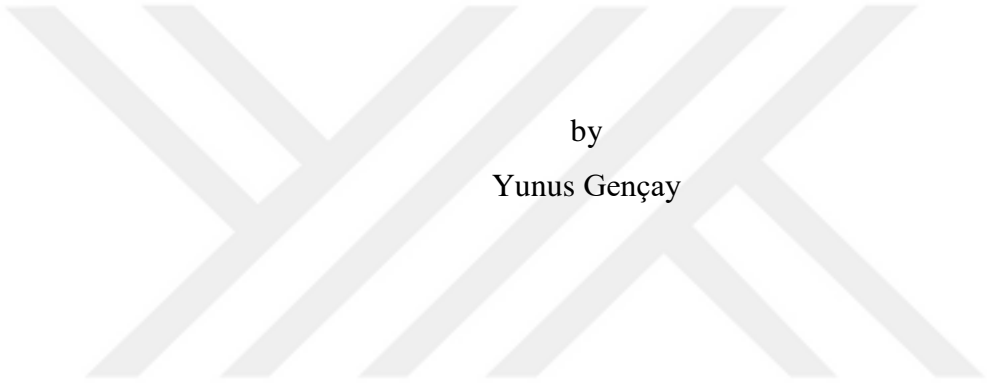


THE MODELING AND SIMULATION OF A STEWART PLATFORM USING
THE LABVIEW ENVIRONMENT



by
Yunus Gençay

Submitted to the Institute of Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science
in
Electrical and Electronics Engineering

Bilgi University
2018

THE MODELING AND SIMULATION OF A STEWART PLATFORM USING
THE LABVIEW ENVIRONMENT

APPROVED BY:

Prof. Dr. Ahmet Denker
(Supervisor)

Prof. Dr. Alpaslan Parlakçı

Asst. Prof. Dr. Murat Tümer

DATE OF APPROVAL: 10/01/2018

ACKNOWLEDGEMENT

The author wants to express his great gratitude to Prof. Dr. Ahmet Denker, his thesis advisor, to advice the subject of the study, great support and incentive about the research and also, he gives special thanks to his wife, for her love and patience.



ABSTRACT

THE MODELING AND SIMULATION OF A STEWART PLATFORM USING THE LABVIEW ENVIRONMENT

This thesis presents a prototype of 6 DOF parallel manipulator known as the Stewart Platform with its hardware structure and control principle. It focuses on the main function module and realization of the control system software. The platform is commonly applied in parallel manipulator to carry out angular and linear rotations. The Platform is actuated by the hydraulic motors in the past for the operating mechanism. Servo systems and servo control technologies arise in industry applications, the advantages of low-cost and high-efficiency helps the electric servo system moderately changing standard hydraulic systems in the many fields of industry sector in recent years.

In this thesis, kinematic analysis of the platform was modeled in Processing (IDE) and, dynamic model of the platform is advanced in MATLAB library that is provided to get proper motions and rotations. The platform consists of two major (upper and lower) plates, six servo motors and six legs linking top plate to base plate via aluminum joints. Each servo motor is connected to the myRIO embedded device and motor shield to provide the control rotations and motions of the system. The control panel of the platform is designed based on LabVIEW environment. The system control is performed by entering the specified angles related to the roll, pitch and yaw regarding the coordinates of x , y , and z manually.

Keywords: Stewart Platform, Parallel Manipulator, LabVIEW, myRIO,

ÖZET

LABVIEW ORTAMI KULLANILARAK STEWART PLATFORMUN MODELLENMESİ VE BENZETİMİ

Bu tez, donanım yapısı ve kontrol prensibi ile Stewart Platform, olarak bilinen 6 serbestlik dereceli paralel manipülatörün prototipini sunmaktadır. Sistemin, ana fonksiyon modülüne ve kontrol sistem yazılımına odaklanılmıştır. Platform, genellikle hareket simülatöründe, dinamik doğrusal ve açısal hareketleri gerçekleştirmek için kullanılmıştır. İlk Stewart Platformlar, sürüş mekanizması olarak hidrolik motorlarla hareket ettirilmekteydi. Ancak endüstri uygulamalarında elektronik kontrollü servo sistemi ve seri servo kontrol teknolojileri ortaya çıktıkça, düşük maliyet ve yüksek verimlilikle çalışan bu sistemlerin standart ve pahalı hidrolik sistemlerin yerine geçtiği gözlemlenmiştir.

Bu tezde, Stewart platformun kinematik analizi Processing ortamında tasarlanmış ve simüle edilmiştir. Ayrıca, dinamik modeli uygun hareketler ve rotasyonlar elde edebilmek amacıyla MATLAB ortamında geliştirilmiştir. Platform iki ana (üst ve alt) plaka, altı servo motor ve üst plakayı alt plakaya bağlayan altı alüminyum bacadan oluşmaktadır. Her bir servo motor, sistemin kontrolünü sağlamak amacıyla myRIO gömülü sistemi ve tasarlanan motor sürücü kullanılarak çalıştırılmıştır. Platformun kontrol paneli LabVIEW kütüphanesinden faydalanılarak tasarlanmıştır. İlgili sistem kontrolü, kontrol paneli üzerinden x, y ve z koordinatlarına göre, roll, pitch ve yaw özel açılarının manuel olarak girilmesiyle belirlenen hareketleri gerçekleştirir.

Anahtar Kelimeler: Stewart Platform, Paralel Manipülatör, myRIO, LabVIEW,

TABLE OF CONTENTS

ACKNOWLEDGEMENT	iii
ABSTRACT	iv
ÖZET	v
TABLE OF CONTENTS	vi
LIST OF FIGURES	vii
LIST OF TABLES	10
LIST OF SYMBOLS / ABBREVIATIONS	11
1. INTRODUCTION	12
2. MATHEMATICAL MODELING	16
2.1. DYNAMIC STRUCTURE	18
2.2. KINEMATICAL EQUATIONS	19
3. SYSTEM CONSTRUCTION	22
3.1. PRODUCTION	22
3.2. MECHANICAL STRUCTURE	26
3.3. ELECTRONIC STRUCTURE	32
4. SYSTEM MODELING AND ANIMATIONS	36
4.1. SOLIDWORKS MODEL	36
4.2. MATLAB MODEL	38
4.3. PROCESSING MODEL	40
5. CONTROL PANEL	41
6. CONCLUSION	47
REFERENCES	48
APPENDIX A	50
APPENDIX B	54

LIST OF FIGURES

Figure 1.1. A Flight Simulator as Stewart Platform [1].....	13
Figure 1.2. Tire Testing Machine by Eric Gough, as a Stewart Platform [2].....	14
Figure 2.1. Stewart Platform with located the position of servo motors [3]	16
Figure 2.2. Dynamic Model of Stewart Platform [4].....	18
Figure 2.2.1. Servo Motor Coordinate System [5].....	20
Figure 3.1.1. The technical drawing for the upper base plate.....	22
Figure 3.1.2. The technical drawings for the lower base plate.....	22
Figure 3.1.3. The technical drawing of the servo motor.....	23
Figure 3.1.4. The technical drawings for the servo extension.....	23
Figure 3.1.5. The technical drawing of the bearing's external piece.....	24
Figure 3.1.6. The technical drawing of the bearing's internal piece.....	24
Figure 3.1.7. The technical drawing of the L part.....	24
Figure 3.1.8. The technical drawing of the plate.....	25
Figure 3.2.1. The DXF format of the lower base as an example.....	26
Figure 3.2.2. The CNC router during the laser cutting.....	26
Figure 3.2.3. The base and the plate parts.....	27
Figure 3.2.4. One of the upper base plates with the cuts.....	27
Figure 3.2.5. The metal saw during operation.....	27
Figure 3.2.6. The CNC router during metal cutting	28

Figure 3.2.7. Metal-bending machine in operation.....	28
Figure 3.2.8. The screwing of the helix pattern inside.....	29
Figure 3.2.9. The helix pattern driven by the screwing machine.....	29
Figure 3.2.10. The servo extensions, after sanding and galvanization.....	30
Figure 3.2.11. One of the servos with its extension assembled.....	30
Figure 3.2.12. The base being assembled.....	30
Figure 3.2.13. Last view of the Stewart Platform.....	31
Figure 3.3.1. Servo Motor – HITEC HS-5485HB.....	32
Figure 3.3.2. Technical drawing of Servo Motor.....	32
Figure 3.3.3. MyRIO microprocessor for controlling the system.....	33
Figure 3.3.4. PCBA design drawing.....	33
Figure 3.3.5. PCBA design.....	34
Figure 3.3.6. MyRIO – Motor connections.....	34
Figure 3.3.7. General aspect of Stewart Platform.....	35
Figure 4.1.1. SolidWorks Model.....	36
Figure 4.1.2. System Cockpit Model.....	37
Figure 4.1.3. 3D printed model.....	37
Figure 4.2.1. MATLAB model & animation.....	38

Figure 4.2.2. (a) Upward Pitch Motion, (b) Right Side Roll Motion.....	39
Figure 4.2.3. (a) Downward Pitch Motion, (b) Left Side Roll Motion.....	39
Figure 4.3.1. Processing model.....	40
Figure 5.1. LabVIEW control panel.....	41
Figure 5.2. Constant Parameters and Vectors of the Platform from Main.vi.....	42
Figure 5.3. Formula Representation as Rotational Matrix.....	42
Figure 5.4. Code Section for Servo Motors.....	43
Figure 5.5. Motor Angle Block diagram.....	43
Figure 5.6. Platform Angle, Magnitude and Time Out Table.....	44
Figure 5.7. Output Vectors.....	44
Figure 5.8. Communication Scheme Servos to myRIO.....	44
Figure 5.9. General view of control panel with PWM.....	45
Figure 5.10. (a) Upward Pitch Motion, (b) Downward Pitch Motion.....	46
Figure 5.11. (a) Left Side Roll Motion, (b) Right Side Motion	46

LIST OF TABLES

Table 1.1. Properties of driving mechanism for servo and. hydraulic motors.....	12
Table 2.1. Fixed points of platform and angular coordinates.	16
Table 3.1.1. The list of connector hardware to be.....	24
Table 5.1. Motion variables of special rotations.....	44

LIST OF SYMBOLS / ABBREVIATIONS

L_i	The length of the leg
p_i	The platform attachment points.
b_i	The base attachment points
P_i	Platform attached points
R_b	Radii of the circle of base.
R_p	Radii of the circle of system.
m_i	The middle of platform joint linked to the servo horn.
Δ_i	The changes of rotation angle dependence.
D	The length of located rods.
M_i	The vector by origin to m_i .
R_m	Radii of the point that is combined to the servo horn.
a_i	Positive angles for servos.

1. INTRODUCTION

The most graceful way the usage of a robotic constellation named as a Stewart platform, first employed by Eric Gough in 1954 and published by D. Stewart in 1965. It is the parallel kinematic mechanism that can provided as controlling of base movement via 6-Degree of Freedom, such as production operations and related tasks.

The Stewart Platform comprises of located plate (fixed platform) and movable platform (moving plate) that are linked by six platform legs. The joints of the platform connecting the legs to the fixed plate and moving plate is spherical joint, that could present a useless rotation of the legs for their axes. They will not affect the all platform performance and, that is extinguished by changing according to the spherical joint on end of the linked with extensive joints. The craved position and rotation of the movable plate are achieved by connecting the lengths of the all platform legs, performing the six degrees of freedom which are transitional into three positions (surging/swaying/heaving) and three orientations (pitching/rolling/yawing). The lengths of all legs cannot be modified independently, but in such a cases which the system construction is allowed. Stewart Platform comprises of six extensible actuators, and located by the stationary platform and the movable platform. By severally, controlling the length of each servo motor bottomed on so complicated mathematical equations. Stewart Platform can accurately move for 6-DOF (up/down, left/right, forward/back, and other rotations and positions according to each axis). In this study, servo motors are used instead of the hydraulic actuators.

Driving Mechanism	
<i>Advantages</i>	
<i>Hydraulic</i>	1. High load capacity and a higher relative strength of an electrical drive.
	2. Produce higher acceleration
	3. Oil is substantially incompressible and cannot absorb any of the energy systems.
	4. The actuator is small and durable.
	5. System operating conditions consistent
<i>Servo</i>	1. The system is clean and does not need to apply any other tool or equipment.
	2. Acceleration is a large change.
	3. High Capability and Efficiency.
	4. Easy to install and maintain
	5. Not complicated actuation component
<i>Disadvantages</i>	
<i>Hydraulic</i>	1. There are concerns about the oil spill, or on fire.
	2. Acceleration has small changes and the dynamic response is low.
	3. Take big space, very loud and difficult to maintain.
	4. Installation is very complicated and inefficient.
<i>Servo</i>	1. Acceleration is poor.
	2. The actuator mechanism and safety equipment are complicated.
	3. Large power changing requirement in movement.

Table.1. Properties of driving mechanism for servo and hydraulic motors

The advantages of the Stewart platform are 6 DOF which create usable payload to weight ratio, high dynamics, great positioning accuracy due to parallel kinematics, excellent flexibility with many accessible-DOF and mobile platform positioning.

Stewart Platform structure has been broadly used in various areas of industrial such as medicine, aerospace, defense, automotive and machining. Examples from these industrial regions, the throwing platform of rockets, helicopter runway, surgical operations, sensitive laser cutting, but, it's probably best known for its use in producing the precise movement required of flight simulators.

Stewart Platform was invented as a flight simulator by D. Stewart in Figure 1.1. that became three parallel linear actuators, but also, V. Eric Gough had previously suggested similar tire test machine to model of D. Stewart that included six motors were used as a system. V. Eric Gough is the first person who advanced, utilized and used this type of a parallel manipulator in Figure 1.2. However, Stewart platform is mostly called as a Stewart-Gough Platform.



Figure 1.1. A Flight Simulator as Stewart Platform^[1]



Figure 1.2. Tire Testing Machine by Eric Gough, as a Stewart Platform^[2]

There are lots of different descriptions used by researchers in the parallel manipulators' field. It's entirely required to identify them for current research to understand the meanings better. The most common terms are stated for this thesis:

Parallel systems: A system in which the movable plate was linked to the fixed plate by two independent kinematic systems.

