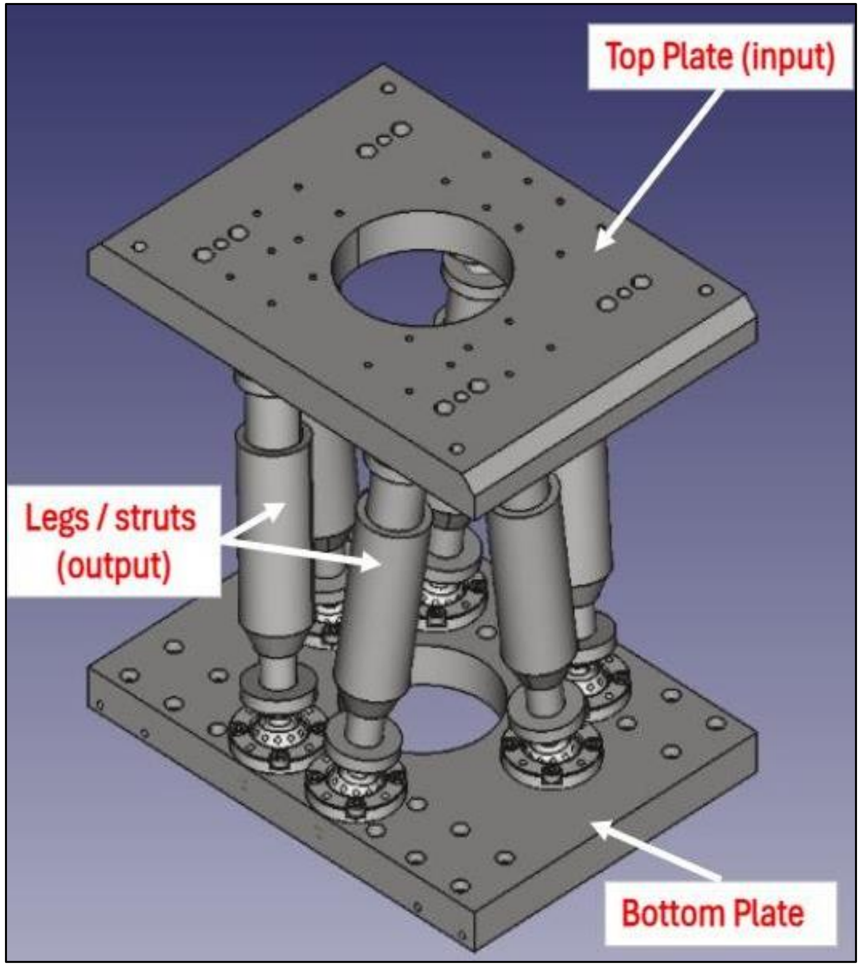
Forward Kinematics

- **Forward kinematics** involves determining the position and orientation of the top platform given the lengths of the six struts. This means calculating where the top platform is and how it is oriented in 3D space based on the known strut lengths.
- Steps:
  - **Input:** Known lengths of the six struts.
  - **Process:** Use geometric and trigonometric relationships to compute the position (x, y, z) and orientation (roll, pitch, yaw) of the top platform.
  - **Output:** The position and orientation of the top platform.
  - Forward kinematics for a Stewart platform is generally complex due to the non-linear relationships between the strut lengths and the platform's pose.



Inverse kinematics

- **Inverse kinematics** involves determining the required lengths of the six struts to achieve a desired position and orientation of the top platform. This means calculating how each strut should extend or retract to move the top platform to a specific pose.
- Steps:
  - **Input:** Desired position (x, y, z) and orientation (roll, pitch, yaw) of the top platform.
  - **Process:** Use the geometric and trigonometric relationships to compute the lengths of the struts.
  - **Output:** The lengths of the six struts.
  - Inverse kinematics is typically easier to solve than forward kinematics for a
  - Stewart platform because it involves direct geometric relationships.



Summary:

Forward Kinematics:

Strut lengths -> Position and Orientation of Top Plate

Inverse Kinematics:

Position and Orientation of Top Plate -> Strut lengths

https://github.com/kodandachalla/Hexapod_Inverse_Kinematics

https://github.com/kodandachalla/Hexapod_Forward_Kinematics

Comparison Forward vs Inverse results

Inverse Kinematics of Stewart Platform

Input:	desired_position (mm) = [X; Y; Z] 3x1 matrix desired_orientation (deg) = [rotX; rotY; totZ] 3x1 matrix
output:	6 Leg lengths in mm (leg_lengths) 1x6 Matrix

```
% Define the desired position and orientation of the moving platform
desired_position = [-2.913;   10.848;  182.83975]; % Desired position [x, y, z]
desired_orientation = [-0.5034;    1.371;    3.179]*pi/180; % Desired orientatic
```

```
Leg Lengths are:
  180.7695

  184.5884

  185.4980

  184.7730

  185.0452

  184.5029
```

Forward Kinematics of the Stewart Platform

Input:	6 Leg lengths in mm (leg_lengths) 1x6 Matrix
output:	desired_position (mm) = [X; Y; Z] 3x1 matrix desired_orientation (deg) = [rotX; rotY; totZ] 3x1 matrix

leg_lengths = [180.769486760529 184.588371003279 185.497976872750 184.773016742414 185.045189780015 184.502949692505];

```
Position and orientation of the platform:
x: -2.9130
y: 10.8480
z: 182.8398
roll: -0.5034
pitch: 1.3710
yaw: 3.1790
```

Calibration of Platforms using CMM*

Calibration of individual plates are necessary to locate real universal joint points for

- Fixed Platform
- Moving Platform

Leg center points are Recorded

Note: calibration points numbering is different from assembly points numbering

*CMM (Coordinate Measuring machine)

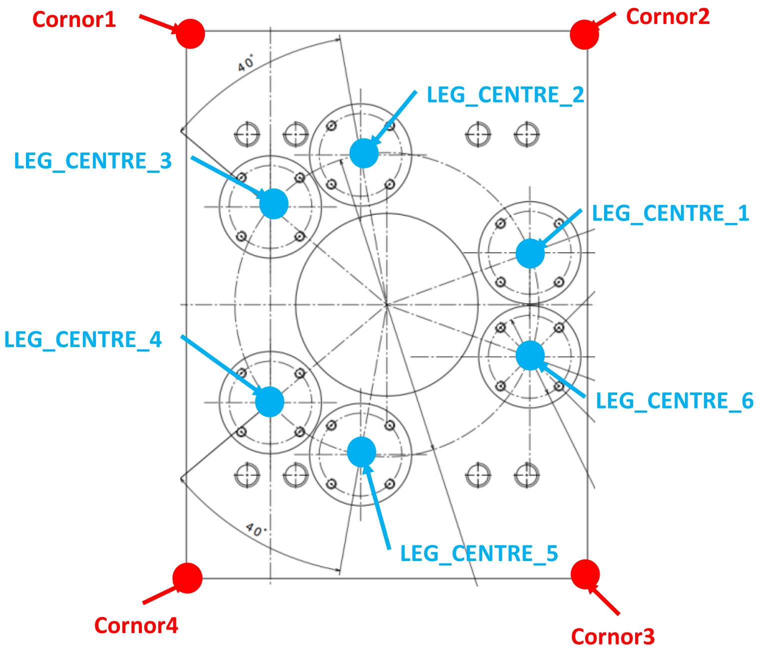
Fixed Platform:

CORNOR1	THEO/<-82.5,112.5,0>,<-1,0,0>	ACTL/<-82.468,112.527,0.002>,<-0.9999997,0.0000663,0.000812>
CORNOR2	THEO/<82.5,112.5,0>,<0,0,1>	ACTL/<82.493,112.516,-0.132>,<-0.0007324,0.0030107,0.9999952>
CORNOR3	THEO/<82.5,-112.5,0>,<0,0,1>	ACTL/<82.447,-112.499,0.568>,<-0.000733,0.0000002,0.9999997>
CORNOR4	THEO/<-82.5,-112.5,0>,<0,0,-1>	ACTL/<-82.468,-112.465,0.702>,<0,0,-1>
LEG_CENTRE_1	THEO/<61.08,22.231,-11>,<0,0,1>	ACTL/<61.06,22.233,-10.803>,<0.0004847,0.0034285,0.999994>
LEG_CENTRE_2	THEO/<-11.287,64.013,-11>,<0,0,1>	ACTL/<-11.3,64.009,-10.898>,<0.0009114,0.0037219,0.9999927>
LEG_CENTRE_3	THEO/<-49.793,41.781,-11>,<0,0,1>	ACTL/<-49.825,41.784,-10.779>,<0.0012924,0.0036046,0.9999927>
LEG_CENTRE_4	THEO/<-49.793,-41.781,-11>,<0,0,1>	ACTL/<-49.834,-41.765,-10.52>,<0.00109,0.003147,0.9999945>
LEG_CENTRE_5	THEO/<-11.287,-64.013,-11>,<0,0,1>	ACTL/<-11.327,-63.988,-10.504>,<0.0006666,0.0030836,0.999995>
LEG_CENTRE_6	THEO/<61.08,-22.231,-11>,<0,0,1>	ACTL/<61.058,-22.228,-10.654>,<0.0005216,0.0032783,0.9999945>
CENTRE_CIRCLE_CENTRE	THEO/<0,0,0>,<0,0,1>	ACTL/<0.015,0.025,0.285>,<0.000489,0.0030945,0.9999951>

Moving Platform:

Text Document