In-parallel system: 6 Degrees of Freedom parallel platform with two plates linked the six connections.

Base Platform: The fixed plate of the parallel manipulator.

Movable Platform: The top plate which is combined to the fixed plate with legs.

Legs: The kinematic chains linking the upper plate and the lower plate in parallel. The other name is named as a connector.

Joint: The real connection between two plates to provide platform rotations.

DOF: Degree of freedom.

SP: Stewart Platform.

2. MATHEMATICAL MODELING

This part represents the mathematical model used in this work in order to define the kinematics of Stewart platforms. Although the Stewart platform lends itself to the description in the framework of screw theory, the mathematical treatment in this study only assumes the basic knowledge of linear transformations. The upper and the lower platform are linked by 6 extensible legs which are attached to get arbitrary locations b_i on the base and p_i on the surface of platform ($i \in \{1, \dots, 6\}$). The transformations between the platform, the base coordinates and the inverse transformations are realized by means of three consecutive Euler rotations in the x - y - z coordinates and a subsequent translation with the rotation matrix defined as

$$\mathcal{R} = \mathcal{R}_z(\gamma)\mathcal{R}_y(\beta)\mathcal{R}_x(\alpha) \quad (1)$$

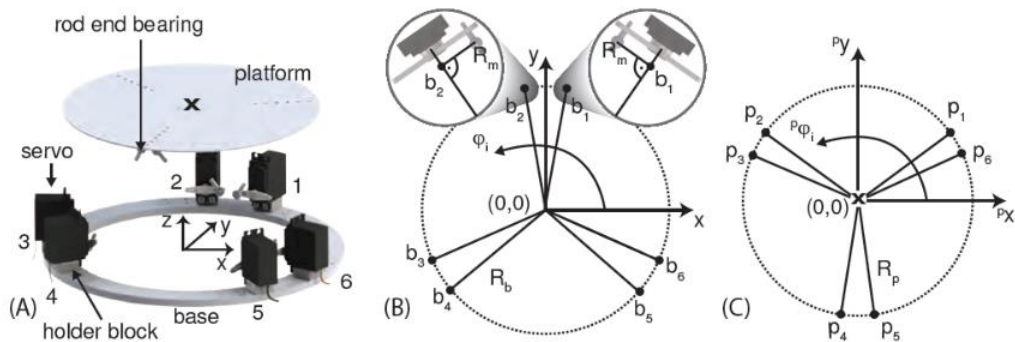


Figure 2.1. Stewart Platform with located the position of servo motors. ^[3]

with being translation vector;

$$\boldsymbol{\tau} = (t_x \ t_y \ t_z)^T \quad (2)$$

The transformation which is corresponding is indicated by improving two designators (b and p), thus ${}^b\mathcal{R}_p = \mathcal{R}$ ${}^b\boldsymbol{\tau}_p = \boldsymbol{\tau}$ which means to base plate position and rotations

$${}^b\mathcal{R}_p = ({}^b\mathcal{R}_p)^{-1} = ({}^b\mathcal{R}_p)^T \quad (3a)$$

$${}^b\boldsymbol{\tau}_p = -({}^p\boldsymbol{\tau}_b) \quad (3b)$$

are the opposite relations. Vectors can be written in uppercase and become the prefix p only if they are stated in the platform. According to the above definitions the leg vector L_i of leg i in base becomes

$$L_i = {}^p\tau_b + {}^P\mathcal{R}_b P_i - B_i = P_i - B_i \quad (4)$$

The Euclidean norm vector which is defined length of the leg,

$$|L_i| = \|L_i\|_2 \quad (5)$$

The related calculations are used to calculate the lengths of platform legs in the structure in Figure 2.1 shown the related angles which motors made. The fixed attachment points shown b_i are defined projection of joints center at the axes rotation as system rotations. The attached points of platform p_i is used to centers of the related joints linked to the platform. All decided points are found easily,

$$b_i = (x_i \ y_i \ z_i)^T = (\mathcal{R}_b \cos(\gamma_i) \ \mathcal{R}_b \sin(\gamma_i) \ 0)^T \quad (6a)$$

$$p_i = ({}^P x_i \ {}^P y_i \ {}^P z_i)^T = (\mathcal{R}_p \cos({}^P \gamma_i) \ \mathcal{R}_p \sin({}^P \gamma_i) \ 0)^T \quad (6b)$$

The related angles are presented as a table in Table 2. 1. Being the radii of circle with \mathcal{R}_b and \mathcal{R}_p , b_i and p_i is also in table. By the way, all angles are arranged in degrees and related calculations need to be made in radii.

i	1	2	3	4	5	6
γ_i	77°	103°	197°	223°	317°	343°
${}^P \gamma_i$	37.5°	142.5°	157.5°	262.5°	277.5°	22.5°

Table .2.1. Fixed points of platform and angular coordinates.^[4]

2.1. DYNAMIC STRUCTURE

The Stewart Platform is a six degree of freedom parallel manipulator which consists of a strict body movable platform, linked to a fixed platform through six extensible legs. All legs are at the same kinematic chains, the moveable top and fixed base platform via independent joints. Every legs include a motor that lengths of the legs are changeable that can be controlled to provide the precise motion of movable platform.

The dynamic structure of the Stewart Platform is hard in comparing with the serial mechanism because of the various kinematic chains are all connected to the top plate. The key point that have been used to define the problem of the system and achieve the dynamical modelling of the Stewart Platform model. There is no too much formulation to define the system. In this structure, the used formulation is Lagrange which is provided the related analytical structure. The derivation of the calculations that system is divided at two parts such as top platform and the extensible legs. The types of energies that are calculated to used the related equations, are kinetic and potential.

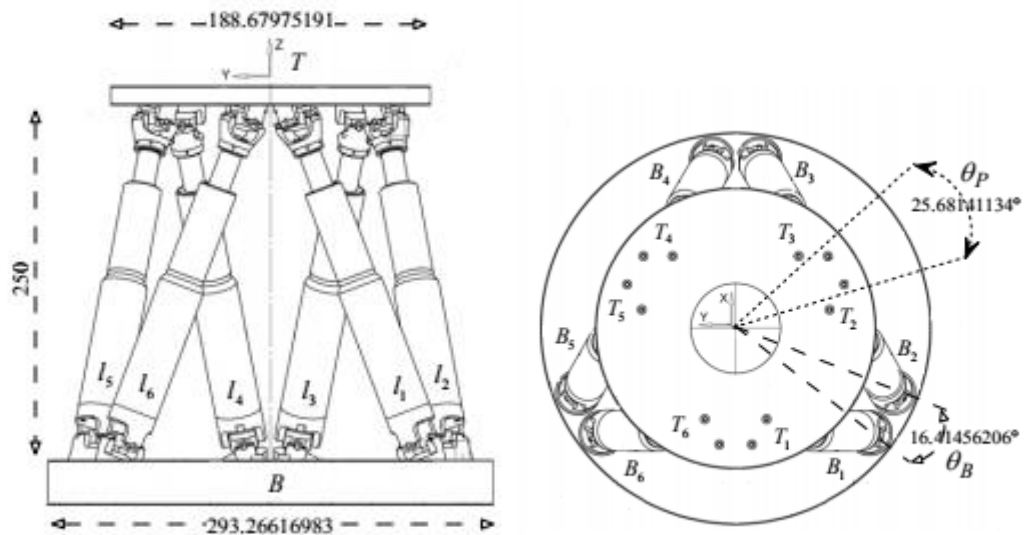


Figure 2.1.1. Dynamic model of Stewart Platform. [5]

2.2. KINEMATICAL EQUATIONS

The main purpose of controlling the Stewart platform is to solve the kinematic equations to find all the lengths of the platform for the desired positions and orientations of the platform. The Inverse Kinematic problem is already solved which is shown in Equation 4. The problem presents a unique analytical solution, unlike the advanced kinematic problem, which is not highly linear and usually requires iterative approaches or additional sensory information in table 2.1. However, since rotary actuators and fixed length rods are used, there is no real variable length of legs on the platform of the section. Attaching the defining of a leg that linked the value between b_i and p_i , the related length is accomplished essentially to change the locations of p_i , the lower platform rotation of servo motor i . Local coordinates of the origin for every servo motors b_i and rotation axis m_z the fixed coordinates which are pointed at the origin. The linking which is center of the joint to the servo horn m_i changes rotation angle immediately Δ_i . Via R_m radius of joint to servo horn, D the linked length of rod and M_i the vector point from the base between origin m_i .

$$R_m = R_{mi} = |M_i - B_i| \quad (7a)$$

$$D = D_i = |P_i - M_i| \quad (7b)$$

The vector which is end spot M_i , found the related transformations,

$$M_i = (x_{mi} \ y_{mi} \ z_{mi})^T = {}^{mi}\tau_b + {}^{mi}\mathcal{R}_b(\mathcal{R}_m \ 0 \ 0)^T \quad (8)$$

where

$${}^{mi}\tau_b = (x_i \ y_i \ z_i)^T, \quad (9a)$$

$${}^{mi}\mathcal{R}_b = \mathcal{R}_z(\gamma_i - \frac{\pi}{2})\mathcal{R}_y(-\Delta_i) \quad (9b)$$

The upper calculation is shown rotation matrix which is related the even servo motor angles and odd values of servo motors have their joints linked to the servo horns which is shown in Figure 2.2.1.

The joints that are such an adjustment of small differences between the design parameter [28, 16] to non-equal design of the system. The related coordinates which are calculated odd number m_i , the rotation depended terms of sign chances x_{mi} and y_{mi} . In short computing,

$$\begin{pmatrix} x_{mi} \\ y_{mi} \\ z_{mi} \end{pmatrix} = R_m \begin{pmatrix} \cos(\Delta_i) \sin(\gamma_i) + x_i \\ \cos(\Delta_i) \cos(\gamma_i) + y_i \\ \sin(\Delta_i) + z_i \end{pmatrix} \quad (10)$$

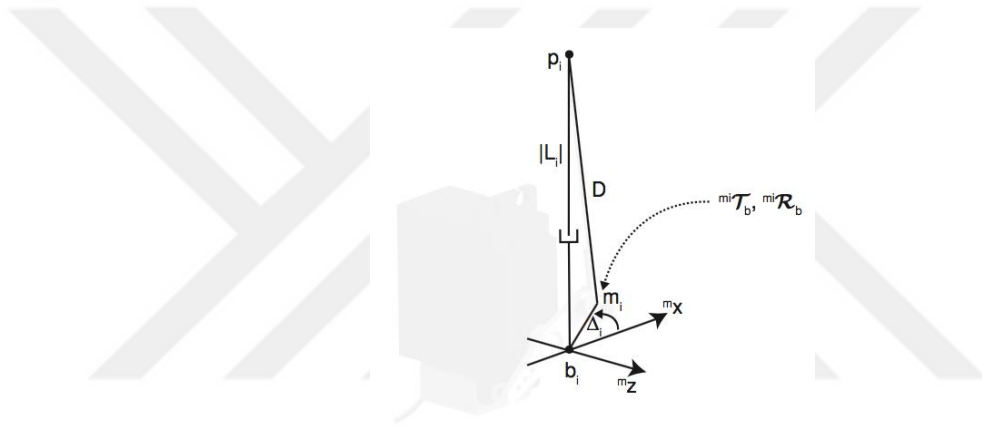


Figure 2.2.1. Servo Motor Coordinate System (i) with expression of values and constants included in the calculation of the length of leg L_i .^[7]

Positional or orientation changes of Stewart Platform which is represent as a vector P_i and length of legs L_i . The calculation found to solve Inverse Kinematic problem which includes of Δ_i .

$$\mathcal{R}_m^2 = (M_i(\Delta_i) - B_i)^T (M_i(\Delta_i) - B_i) \quad (11a)$$

$$D^2 = (P_i - M_i(\Delta_i))^T (P_i - M_i(\Delta_i)) \quad (11b)$$

$$|L_i|^2 = (P_i - B_i)^T (P_i - B_i) \quad (11c)$$

for all $i \in \{1, \dots, 6\}$. Integrating the upper calculations,

$$|L_i|^2 - D^2 + \mathcal{R}_m^2 = 2(B_i - M_i(\Delta_i))^T (B_i - P_i) \quad (12)$$

which after substituting from (8) resolves into

$$\begin{aligned} \pm(|L_i|^2 - D^2 + \mathcal{R}_m^2) &= 2\mathcal{R}_m(z_{Pi} - z_i) \sin(\Delta_i) + 2\mathcal{R}_m[\sin(\gamma_i)(x_{Pi} - x_i) \\ &\quad - \cos(\gamma_i)(y_{Pi} - y_i)] \cos(\Delta_i) \end{aligned} \quad (13)$$

via the upper sign corresponding to even and the lower sign to odd servo motors as previous value. The equation is related with sinusoidal functions which include the linear combinations. By using the trigonometric identity.

$$a \sin(\phi) + b \cos(\phi) = \sqrt{a^2 + b^2} \sin(\phi + \varphi) \text{ with } \varphi = \arctan\left(\frac{b}{a}\right) + \begin{cases} 0, a \geq 0 \\ \pi, a < 0 \end{cases}$$

having

$$a_i = 2\mathcal{R}_m(z_{Pi} - z_i) \quad (14a)$$

$$b_i = 2\mathcal{R}_m[\sin(\gamma_i)(x_{Pi} - x_i) - \cos(\gamma_i)(y_{Pi} - y_i)] \quad (14b)$$

$$c_i = |L_i|^2 - D^2 + \mathcal{R}_m^2 \quad (14c)$$

To consider ai servo angles are found to be which is positive,

$$\Delta_i = \arcsin\left(\frac{\pm c_i}{\sqrt{(a_i^2 + b_i^2)}}\right) - \arctan\left(\frac{b_i}{a_i}\right) \quad (15)$$

Considered the joints with very wide range of angular motion, the platform can be reached the desired positions and orientations if the absolute solution is (15) available for all values of i .

3. SYSTEM CONSTRUCTION

3.1. PRODUCTION

The production part began by the drawing of the base plates with SolidWorks first. The servo motors were to be placed under two identical-in-area base plates, with one having 5 mm thickness and the other having 10 mm in Figure 3.1.1. The edges were nulled to reduce the stress concentration and avoid having sharp edges that can do harm. Holes need to be planted for the servos to be placed, for bolts to be adjusted and the servo cables to be driven through. The final result was as the following:

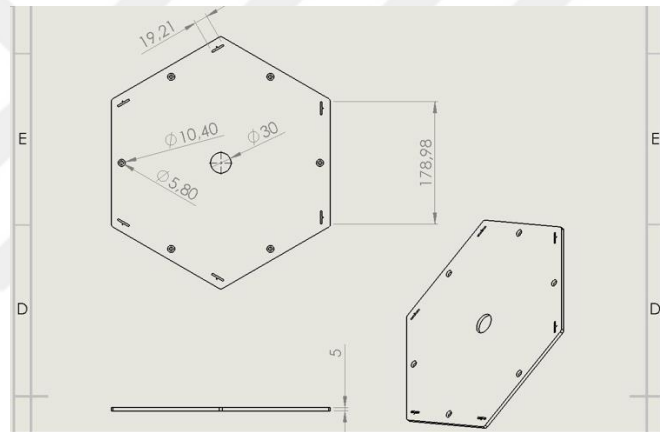


Figure 3.1.1. The technical drawing for the upper base plate

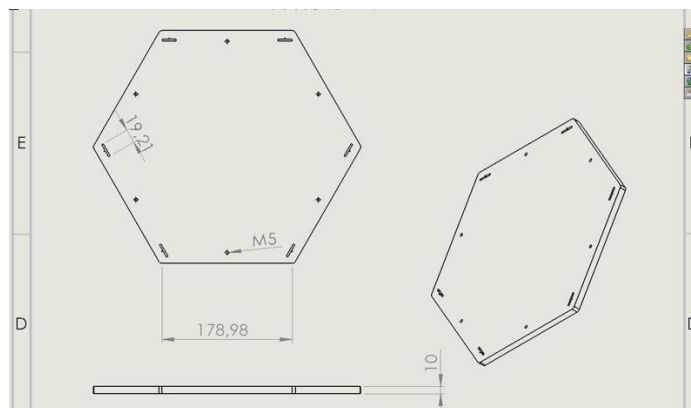


Figure 3.1.2. The technical drawings for the lower base plate

The SolidWorks drawing of the HS-5485HB servo was found using the website GrabCAD, and all six motors were fitted into the model easily in Figure 3.1.3. (with a distance of 12mm from the edges).

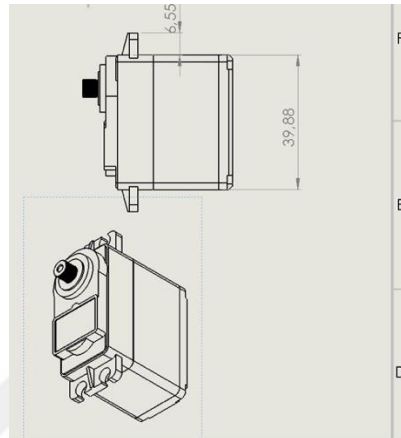


Figure 3.1.3. The technical drawing of the servo motor

The first servo arms were not sufficient to provide us with the desired length, so the next drawings were the servo extensions in Figure 3.1.4.

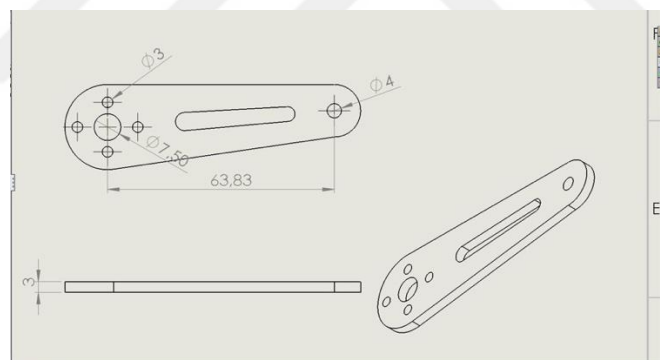


Figure 3.1.4. The technical drawings for the servo extension

However, the legs were to be attached using M4 bearings, so their lengths were also taken into account while adjusting the optimal leg lengths. The bearings consist of two parts: the external housing and the internal moving spherical piece in Figure 3.1.5. They were decided to be attached to both ends of the legs, to adjust the parts that would later carry the plate. The bearings were chosen because it is harder to manufacture and maintain bent legs than straight ones. Fortunately, the M4 bearings allow about up to 10° of tilting angle as desired.

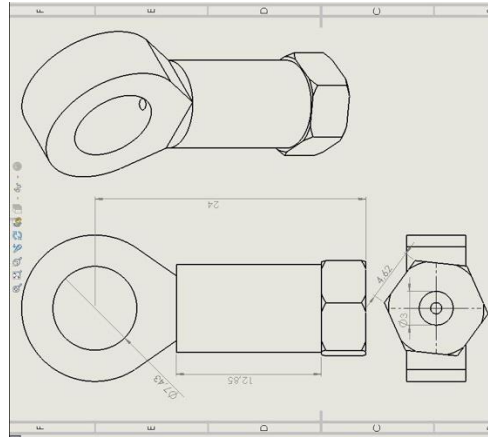


Figure 3.1.5. The technical drawing of the bearing's external piece.

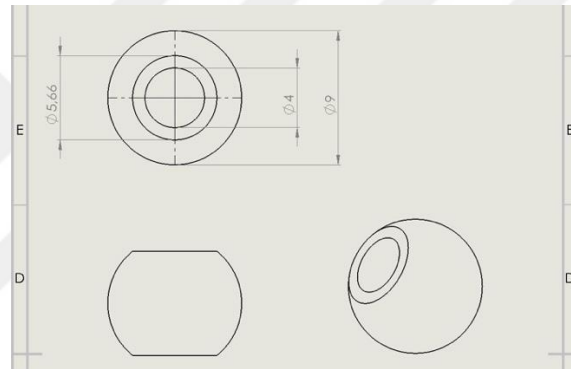


Figure 3.1.6. The technical drawing of the bearing's internal piece.

The upper plate was to be linked to the top bearings of the legs by specially-constructed L-pieces with M4 bolts in Figure 3.1.7.

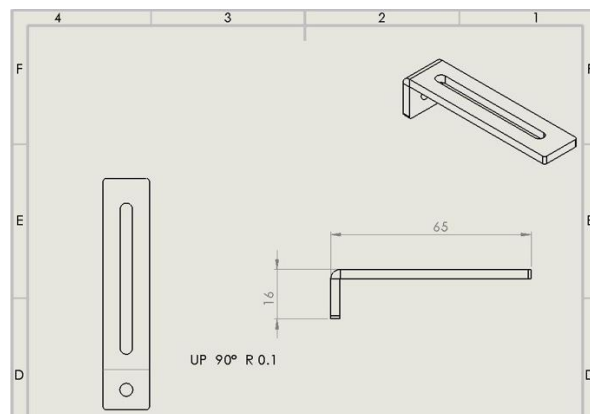


Figure 3.1.7. The technical drawing of the L part.

L part was designed to be manufactured by metal-bending, so the upside view seen on the left is drawn as an unbent piece. The lengthy hole was intended to house the M4 bolts to connect the plate to the L-part.

The last piece to be sketched and modelled was the upper plate with the same type of lengthy holes to increase the assembly flexibility, the initial locations were marked again after the final assembly in Figure 3.1.8.

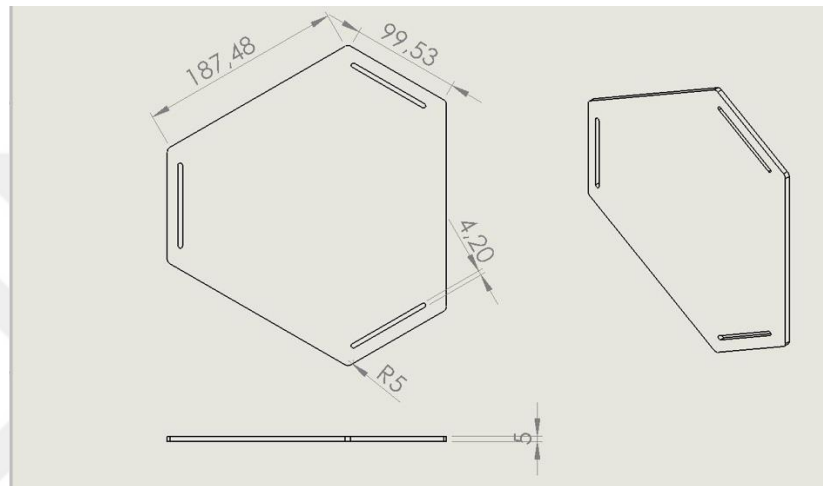


Figure 3.1.8. The technical drawing of the plate

After the modeling of this last part, the necessary assembly adjustments including the bolts, countersinks, setscrews, and bolts had to be added to the final assembly which is stated in Table 3.1.1.

METRIC	TYPE	COUNT
M5 X 55	Countersink	6
M4 X 20	Nut	12
M4 X 14	Nut	6
M4	Bolt	12
M3 X 8	Nut	24
M4 X 16	Setscrew	12

Table 3.1.1. The list of connector hardware to be used.

3.2. MECHANICAL STRUCTURE

The first parts to be produced were the plate and base (upper & lower). The upper base and lower base plates were produced from white Plexiglas, while the plate was produced from fully transparent Plexiglas, and was cut with laser on a CNC laser-cutting counter in Figure 3.2.2. For the pieces to be used with the necessary laser cutting software (AutoCAD ArtCAM Pro in the case of our manufacturer), however, the models needed to be converted from SLDPRT to .DXF format.

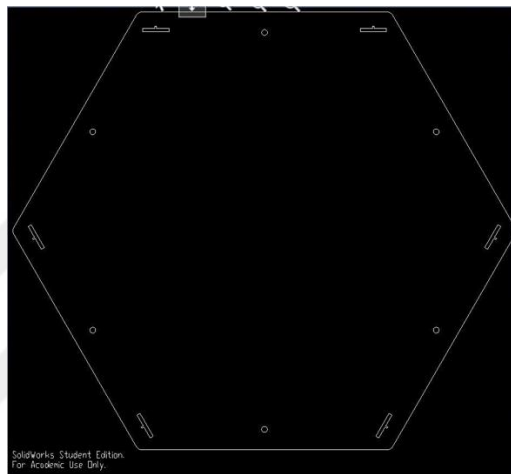


Figure 3.2.1. The DXF format of the lower base as an example



Figure 3.2.2. The CNC router during the laser cutting.

However, while cutting the upper base plate, so that the bolts to be attached to the rear of the servos do not prevent the servos' rotary movement upwards. Some 5mm-deepcuts needed to be added in Figure 3.2.4.

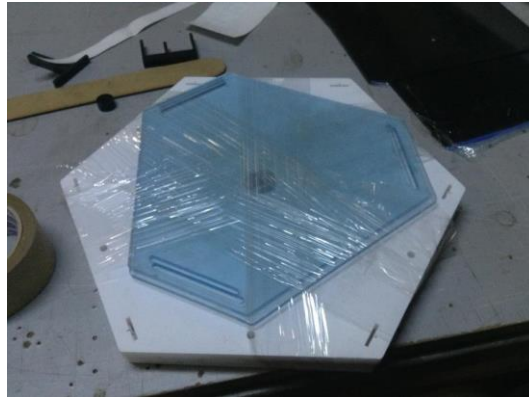


Figure 3.2.3. The base and the plate parts.

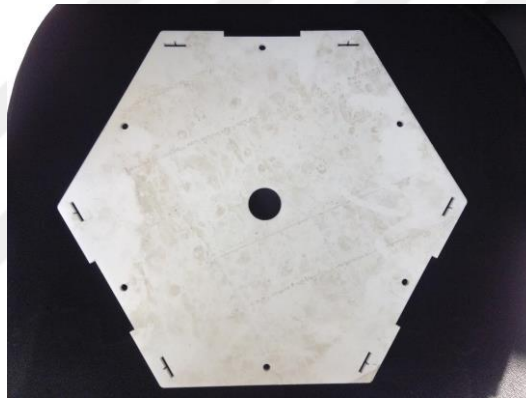


Figure 3.2.4. One of the upper base plates with the cuts

The legs were obtained from a specialized manufacturer in the industrial district in İkitelli, İstanbul that had aluminum rods of different diameters and sizes. A metal cutting saw produced the rods with our desired length of 251mm in Figure 3.2.5.



Figure 3.2.5. The metal saw during operation

Note that these rods were for the outer 8mm cylinders of the legs and not for the smaller cylinders on the ends for ease of manufacturability. For the smaller cylinders, we had to use M3 helix nuts on the ends, which will be shown later.

The servo extensions and the straight sheet-metal forms of the L-parts were also made from aluminum and were produced by laser cutting in a different manufacturer in İkitelli who specialized in laser-cutting metal parts in Figure 3.2.6. The 2D drawings were also converted into the .DXF format.



Figure 3.2.6. The CNC router during metal cutting.

The L-parts' sheet-metal forms were taken to another manufacturer who specialized in metal-bending in Figure 3.2.7. The sheet metal was bent there, and calipers checked the resulting pieces' dimensions for quality control.



Figure 3.2.7. Metal-bending machine in operation

However, this was not the end for the metal parts, some slight damages and homogenous apperals due to the heat applied during the laser cutting process. To solve the issue, the metal parts (servo extensions, legs, and L-parts) were first treated with sandpaper to smoothen the surface. After the sanding, the parts were taken to be galvanized not to catch dust or get scratched easily. The galvanization took place in a manufacturer inside the galvanize manufacturers' district in İkitelli. The entire process for the metal parts took about four days. After all the parts were collected and brought, the assembly began by the removal of their protective wrappings and manipulation of the legs. The internal of the leg rods had to be screwed to be able to bear the M3 nuts to attach the bearings to the legs.