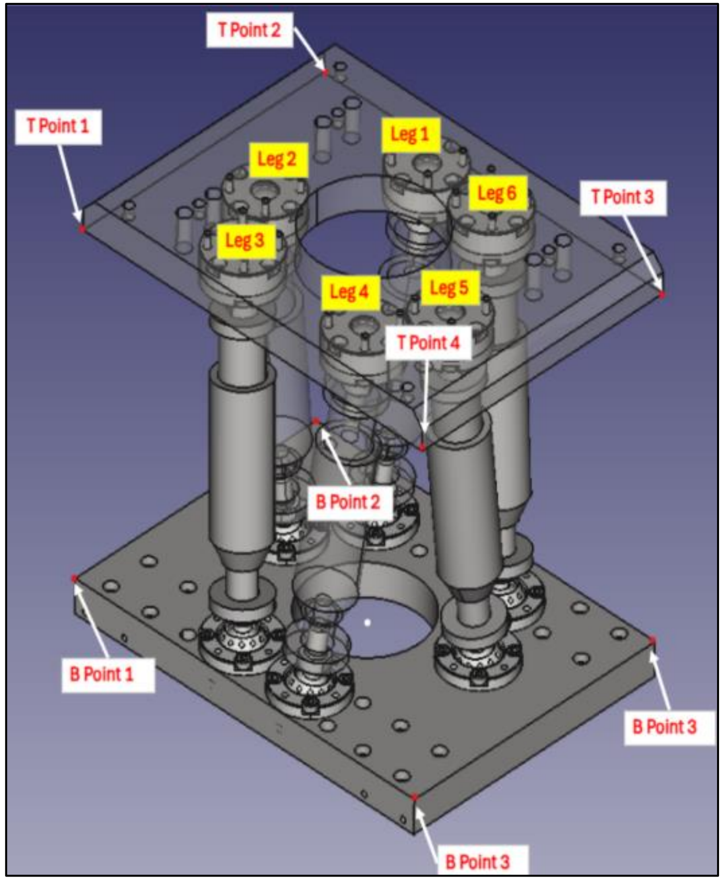
CORNOR1	THEO/<-82.5,112.5,0>,<0,0,-1>	ACTL/<-82.474,112.483,-0.029>,<0,-0.0015488,-0.9999988>
CORNOR2	THEO/<82.5,112.5,0>,<0,0,-1>	ACTL/<82.459,112.49,-0.407>,<0.0011578,-0.0015488,-0.9999981>
CORNOR3	THEO/<82.5,-112.5,0>,<0,0,-1>	ACTL/<82.466,-112.432,-0.052>,<0.0011577,0.0000006,-0.9999993>
CORNOR4	THEO/<-82.5,-112.5,0>,<0,0,-1>	ACTL/<-82.474,-112.517,0.326>,<0,0,-1>
LEG_CENTRE_1	THEO/<58.731,21.376,-11>,<0,0,1>	ACTL/<58.704,21.364,-11.243>,<0.0020898,0.0016651,0.9999964>
LEG_CENTRE_2	THEO/<-10.853,61.55,-11>,<0,0,1>	ACTL/<-10.873,61.536,-11.135>,<0.0022517,0.0015893,0.9999962>
LEG_CENTRE_3	THEO/<-47.878,40.174,-11>,<0,0,1>	ACTL/<-47.907,40.162,-11.017>,<0.0019364,0.0015196,0.999997>
LEG_CENTRE_4	THEO/<-47.878,-40.174,-11>,<0,0,1>	ACTL/<-47.912,-40.187,-10.888>,<0.0023093,0.0016937,0.9999959>
LEG_CENTRE_5	THEO/<-10.853,-61.55,-11>,<0,0,1>	ACTL/<-10.875,-61.556,-10.94>,<0.0023827,0.0017795,0.9999956>
LEG_CENTRE_6	THEO/<58.731,-21.376,-11>,<0,0,1>	ACTL/<58.708,-21.383,-11.171>,<0.0021745,0.0016568,0.9999963>
CENTRE_CIRCLE_CENTRE	THEO/<0,0,0>,<0,0,1>	ACTL/<0.013,-0.024,-0.04>,<0.0025942,0.0016802,0.9999952>