Figure 3.2.8. The screwing of the helix pattern inside



Figure 3.2.9. The helix pattern driven by the screwing machine



Figure 3.2.10. The servo extensions, after sanding and galvanization.

The initial servo horns were drilled to bear M3 bolts and the extensions were connected, as showed in Figure 3.2.11.



Figure 3.2.11. One of the servos with its extension assembled.

The base was completed after the servos fitted with extensions were placed in their designated locations in Figure 3.2.12.

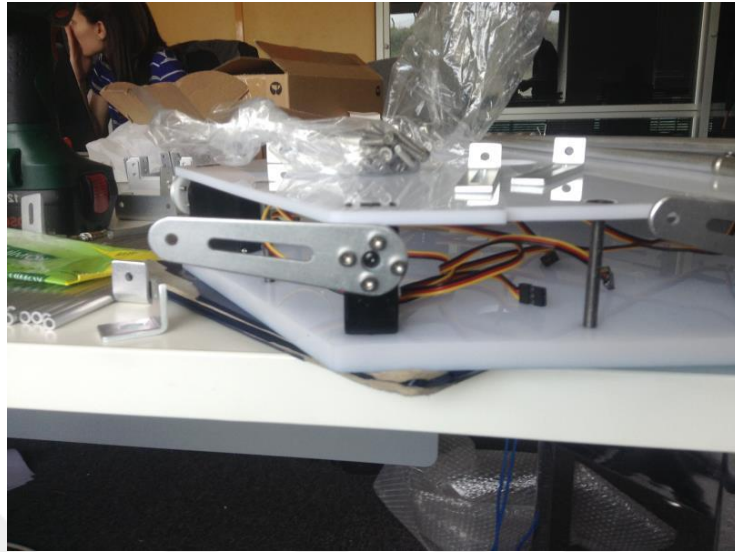


Figure 3.2.12. The base being assembled.



Figure 3.2.13. Last view of the Stewart Platform

Afterwards, with the assembly of the bearings to the legs and the legs to the extensions, and the assembly of the plate to the entire system by the L-parts, the mechanical building was completed in Figure 3.2.13.

3.3. ELECTRONIC STRUCTURE

A 6 DOF Stewart platform is actuated by six standard analog servo motors. The motors located to the base platform. However, selection of the servo motor is Hi-Tec HS 5485HB in Figure 3.3.1. The selected type of this servo is one of the most long-lived and reliable, the features of the Hi-Tec HS 5485HB centering and resolution is excellent to use for the platform.



Figure 3.3.1. Servo Motor – HITEC HS-5485HB

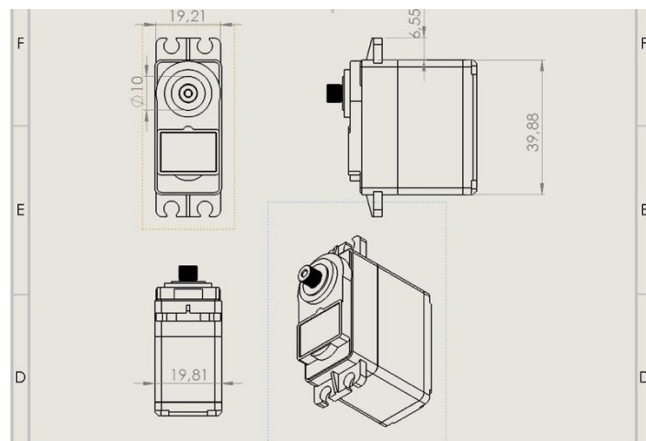


Figure 3.3.2. Technical drawing of Servo Motor – HITEC HS-5485HB

The Controlling of Stewart Platform is provided with myRIO that is a real-time embedded evaluation board made by National Instruments. It is used to develop applications that utilize its onboard FPGA and microprocessor in Figure 3.3.2.



Figure 3.3.3. myRIO microprocessor for controlling the system^[8]

The beginning step of controlling part is started to use the motor shield for providing the electrical connection between myRIO and motors. The motor shield design consists of six sockets for three pins by six motors such as signal (yellow), ground (black) and power (red). The power of the circuit is 5 volts. The PCBA design is shown in Figure 3.3.4.

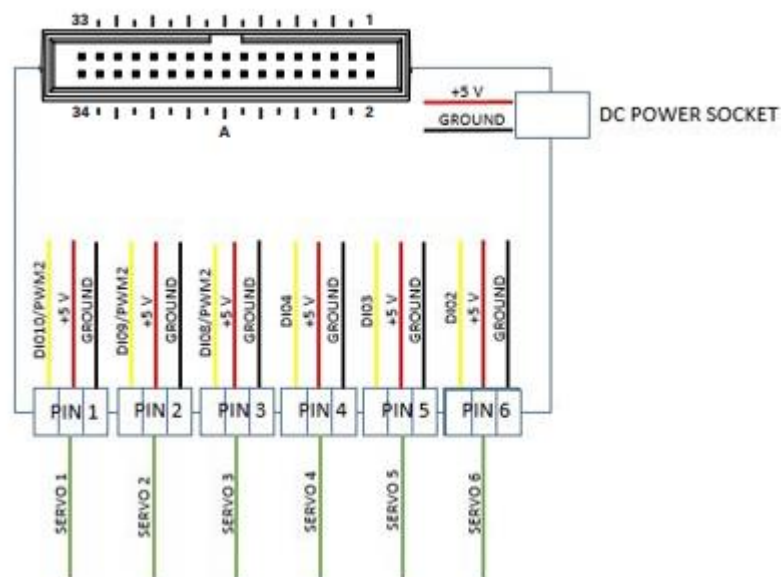


Figure 3.3.4. PCBA design drawing

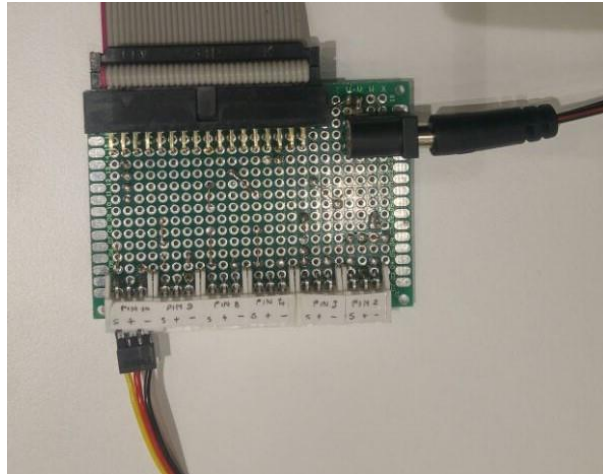


Figure 3.3.5. PCBA design

34 pin data cable provides the communication between myRIO and the motor shield, and motor signal cables (yellow) are connected to the myRIO directly as shown in Figure 3.3.5. Open-Close Signals of myRIO is symbolized 1 and 0 for digital pins. All motors are directly linked to selected pins as illustrated in Figure 3.3.6. That is digital I/O pins.

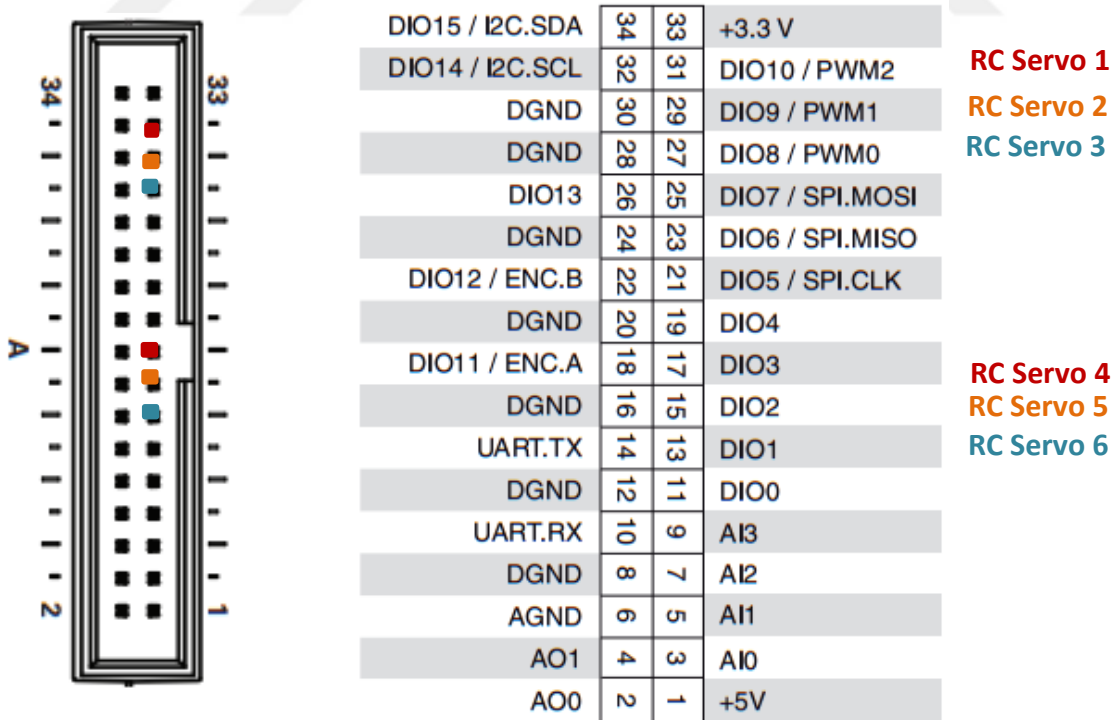


Figure 3.3.6. myRIO – Motor connections

The installation of the platform is started by positioning servo motors between the base and movable plates. Length of the immovable rods are provided to connection between the servo motor horns and upper platform. End of the rod bearings are located to servo horns and holders connected to the platform. The controller board of the platform is mounted by the middle of the center plate in Figure 3.3.7.

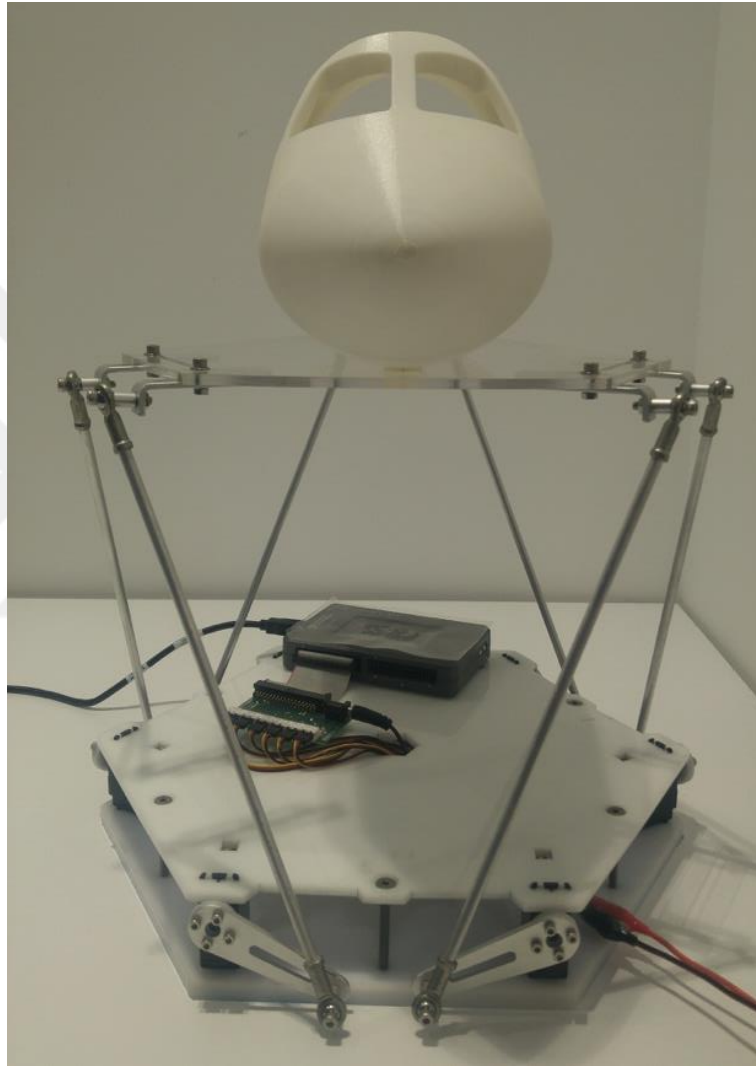


Figure 3.3.7. General aspect of Stewart platform

4. SYSTEM MODELING AND ANIMATIONS

4.1. SOLIDWORKS MODEL

SolidWorks is a solid modeling software used for 3-dimensional design. It uses a parametric feature-based approach to build models and assemblies. Stewart platform and its cockpit design is built up with SolidWorks.

The SolidWorks model of Stewart platform consists of three bodies (upper, middle and lower plates), six servos, other connection materials (horns, legs, rods, bearings, etc.) and control card (myRIO). The model is created to use special solid design tools in SolidWorks. All parts are designed separately and then combined in one study at Figure 4.1.1.

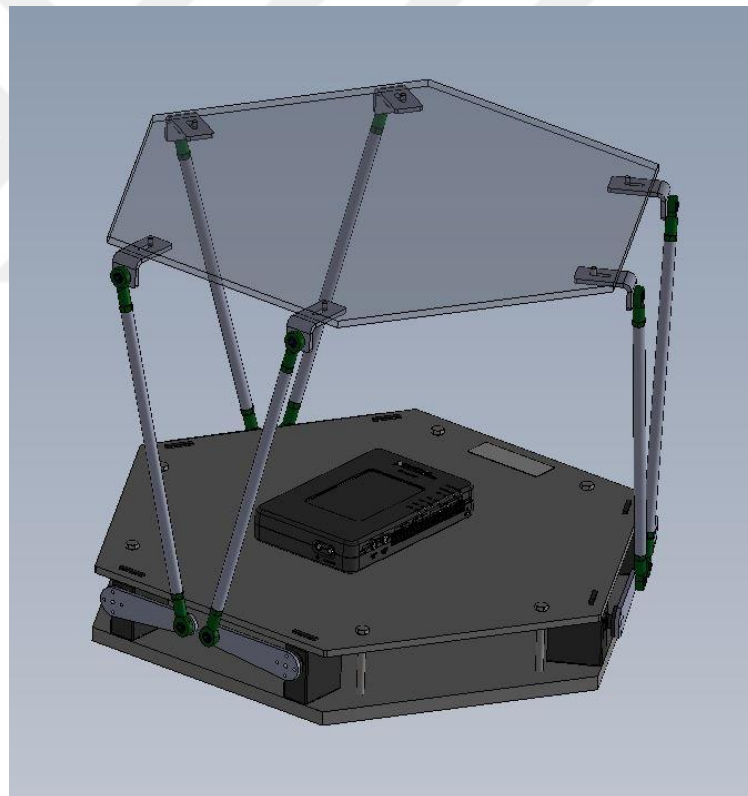


Figure 4.1.1. SolidWorks Model

A cockpit model is also designed with the help of SolidWorks that use the solid modeling technique to provide printable results from the 3D printer in Figure 4.1.2.

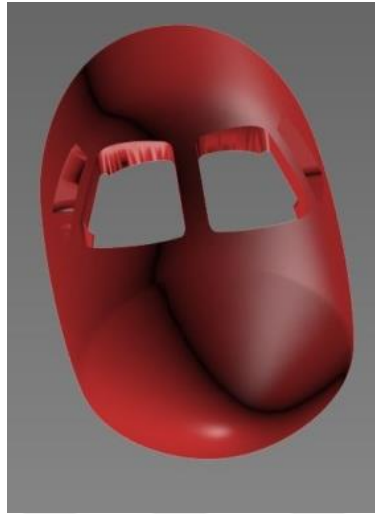


Figure 4.1.2. System Cockpit Model

Three-dimensional printing is one of the many processes used to create a 3D object. Besides 3D printing, additive operations in which successive layers of material are placed in computer control. There are some various polymers in the industry, but the raw material of cockpit model is selected the filament to print the design that is cheap and easy to paint.

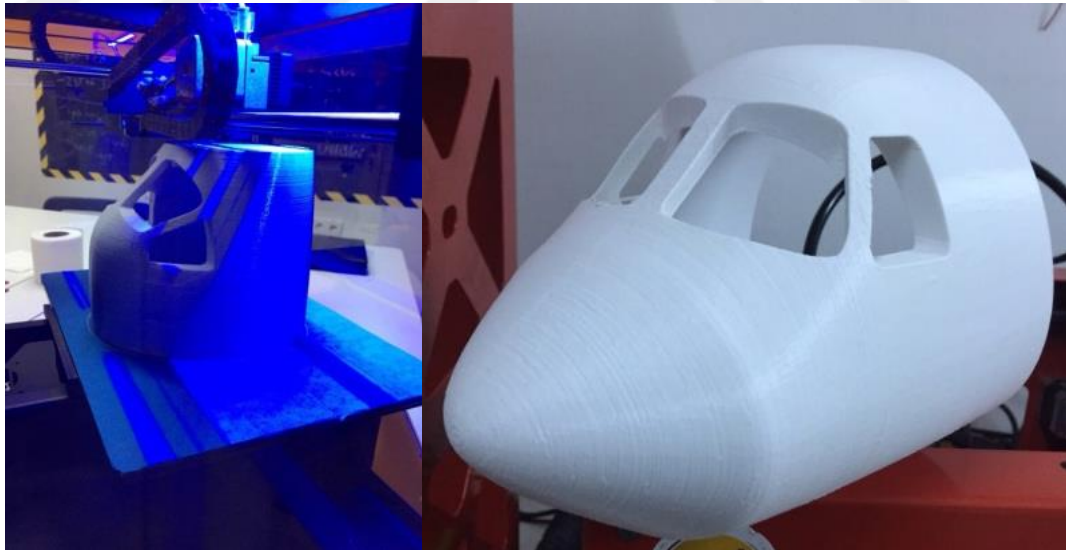


Figure 4.1.3. 3D printed model

The cockpit model is mounted by on the top plate of Stewart platform and the final design of our project is completed as well.

4.2. MATLAB MODEL

The model of Stewart platform is made by MATLAB that is a mathematical language for technical computations. It makes matrix calculations, control system design, linear system analysis and also has too many toolboxes to realize engineering calculations.

In this study, the model of Stewart platform aims to use the robotic toolbox of MATLAB to provide the special motions and rotations (roll/pitch/yaw). There are six rotary points and six spherical joints in the model that is became the standard view of the platform. It helps to realize that is produced the animation of the platform according to the specified inputs.

Beginning of the MATLAB Model, the first point determines the alpha and beta angles and also phase angular position for the model. The angular variables are referred to base node pairs and determined the boundaries of the platform according to the coordinates of x-axis, y-axis and, z-axis. One of the key mark is arranging related coordinates of base points (sin and cos variables) and also coordinates of the platform to specify the position. However, the model is created after assembling the base coordinates to plot the platform. The animation part of the design starts to assign each point as a servo motor and its angle. The angles are changing for each point and joint one by one automatically. It becomes an animation of the Stewart platform that is moving randomly.

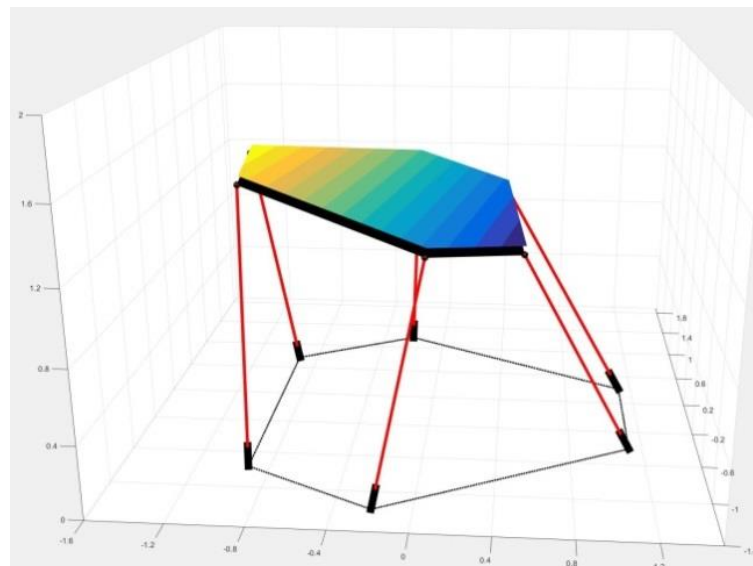


Figure 4.2.1. MATLAB model

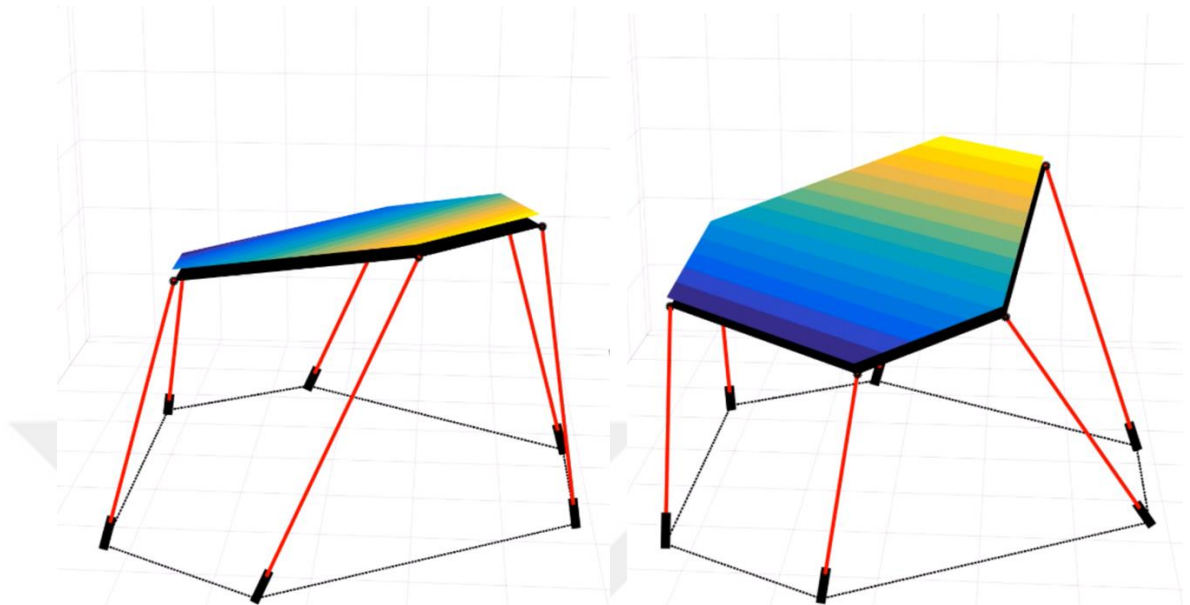


Figure 4.2. 2.. (a) Upward Pitch Motion

(b) Right Side Roll Motion

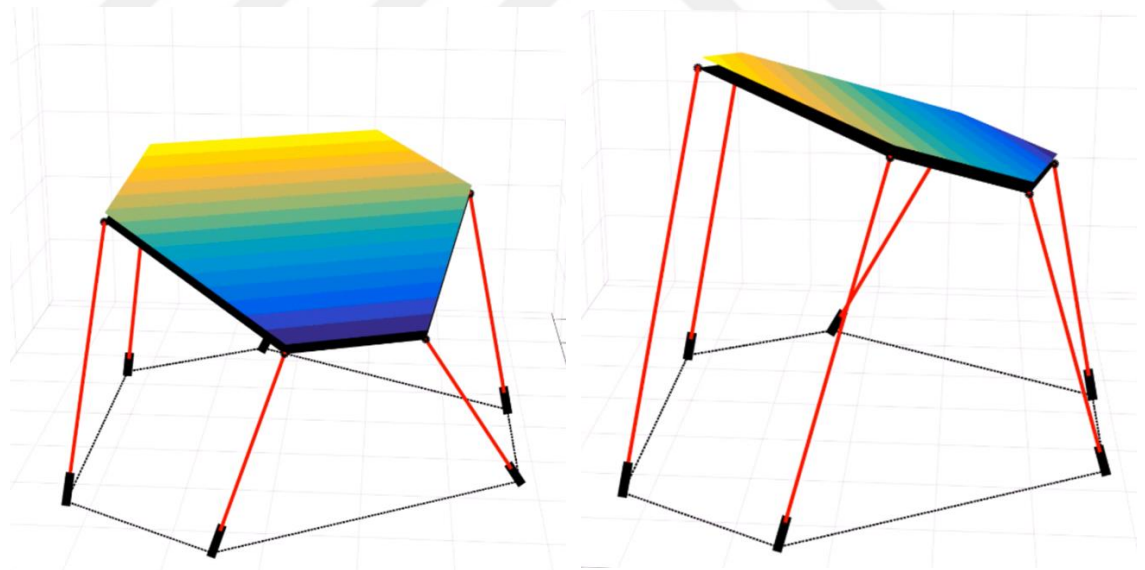


Figure 4,2,3. (a) Downward Pitch Motion

(b) Left Side Roll Motion

4.3. PROCESSING MODEL

One other is made by Processing which is an open source language and combined development environment for the new media art, visual design communities, and electronic arts.

In this study, the model consists of the two plates, six servo horns, six legs and an interface to change the all servo angles of the system which is shown in Figure 4.3.1. Each servo horn provides the motion by connecting the positional and rotational angle rods. These rods have represented the angles of the platform which is potentially feasible. When the control bars moved right or left, the system will start to provide the motion, according to the type of positions (posX, posY, posZ) and rotations (rotX, rotY, rotZ). At the same time, the system calculates the position angle of each rod according to the its motions in the system.

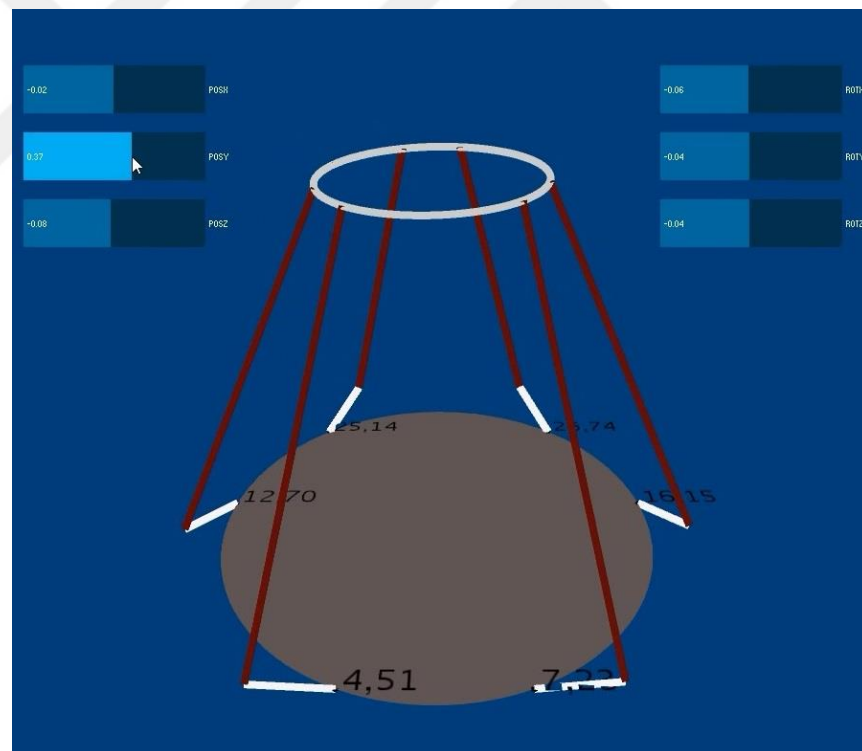


Figure 4.3.1. Processing model

5. CONTROL PANEL

The control panel of the platform is designed based on LabVIEW library that is provided for special motions and rotations from the system. The design of control panel consists of motor angles for each motor by translational movement (x, y, z) and rotational movement (roll, pitch, yaw).