Calibration of Platforms

Moving Measured Points

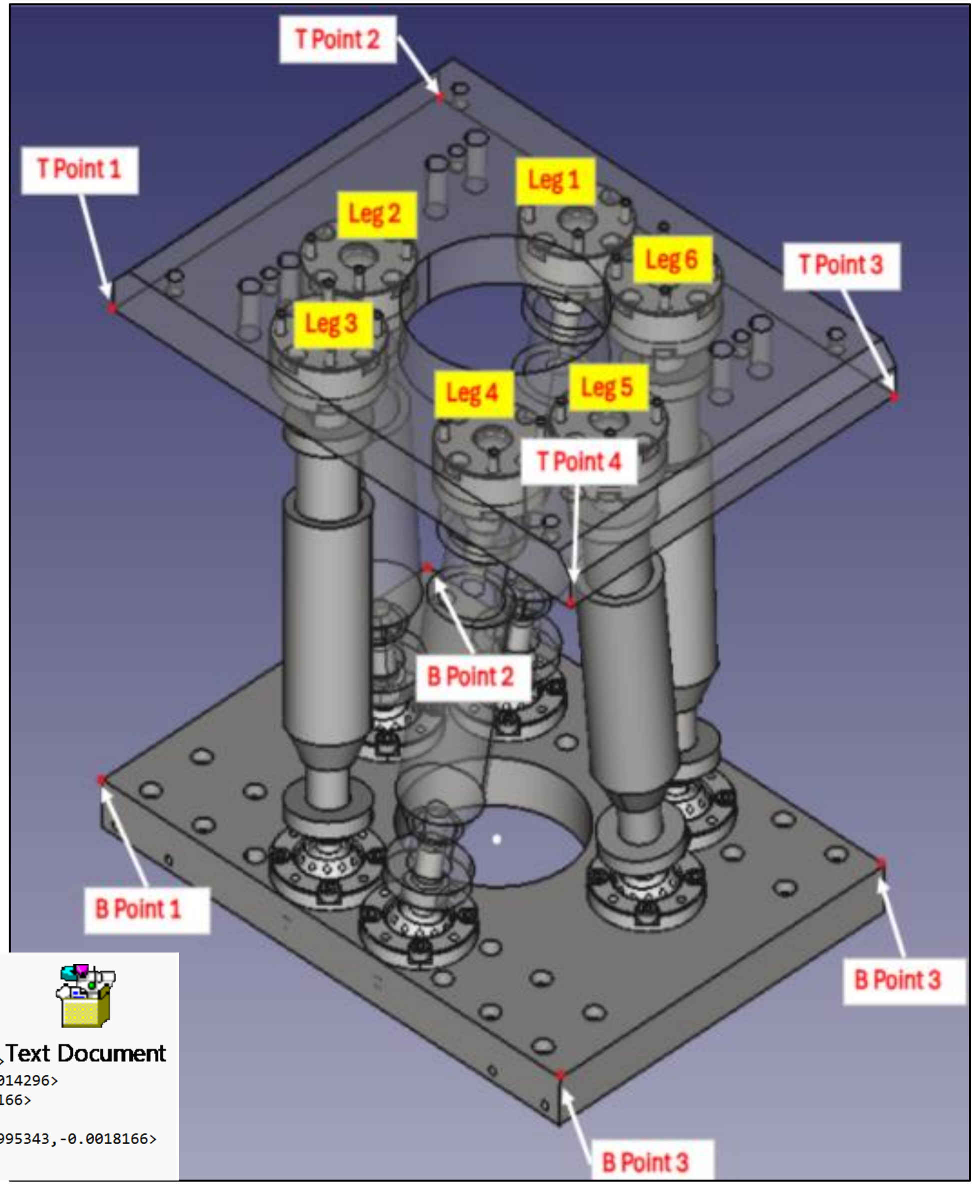
Platform Calibration points are different from assembly Points



Fixed platform		Moving platform	
leg 1	Leg_centre_4	leg 1	Leg_centre_6
leg 2	Leg_centre_5	leg 2	Leg_centre_5
leg 3	Leg_centre_6	leg 3	Leg_centre_4
leg 4	Leg_centre_1	leg 4	Leg_centre_3
leg 5	Leg_centre_2	leg 5	Leg_centre_2
leg 6	Leg_centre_3	leg 6	Leg_centre_1

Calibrated assembled Hexapod and captured following data

Kept all legs gauges at *Zero position (Case 1)**, and captured B points and T points (fig) using CMM



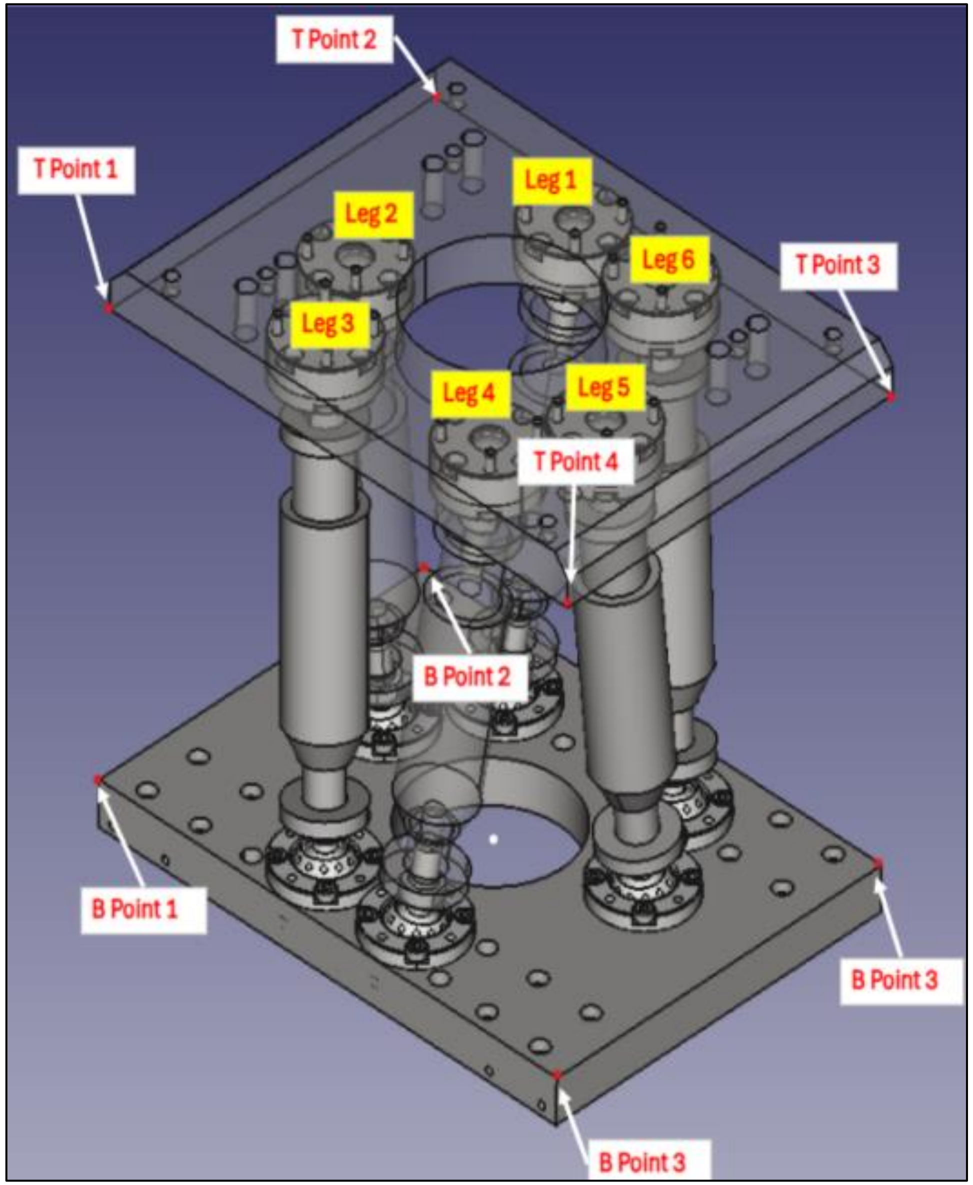

```
Stewart Platform (Hexapod): All Legs at Zero:

B1      THEO/<-82.5,112.5,0>,<1,0,0>                 ACTL/<-82.478,112.495,-0.004>,<0.9999997,0.0000846,0.0007034>
B2      THEO/<82.5,112.5,0>,<0,-1,0>                 ACTL/<82.454,112.508,0.112>,<0.0002142,-0.9999999,-0.0004765>
B3      THEO/<82.5,-112.5,0>,<0,-1,0>                ACTL/<82.502,-112.489,0.004>,<0.0002142,-0.9999999,-0.0004765>
B4      THEO/<-82.5,-112.5,0>,<-1,0,0>               ACTL/<-82.478,-112.513,-0.112>,<-0.9999997,-0.0001421,-0.0007034>
P1      THEO/<-82.5,112.5,181.195>,<-1,0,0.0000022>      ACTL/<-78.089,115.325,180.165>,<-0.9995178,0.0310181,0.0014296>
P2      THEO/<82.5,112.5,181.194>,<0,1,-0.0000003>       ACTL/<86.773,110.209,179.929>,<0.0305042,0.999533,0.0018166>
P3      THEO/<82.5,-112.5,181.195>,<0,0,-1>          ACTL/<79.912,-114.609,179.521>,<-0.0049809,0.0024965,-0.9999845>
P4      THEO/<-82.5,-112.5,181.195>,<0,-1,0.0000003>         ACTL/<-84.943,-109.583,179.756>,<-0.0304627,-0.9995343,-0.0018166>
```