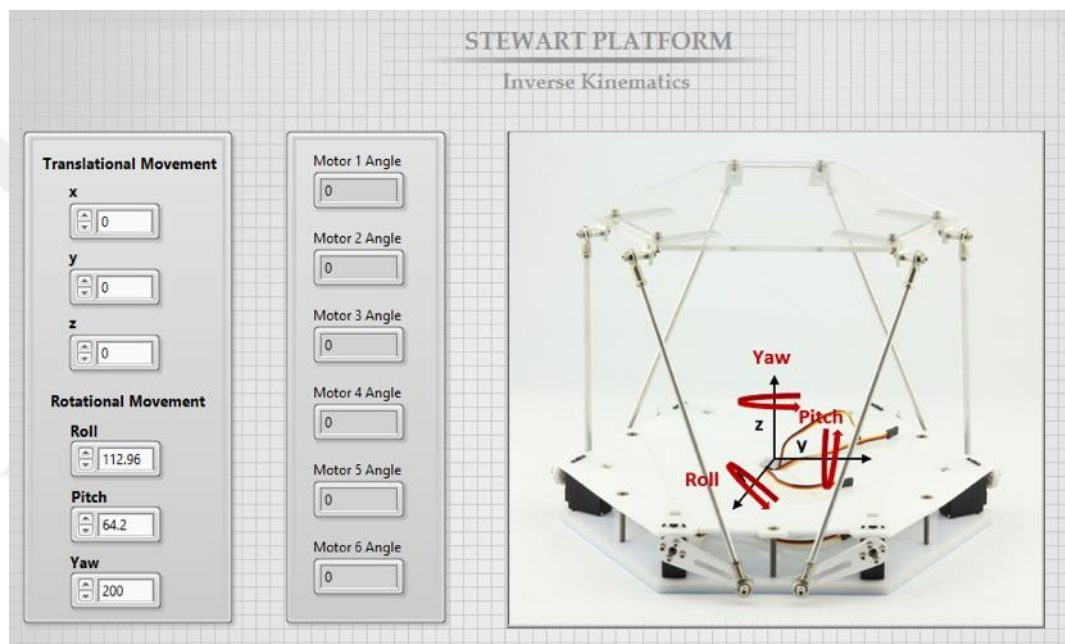


Figure 5.1. LabVIEW Interface Control Panel

The system control is performed by entering the specified angle regarding the coordinates of x, y, and z. The motor angles are calculated automatically for each according to the motion of the platform.

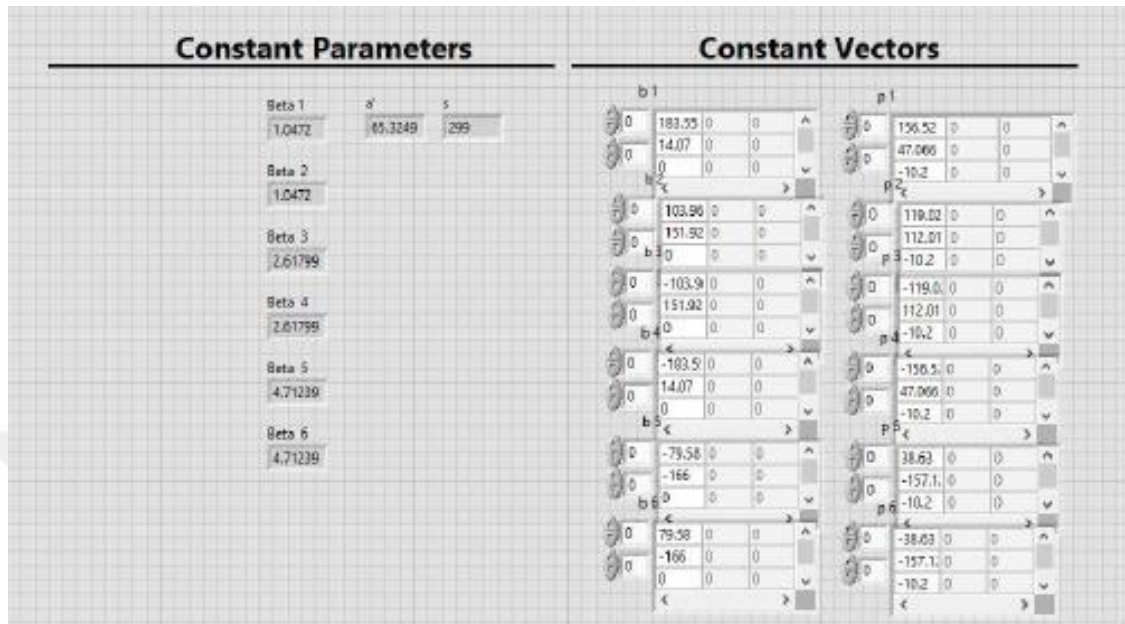


Figure 5.2. Constant Parameters and Vectors of the Platform from Main.vi

The all parameters are entered in arrays according to the real system geometry to design the system as a constant. All angles are calculated in radians as stated in formulas. The angles and parameters are coming from the inverse kinematic calculation of Stewart Platform which are specified inside of Main.vi file as in above Figure 5.2.

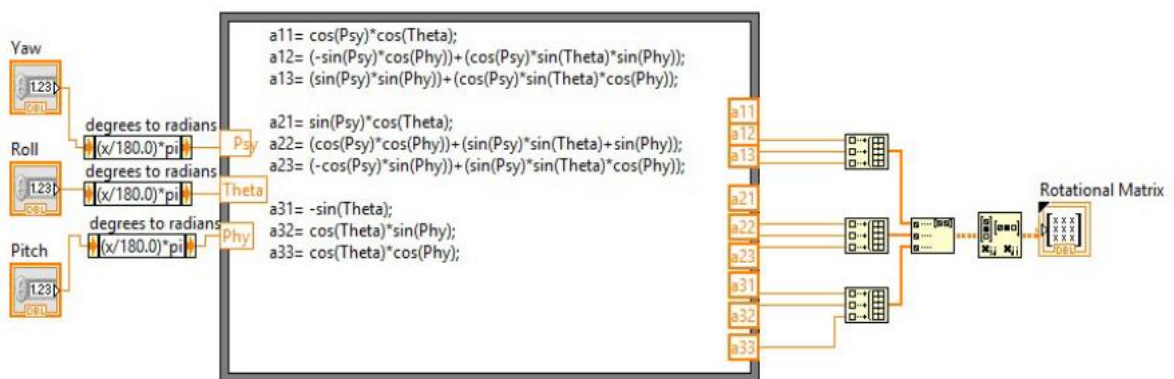


Figure 5.3. Formula Representation as Rotational Matrix

All formula which are specified in calculation section node to obtain entities of the rotation matrix and append them in a matrix in Figure 5.3.

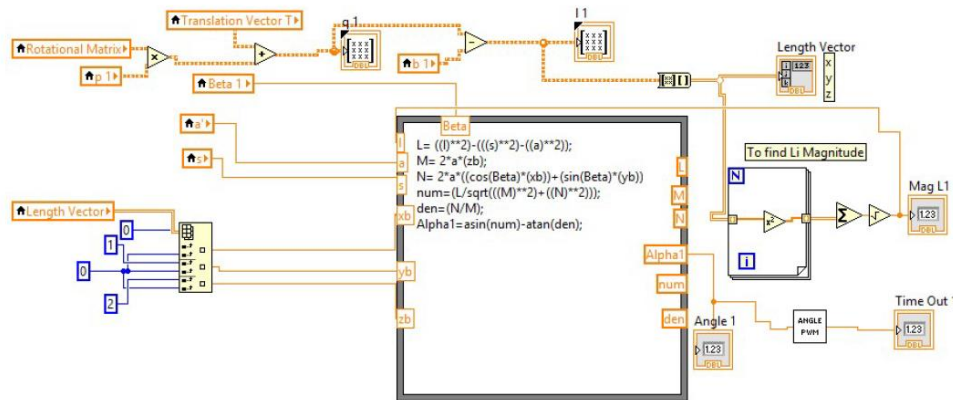


Figure 5.4. Code Section for Servo Motors

Here is the code section to obtain servo angle and hence determine the time required to be sent to the servo motors through myRIO. By the way, calculated motor angles have transformed the Pulse Width Modulation signals by using subVI (Sub-Program). This transformation is stated in the front panel and its' block diagram in Figure 5.5. It means that the system calculates on time for each motor.

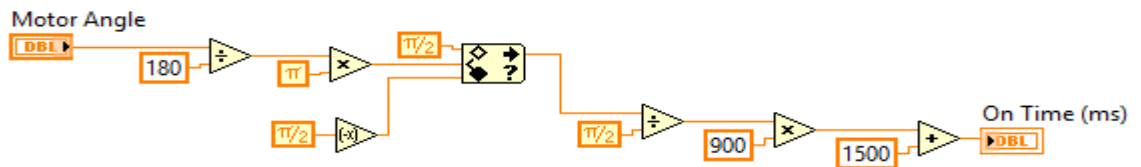


Figure 5.5. Motor Angle Block diagram

The following parameters are entered and the result is shown clearly. Where 257.23 mm is the minimum distance between the base and the platform in the z direction.

X	0	Angle 1	Mag L 1	Time Out 1
Y	0	Angle 2	Mag L 2	Time Out 2
Z	257.23	Angle 3	Mag L 3	Time Out 3
Roll	0	Angle 4	Mag L 4	Time Out 4
Pitch	0	Angle 5	Mag L 5	Time Out 5
Yaw	0	Angle 6	Mag L 6	Time Out 6

Figure 5.6. Platform Angle, Magnitude and TimeOut Table

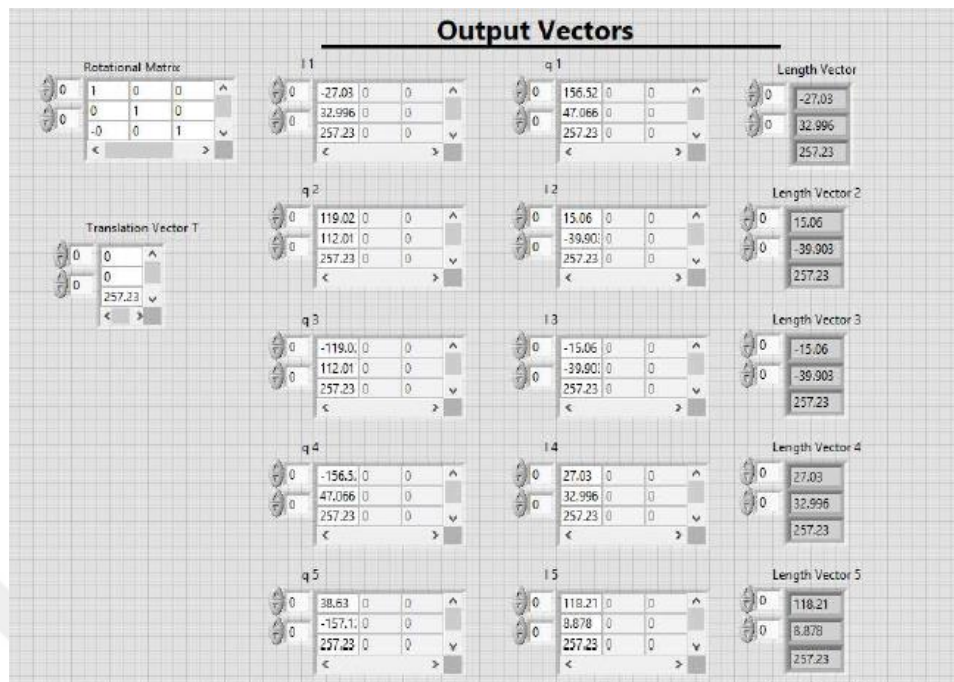


Figure 5.7. Output Vectors

It is the output vectors according to the input parameters, vectors and angles. It shows that, when the rotational matrices and translation vectors are changed as specified, all vector length, platform legs, motors are immediately changed in Figure 5.7.

To apply the desired angles to the servos are applied to myRIO device by the following using an FPGA reference from a bit file that contains indicators of Motor and index (motor's number) and the total time for execution.

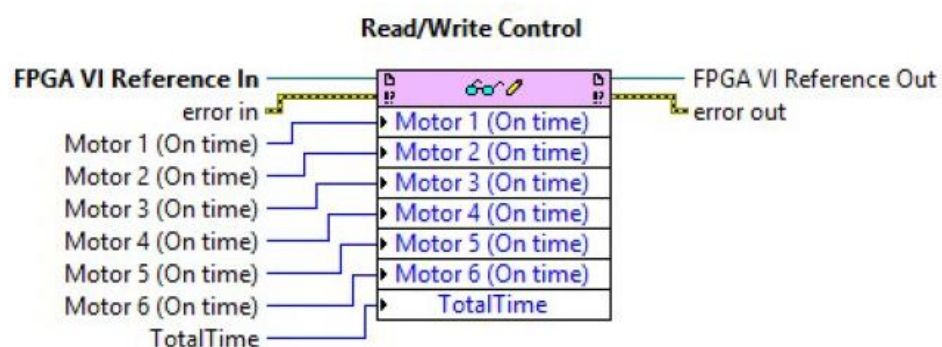


Figure 5.8. Communication Scheme Servos to myRIO.

The system reads the value from FPGA Reference and writes the value to control the FPGA module of myRIO. FPGA Reference value and communication is stated in Figure 5.8.

All motors can be controlled any angles by the control panel in below which can be made only open or close in Figure 5.9.