Text Document

CMM Coding

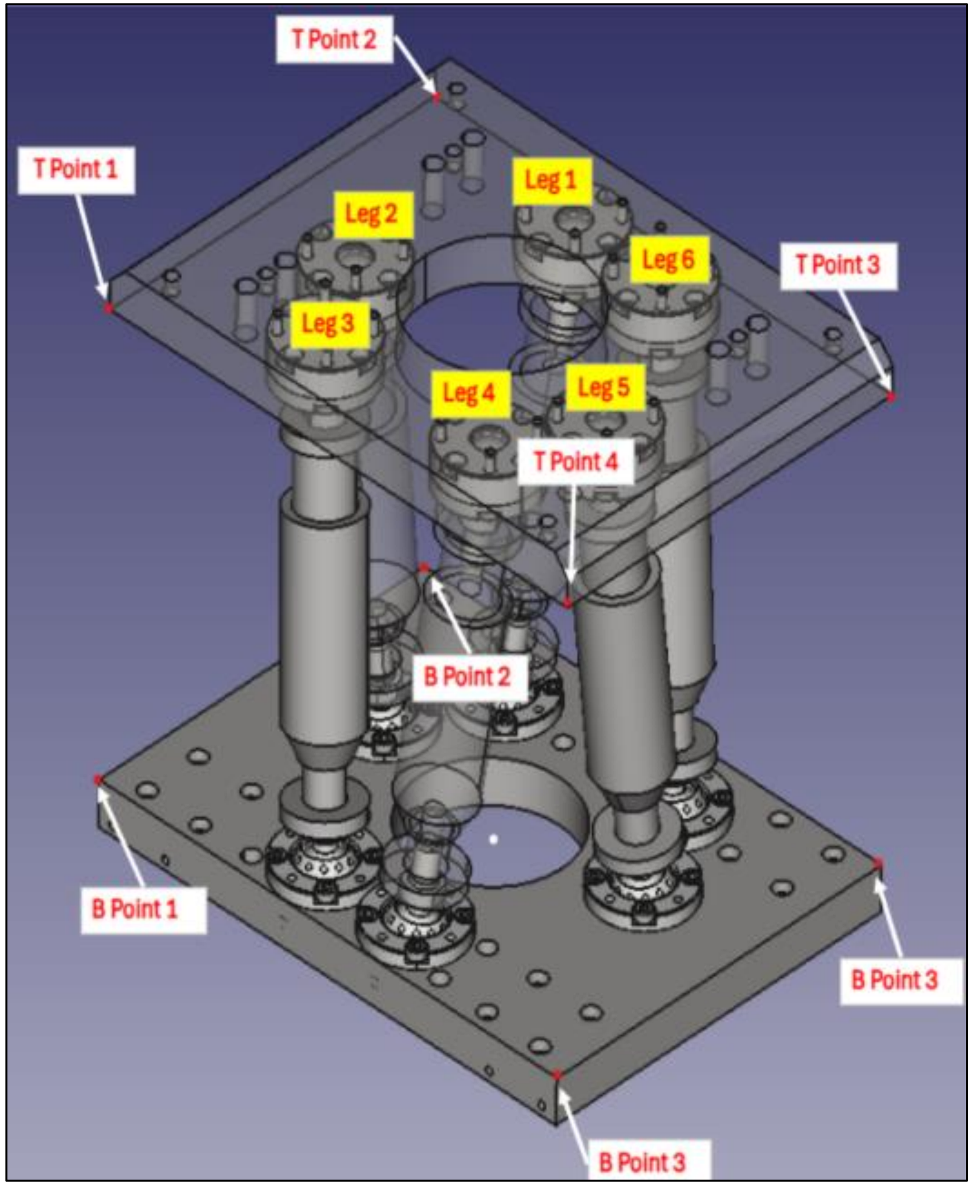
- Take points along the four sides and Top surface of the Fixed platform using CMM.
- Using the intersection of the four side planes and the top plane, locate the four corner points (B points fig) of the top plane of the bottom platform.
- Remember origin at center of rectangle because Simulink MATLAB modeled according to it.