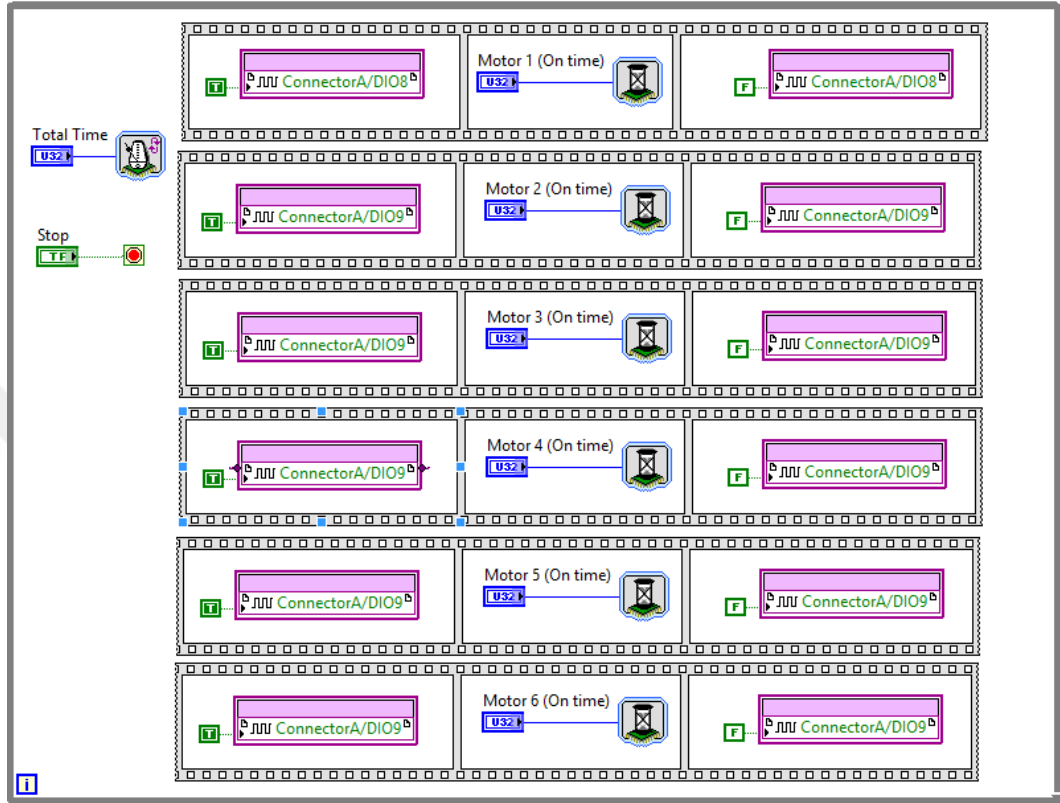


Figure 5.9. General view of control panel with PWM

Motions/Rotations	Upward	Downward	Right Hand	Left Hand
x	0	0	3	-3
y	7	-7	1	-1
z	8	8	4	4
yaw	0	0	15	-15
pitch	30	-30	5	-5
roll	0	0	30	-30

Table 5.1. Motion variables of special rotations

The table 5.1. is shown that the special motions of Stewart platform are provided by control panel, according to the variables of (x, y, z) positions and (yaw, pitch, roll) rotations.



Figure 5.10. (a) Upward Pitch motion



(b) Downward Pitch motion

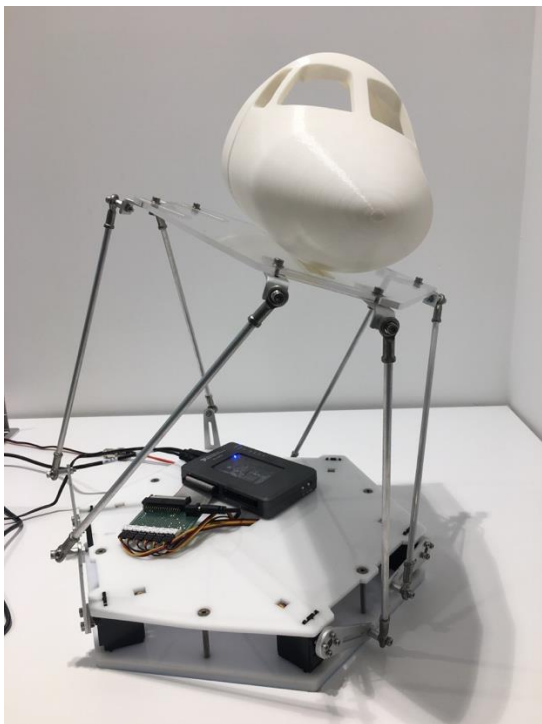
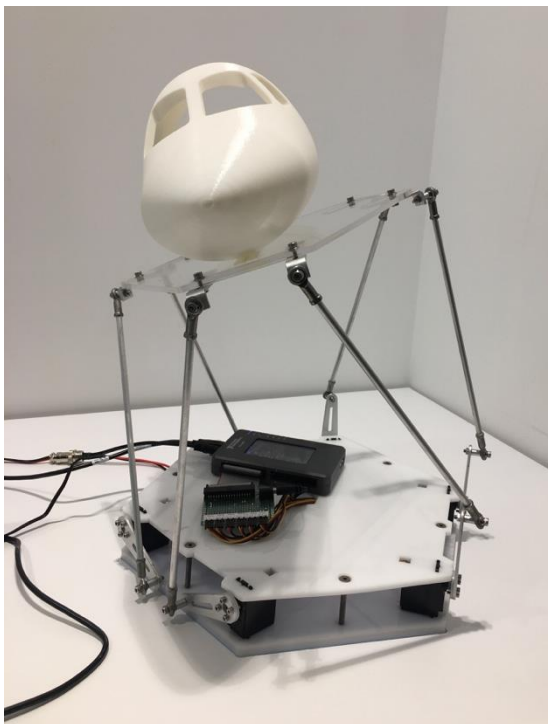


Figure 5.11. (a) Left Side Roll Motion



(b) Right Side Roll Motion

6. CONCLUSION

This thesis presented a prototype of 6-DOF parallel manipulator known as the Stewart Platform with its hardware structure and control principle. It focused on the main function module and realization of the control system software. Stewart Platform is commonly applied in parallel manipulator to perform linear and angular rotations.

In this thesis, dynamic and kinematic analysis of the 6 DOF Stewart Platform was calculated by the formula in the chapter of mathematical design. The system payload is averagely 15kg. Kinematic analysis of the platform was modeled in Processing (IDE), dynamic model of the platform is advanced in MATLAB library that is provided to get proper motions and rotations like upward pitch, downward pitch, left side roll and right side roll motions. The myRIO microcontroller controls with motor shield to each servo motor within the platform regarding the special motions. The control panel of the platform is designed based on LabVIEW library. The system control is performed by entering the specified angles related to the roll, pitch and yaw regarding the coordinates of x , y , and z .

REFERENCES

1. Banas, W., Sekala A., Gwiazda A., Foit K., Kost G., "*Stewart Platform Simulation Using the LabView Environment*", *Achieves of Materials Science and Engineering*, Vol. 75, pp. 82-88, 2015.
2. Szufnarowski, F., "*Stewart Platform with Fixed Rotary Actuators: A Low Cost Design Study*", *Advances in Medical Robotics*, Chapter 4, 1st Ed, 2013.
3. Zhang, B., "*Design and Implementation of a 6 DOF Parallel Manipulator with Passive Force Control*", Doctor of Philosophy Thesis, University of Florida, 2005.
4. Lou, Ji-Hung. and Stephen P. Tseng. "*Developing a Real-Time Serial Servo Motion Control System for Electric Stewart Platform*", 2014, Conference on Advanced Robotics and Intelligent Systems (ARIS).
5. M.A. Facas Vicente. "*A New Artefact, Hexapod Based, for Local Calibration of Coordinate Measuring Machines*", *Arabian Journal for Science and Engineering*, 01/15/2011.
6. Serdar Ay. "*The effect of radius of joint location on workspace analysis of the 6-6 Stewart Platform Mechanism*", 2009 4th International Conference on Recent Advances in Space Technologies, 06/2009.
7. Qu, Zhi Yong, and Zheng Meo Ye. "*Driving and Power Units Design on Hydraulic Stewart Platform*", *Applied Mechanics and Materials*, 2011.
8. T. Brezina. "*Controller Design of the Stewart Platform Linear Actuator*", *Recent Advances in Mechatronics*, 2010.
9. Brezina, L., Andrs O., Brezina, T. "*NI LabView – Matlab SimMechanics Stewart Platform Design*", *Applied and Computational Mechanics* 2, 2008.
10. Bingul, Zafer and Oguzhan Karah. "*Dynamic Modeling and Simulation of Stewart Platform, Serial and Parallel Robot Manipulators - Kinematics, Dynamics, Control and Optimization*", 2012
11. Güneri, B., "*A Complete Dynamic Analysis of Stewart Platform Including Singularity Detection*", Graduate Report, Izmir, 2007

12. Hersan, T., and Ajna R., "Maths of Stewart Platform," [online database], <http://mememememememe.me/>
13. Koszewnik, A., Troc K., Slowik, M., "*PID Controllers Design Applied to Positioning of Ball on the Stewart Platform*", Bialystok University of Technology, Poland, 2014.
14. Hajimirzaalian, H., Moosavi, H., "*Analyzing and Simulating the Inverse and the Direct Dynamics of Parallel Robot Stewart Platform*", Second International Conference on Computer and Network Technology Iran.
15. Burgdörfer, H., Rühle F., "*Building a Stewart-Platform*", Interdisciplinary Center for Scientific Computing Ruprecht-Karls Universität Heidelberg, 2007.
16. Nieuwenhuizen, F.M., Van Paassen M. M., Stroosma O., Mulder, M., Bühlhoff, H. H., "*Cross-platform Validation for a Model of a Low-cost Stewart Platform*", Journal of Modeling Simulation, Identification, and Control Columbia, 2013.
17. Graf, R., Vierling, R., Dillman R., "*A Flexible Controller for a Stewart Platform*", Institute for Realtime-Systems and Robotics University of Karlsruhe, Kaiserstr, 1998.
18. Fond, W. T., "*Design and Testing of a Stewart Platform Augmented Manipulator for Space Applications*", Massachusetts Institute of Technology, 1990.
19. Wang, Xinglong, Bin Wang and Xiaojuan Wu. "*A rapid and high precise calibration method for long-distance cruise missiles*", Aerospace Science and Technology, 2013.

APPENDIX A

Algorithm 4.1. The selection sort algorithm implemented in MATLAB language.

```

%Defining the stewart platform
R=1.2;% Base point radius
r=.9;% Platform radius
length_cyl=.11;% Cylinder length
ALPHA1_X=40; % angle (degree) between base node pairs
BETA1_X=80; % angle btw platform and node points.
PHASE1_X=[0 110 230]; % to definition of the motions of platform.

% node pairs (degrees)
% Specify position and in center of platform.
X1=0;% posX = 0
Y1=0;% posY = 0
Z1=1;% poz = 0

% Specify the angles of tilt
PST=90;% Tilt angle of system
THETA_X=90;% Yaw angle of system
PHT=90;% Tilt axis angle from y-axis
animation=1;% 1=Animation enabled
camtype=3;% 3=Third Person View
x1_min=-2.6; x1_max=2.5; %Boundries for x coordinate
y1_min=-2.4; y1_max=2.9; %Boundries for y coordinate
z1_min=0; z1_max=2; % Boundries for z coordinate

%to define base points coordinates
AX_1=[R*cosa(PHASE1_X(1)+ALPHA1_X/2),R*sina(PHASE1_X (1)+ALPHA1_X/2),0];
AX_2=[R*cosa(PHASE1_X (1)-ALPHA1_X/2),R*sina(PHASE1_X (1)-ALPHA1_X/2),0];
BX_1=[R*cosa(PHASE1_X (2)+ALPHA1_X/2),R*sina(PHASE1_X (2)+ALPHA1_X/2),0];
BX_2=[R*cosa(PHASE1_X (2)-ALPHA1_X/2),R*sina(PHASE1_X (2)-ALPHA1_X/2),0];
CX_1=[R*cosa(PHASE1_X (3)+ALPHA1_X/2),R*sina(PHASE1_X (3)+ALPHA1_X/2),0];
CX_2=[R*cosa(PHASE1_X (3)-ALPHA1_X/2),R*sina(PHASE1_X (3)-ALPHA1_X/2),0];

% animation part
ctrz=1;
for index=0:60:3000
ctrz=ctrz+1;
% Animation switch
switch animation
case 1

```

```

%Defining animation dynamics
PST=40*sina(index*.5);
THETA_X=10*sina(index*.5);
PHT=index*1-10;
Z1=.2*sina(index)+1.4;
X1=.2*cosa(index);
Y1=.2*sina(index);
otherwise
end

%Determine the platform coordinates
index=1;
for ii=1:3
for jj=1:2
if jj==1

index=index+2;
X_1=r*cosa(-BETA1_X/2+THETA_X+PHASE1_X(ii));
Y_1=r*sina(-BETA1_X/2+THETA_X+PHASE1_X(ii));
else

index=index+2;
X_1=r*cosa(BETA1_X/2+THETA_X+PHASE1_X(ii));
Y_1=r*sina(BETA1_X/2+THETA_X+PHASE1_X(ii));
end
A_1=[cosa(-PHT) -sina(-PHT);sina(-PHT) cosa(-PHT)]*[X_1;Y_1];
X_2=A(1,1)*cosa(PST);
Y_2=A(2,1);
B_1=[cosa(PHT) -sina(PHT);sina(PHT) cosa(PHT)]*[X_2;Y_2];
Dx1=B(2,2);
Dy1=B(3,2);
Dz1=A(3,3)*sina(PST);
PLATFORM(index,:)= [x1+dx1,y1+dy1,z1+dz1];
end

end

%Determine the base coordinates to plot platform
BASE=[AX_1(0),AX_1(1),AX_1(3);
AX_1(1),AX_1(2),AX_1(3);
BX_2(1),BX_2(2),BX_2(3);
BX_1(1),BX_1(2),BX_1(3);
CX_2(1),CX_2(2),CX_2(3);
CX_1(1),CX_1(2),CX_1(3);
AX_2(1),AX_2(2),AX_2(3)];

%Top Camera view options
VX_1=PLATFORM(1,:)-[x y z];
VX_2=PLATFORM(2,:)-[x y z];

```

```

%The top camera position and type.
switch camtype
case 1

%CTRZ Cam Setting Pos
set(gca,'CamUPSTVec', normal, 'Projected ', 'view','CamAngle',60)
camoffset_x1=4*(VX_1+VX_2);
campos(normal/1.5+[x y z]-camoffset_x)

%The Settings of Plot
clf,hold on
colormap(parula(15)) %Platform Color
material shiny % Material & Lighting Part.
dzaspect([1 1 1]) %Axis settings
axis([x1_min x1_max y1_min y1_max z1_min z1_max])
axis vis3d

case 3 %Out of View
viewx ([2,2,2])
set(gca,'Projected','view','CamAngle',60)
positionx=5*[.2 -2.3 .8];
campos(positionx)
camtarget([1 1 .8])
camzoom(2.3) % zoom of camera

%Background of the display inside sphere
[X1,Y1,Z1]=sphere(200);
scale=20;
X1=X1*scale; Y1=Y1*scale; Z1=Z1*scale;
surface(X1,Y1,flipud(Z1),bg1,'FaceColor','texturemap',
'EdgeColor','direct','CDataMapping','none');

%to Set the properties of axis.
grid on
set(gca,'X1Tick',[x1_min:4:x1_max],...
'Y1Tick',[y1_min:4:y1_max],...
'Z1Tick',[z1_min:4:z1_max],...
'visible','off','color','w') end

%Plot the actuators for platform and base
for ii=1:6
MAG_X=norm(PLATFORM_X(ii,:)-BASE_X(ii,:));
unit=length_cyl*(PLATFORM_X(i,:)-BASE_X(i,:))/MAG_X+BASE_X(i,:);
cyl=[BASE_X(ii,:);unit];
ARM=[unit;PLATFORM_X(ii,:)];
plot3(ARM(:,1),ARM(:,2),ARM(:,3),'r','linewidth',4)
plot3(cyl(:,1),cyl(:,2),cyl(:,3),'k','linewidth',10)
end

```

```

%Plot the fixed and movable platform.
plot3(BASE_X(:,1),BASE_X(:,2),BASE_X(:,2),'ok','linewidth',3)
plot3(PLATFORM_X(:,1),PLATFORM_X(:,2)
,PLATFORM_X(:,3),'ok','linewidth',4)
patch(PLATFORM_X(:,1),PLATFORM_X(:,3)
,PLATFORM_X(:,2),'k','edgecolor','k')
PLATFORM2=PLATFORM_X+ones(6,1)*normal/20;
patch(PLATFORMX_2(:,1),PLATFORMX_2(:,3)
,PLATFORMX_2(:,2),PLATFORMX_(:,2),'edgecolor','none')
pause(.111) %to pause the animation of the platform
% arranged the frame
end
void swap1 (int *x_1,int *y_1)
{
    int temp;
    temp = *x_1;
    *x_1 = *y_1;
    *y_1 = temp; }
void selection_sort(int list[], int n)
{ int i, j, min;
for (ii = 0; ii < n - 1; ii++)
    {
        min = ii;
        for (jj = ii+1; jj < n; jj++)
            {if (list[jj] < list[min1])
                {min1 = jj;
                }
            }
        swap1 (&list[ii], &list[min1]);
    }
} void printlist(int list1[],int n)
{
    int ii;
    for(ii=0;ii<n;ii++)
        printf("%c\t",list1[ii]);
}
void main()
{
    const int MX_ELEMENTS_1 = 20;
    int list1[MX_ELEMENTS_1];
    int ii = 0;
    // arrange values randomly
    for(ii = 0; i < MX_ELEMENTS_1; ii++) {
        list1[i] = rand();
    }
    printlist(list1,MX_ELEMENTS1);
}
}
}
}
}

```

APPENDIX B

Algorithm 4.2. The selection sort algorithm implemented in Processing (IDE) language.

```

%Specifying the stewart platform
import peasy.*;//import to peasy library
import controlP6.*; // import to controlp5 library
import oscP6.*; // import to oscp5 library
import netP6.*; // import to netp5 library
float MAX_TRANSLATIONAL = 70; // translation angle
float MAX_ROTATIONAL = PR/7; // rotation angle
ControlP5 cp5; //controlling the P5 system
PeasyCam camera; //configure camera settings
Platform mPlatform; //call the platform class
OscP5 oscP5;
NetAddress mOscOut; // address of the pi connected to motors
float posX1=0, posY1=0, posZ1=0, rotX1=0, rotY1=0, rotZ1=0;
//all translation and rotation angles
void setup() { // pixel value

size(1152, 864, P3D);
smooth();
mOscOut = new NetAddress("Stewart_Platform.local", 8888);
textSize(25); // text size of angles
camera = new PeasyCam(this, 666);
camera.setRotations(-1.0, 0.0, 0.0);
camera.lookAt(8.0, -50.0, 80.0);
mPlatform = new Platform(1.3);
mPlatform. applyTranslationalAndRotationalMotion (new PSVector()
, new PSVector());

.setSize(190, 50).setRange(-2, 2);
cp5.addSlider("posY1") //posy control panel position
.setPosition(20, 130)
.setSize(190, 50).setRange(-2, 2);
cp5.addSlider("posZ1") //posz control panel position
.setPosition(20, 200)
.setSize(190, 50).setRange(-2, 2);
cp5.addSlider("rotX1") //rotx control panel position
.setPosition(width-220, 60)
.setSize(190, 50).setRange(-2, 2);
cp5.addSlider("rotY1") //roty control panel position
.setPosition(width-220, 130)
.setSize(190, 50).setRange(-2, 2);

```

```

cp5.addSlider("rotZ1") //rotz control panel position
.setPosition(width-220, 200)
.setSize(190, 50).setRange(-2, 2);
cp5.setAutoDraw(false);
camera.setActive(true); }

//camera position and drawing the platform and its rot and pos.
void draw() { background(16506);
mPlatform.applyTranslationalAndRotationalMotion
(PSVector.multx(new PSVector(posX1, posY1, posZ1), MAX_TRANSLATIONAL),
PSVector.multx(new PSVector(rotX1, rotY1, rotZ1), MAX_ROTATIONAL));
mPlatform.draw();

hint(DISABLE_DEPTH_TEST); // configurate camera position
camera.beginHUD();
cp5.draw();
camera.endHUD();
hint(ENABLE_DEPTH_TEST); }
//controlling the event of camera.
void controlEvent(ControlEvent theEvent) {
camera.setActive(false);
// to send a OSC package
float[] angles = mPlatform.getAlpha();
for (float f : angles) {
if(Float.isNaN(x)){
return; } } }
//OSC message to add an integer array
OscMessage myMessage = new OscMessage("/Angles");
myMessage.add(angles);
oscP5.flush(myMessage, mOscOut); }
//Use Mouse to control whole system
void mouseReleased() {
camera.setActive(true); }
//Use Space for reset platform
void keyPressed() {
if (key == ' ') {
camera.setRotations(-1.0, 0.0, 0.0);
camera.lookAt(8.0, -50.0, 80.0);
camera.setDistance(666); } }
CLASS OF PLATFORM
class Platform {

private PSVector translation, rotation, initialHeight;
//define the translation, rotation and
initial height

```

```

private PSVector[] baseJointx, platformJointx, q, l, A;
//define the base and platform joints
private float[] alpha;
//define angles
private float baseRadiusx, platformRadiusx, hornLengthx, legLengthx;
// define length of horn and leg ,
//implement base and platform radius // Real Angles
private final float baseAnglesx[] = { // base platform angles
308.5, 351.5, 68.5, 111.5, 188.5, 231.5 };
private final float platformAngles[] = { //platform angles
286.10, 13.9, 46.1, 133.9, 166.1, 253.9};
private final float beta[] = { // rods angles
-8*PR/3, PR/3, 0, -PR, -4*PR/3, -7*PR/3};

// Real Measurements of leg and horn length
private final float SCALE_INITIAL_HEIGHT = 240; //base initial height
private final float SCALE_BASE_RADIUS = 140; //base radius
private final float SCALE_PLATFORM_RADIUS = 40; // p radius
private final float SCALE_HORN_LENGTH = 30; // horn length
private final float SCALE_LEG_LENGTH = 240; //leg length
public Platform(float s) {
translation = new PSVector(); //translation vector
initialHeight = new PSVector(0, 0, s*SCALE_INITIAL_HEIGHT);
//initial height
rotation = new PSVector(); //rotation vector
baseJointx = new PSVector[6]; // base joint

platformJointx = new PSVector[6]; // platform joint
alpha = new float[6]; // angle of platform
Q1 = new PSVector[6];
L1 = new PSVector[6];
A1 = new PSVector[6];
baseRadiusx = s*SCALE_BASE_RADIUS_X; //base radius equation
platformRadiusx = s*SCALE_PLATFORM_RADIUS_X; //platform radius equation
hornLengthx = s*SCALE_HORN_LENGTH_X; // length of horn equation
legLengthx = s*SCALE_LEG_LENGTH_X; // length of leg equation
for (int i=0; i<6; i++) { // implement base Angles
float ma = baseRadiusx*cos(radians(baseAnglesx[i]));
float mb = baseRadiusx*sin(radians(baseAnglesx[i]));
baseJointx[i] = new PSVector(ma, mb, 0); }
for (int i=0; i<6; i++) { // implement platform Angles
float ma = platformRadiusx*cos(radians(platformAnglesx[i]));
float mb = platformRadiusx*sin(radians(platformAnglesx[i]));
platformJointx[i] = new PSVector(ma, mb, 0); //platform's joint variables

```



```

Q1[i] = new PSVector(0, 0, 0);
L1[i] = new PSVector(0, 0, 0);
A1[i] = new PSVector(0, 0, 0); }
calcQ(); } public void applyTranslationalAndRotationalMotion
(PSVector t, PSVector r){
// apply translation and rotation
rotation.set(r);
translation.set(t);
calcQ();
calcAlpha(); }
private void calcQ() {
for (int i=0; i<6; i++) {
// rotation variables of platform and angle equations for each coordinate

Q1[i].x = cos(rotation1.z)*cos(rotation1.y)*platformJointx[i].x + (-sin(rotation1.z)
*cos(rotation1.x)+cos(rotation1.z)*sin(rotation1.y)
*sin(rotation1.x))*platformJointx[i].y
+ (sin(rotation1.z)*sin(rotation1.x)+cos(rotation1.z)
*sin(rotation1.y)*cos(rotation1.x))
*platformJointx[i].z;

Q1[i].y = sin(rotation1.z)*cos(rotation1.y)*platformJointx[i].x
+(cos(rotation1.z)*cos(rotation1.x)
+sin(rotation1.z)*sin(rotation1.y)*sin(rotation1.x))
*platformJointx[i].y +(cos(rotation1.z)
*sin(rotation1.x)+sin(rotation1.z)*sin(rotation1.y)
*cos(rotation1.x)) *platformJointx[i].z;
Q1[i].z = -sin(rotation1.y)*platformJointx[i].x
+cos(rotation1.y)*sin(rotation1.x)*platform
Jointx[i].y +cos(rotation1.y)*cos(rotation1.x)
*platformJointx[i].z; translation variables
Q1[i].add(PSVector.add(translationx, initialHeighta));
L1[i] = PSVector.sub(q[i], baseJointx[i]); } }
private void calcAlpha() {
// leg length, horn length and base
// joint calculations and angles.

for (int i=0; i<6; i++) {
float L1 = l[i].magSq()-(legLengthx*legLengthx)
+(hornLengthx*hornLengthx);
float M = 3*hornLengthx*(Q1[i].z-baseJointx[i].z);
float N = 3*hornLengthx*(cos(beta[i])*( Q1[i].x-baseJointx[i].x) +
sin(beta[i])*( Q1[i].y-baseJointx[i].y));
alpha[i] = asin(L/sqrt(M*M+N*N)) - atan2(N, M);
A1[i].set(hornLengthx*cos(alpha[i])*cos(beta[i]) + baseJointx[i].x,
hornLengthx*cos(alpha[i])*sin(beta[i]) + baseJointx[i].y,
hornLengthx*sin(alpha[i]) + baseJointx[i].z);
float xqxb = (Q1[i].x-baseJointx[i].x);
float yqyb = (Q1[i].y-baseJointx[i].y);

```

```

float h1 = sqrt((legLengthx*legLengthx)+(hornLengthx*hornLengthx) -
(xqxb*xqxb)-(yqyb*yqyb)) - Q1[i].z;
float L2 = 3*hornLengthx*hornLengthx;
float M1 = 3*hornLengthx*(h0+q[i].z);

float a0 = asin(L0/sqrt(M0*M0+N*N)) - atan2(N, M0); } }

public void draw() { // Drawing Base Platform
noStroke();
fill(95,84,84);
ellipse(0, 0, 3*baseRadiusx, 3*baseRadiusx);
for (int i=0; i<6; i++) {
pushMatrix();
translate(baseJointx[i].x1, baseJointx[i].y1, baseJointx[i].z1);
noStroke();
fill(0);
ellipse(0, 0, 5, 5);
text(String.format("%.2f", degrees(alpha[i])), 5,5,5);
popMatrix();
stroke(245);
line(baseJointx[i].x, baseJointx[i].y, baseJointx[i].z, A1[i].x1,
A1[i].y1, A1[i].z1);
PSVector rod = PSVector.sub(q[i], A[i]);
rod.setMag(legLengthx);
rod.add(A1[i]);
stroke(97,10,10);
strokeWeight(8);
line(A1[i].x1, A1[i].y1, A1[i].z1, rod.x1, rod.y1, rod.z1); }

for (int i=0; i<6; i++) { // drawing joints and rods together
pushMatrix();
translate(Q1[i].x1, Q1[i].y1, Q1[i].z1);

noStroke();
fill(0);
ellipse(0, 0, 4, 4);
popMatrix();

line(baseJointx[i].x, baseJointx[i].y, baseJointx[i].z,
Q1[i].x, q[i].y, q[i].z); }
pushMatrix();//check the platform height
translate(initialHeighta.x, initialHeighta.y, initialHeighta.z);
translate(translation1.x, translation1.y, translation1.z);
rotateZ(rotation1.z);
rotateY(rotation1.y);
rotateX(rotation1.x);
stroke(200);
noFill();
ellipse(0, 0, 3*platformRadiusx, 3*platformRadiusx);
JpopMatrix(); } }

```