Repeat for top platform



Simulink Projection

- Construct Top Plane of the Fixed platform using B points (fig) from assembly calibration
- Calculate rotational matrix at center of Fixed platform and project Calibrated Fixed leg points on the plane using 3 non-colinear points Formulae
- Construct Bottom Plane of the Moving platform using T points (fig) from assembly calibration
- Calculate rotational matrix at center of Moving platform and project Calibrated Moving leg points on the Plane using 3 non-colinear points Formulae



Legs length calculation

In the global coordinate measure the distance between fixed platform points and moving platform points gives leg lengths

```
>> case1

case1 =

  180.7243  180.5463  181.5145  180.8529  181.0489  180.4685
```

Checking

Case 1: Kept all gauges at Zero position

- Calculate leg lengths using inverse kinematics (MATLAB -Simulink) and store it as Case1

Case 2: Adjust leg lengths manually Leg 5 and 6 increased by 4mm rest all at Zero position

- Repeat all steps of case 1 and Calculate leg lengths using inverse kinematics (MATLAB -Simulink) and store it as Case2

Case 3: Kept all gauges at +4mm and leg1 at zero position

- Repeat all steps of case 1 and Calculate leg lengths using inverse kinematics (MATLAB -Simulink) and store it as Case3

Verify change:

Case 1 Lenths + increments = Case 2 or 3 Lenths

Conclusion

Case 1: All gauges at Zero position

Case 2: Gauge 5 & 6 +4mm, rest all Zero

Case 3: Gauge 1 Zero, rest all +4mm

Model predicts the change in leg lengths perfectly

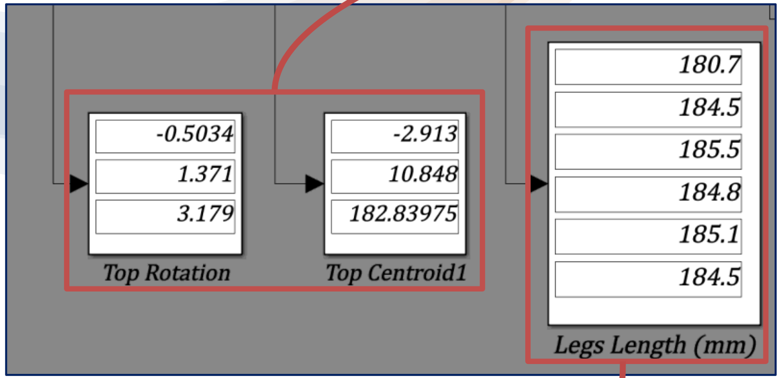
https://github.com/kodandachalla/Hexapod_Calibration

MATLAB Command Window

Text Document

```
>> case1

case1 =

  180.7243  180.5463  181.5145  180.8529  181.0489  180.4685

>> case2

case2 =

  180.7056  180.5515  181.5194  180.8481  185.0625  184.4675

>> case3

case3 =

  180.7076  184.5453  185.5360  184.8335  185.0699  184.4842

>> case2-case1

ans =

   -0.0186    0.0052    0.0049   -0.0048    4.0136    3.9990

>> case3-case1

ans =

   -0.0167    3.9990    4.0215    3.9807    4.0210    4.0157
```

Comparisons



Simulink Model

Inverse Kinematics of Stewart Platform

Input:	**desired_position (mm) = [X; Y; Z] 3x1 matrix** **desired_orientation (deg) = [rotX; rotY; totZ] 3x1 matrix**
output:	**6 Leg lengths in mm (leg_lengths) 1x6 Matrix**

```
% Define the desired position and orientation of the moving platform
desired_position = [-2.913;   10.848;  182.83975]; % Desired position [x
desired_orientation = [-0.5034;    1.371;    3.179]*pi/180; % Desired ori
```

```
Leg Lengths are:
  180.7695

  184.5884

  185.4980

  184.7730

  185.0452

  184.5029
```

Forward Kinematics of the Stewart Platform

Input:	**6 Leg lengths in mm (leg_lengths) 1x6 Matrix**
output:	**desired_position (mm) = [X; Y; Z] 3x1 matrix** **desired_orientation (deg) = [rotX; rotY; totZ] 3x1 matrix**

leg_lengths = [180.7076 184.5453 185.5360 184.8335 185.0699 184.4842];

```
fsolve completed because the vector of fun
as measured by the value of the function t
the problem appears regular as measured by

<stopping criteria details>
Position and orientation of the platform:
x: -2.7641
y: 10.9441
z: 182.8363
roll: -0.5334
pitch: 1.4173
yaw: 3.1751
```

THANK YOU
KODANDA CHALLA