The Stewart Platform

A Proposed Ocean Surface Simulation Platform

University of Cape Town
Dept. of Electrical and Electronics Engineering

Presented by:
Liam James Clark

Prepared for:
J.C. Pead

Submitted to the Department of Electrical and Electronics Engineering at the University of Cape Town in partial fulfilment of the academic requirements for a Bachelor of Science degree in Electrical and Computer Engineering.

October 22, 2018

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the work(s) of other people has been attributed, and has been cited and referenced.
3. This report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

Signature: 
Liam James Clark

Date: October 22, 2018

Acknowledgments

A large amount of time and effort has gone into making this project a success and without the help and advice of others, this would not have been possible.

Firstly, I would like to thank my supervisor, Justin Pead, for the many hours put into helping me with the design and building of the project. His guidance and expertise in both the mechanical and electrical aspects of the project and throughout my studies at UCT have been invaluable.

I would like to thank Robyn Verrinder for her advice and input into the project as well as the enthusiasm always shown when discussing any engineering problems.

I would like to thank Brendon Daniels for his help in laser cutting the parts for my project and going out of his way to get things done timeously.

I would like to thank Jason Meek and Andrew Olivier for their friendship during the project and our engineering degrees. I have thoroughly enjoyed our many hours working together, even though they may have not always been productive. It just wouldn't have been the same without the banter and good company.

I would like to thank Sarah Lawson for all the support during the project and making the time and effort to read through many different iterations of my report. I really appreciate all the advice and help in adding the finishing touches to my report.

Finally, I would like to thank my family who have made it possible for me to study and complete my degree and I am grateful for the support which they have always shown towards any interests of mine.

Abstract

Wave Gliders are a type of autonomous vehicle which operate on the ocean surface and are often used to perform research out at sea. Researchers using these vehicles want to be able to test them before deployment for long periods of time. Due to the cost and time associated with testing these vehicles in the ocean, it is necessary that a hardware-in-the-loop simulator be created which enables researchers to subject the Wave Glider to ocean-like conditions in a laboratory environment.

This project proposes the use of a Stewart platform for use as an ocean surface simulation platform. This is due to the Stewart platform being capable of moving an end-effector platform with six degrees of freedom. This is ideal for ocean simulation as it allows an object mounted to the platform to be subjected to movement along the x, y and z-axes as well as roll, pitch and yaw rotations.

In this project, a detailed design of the proposed platform was performed with the platform ultimately being prototyped and tested.

Contents

List of Figures

List of Tables

Nomenclature

ASV Autonomous Surface Vehicle

DOF Degrees of Freedom

HAL Hardware Abstraction Layer

HIL Hardware-in-the-Loop

IMU Inertial Measurement Unit

PLA Polylactic Acid

Chapter 1

Introduction

1.1 Background to the study

Wave Gliders are autonomous vehicles which operate on the ocean surface. They are known for their unique propulsion mechanism which harnesses energy from the motion of waves shown in Fig. 1.1. Wave Gliders fall under a family of vehicles known as autonomous surface vehicles (ASVs) and the Wave Glider's ability to harness energy from waves has enabled it to be deployed out at sea for months at a time. Wave Gliders are often fitted with a multitude of sensors making them incredibly powerful tools for performing ocean research. They are much cheaper to deploy than research vessels and also do not require a crew.

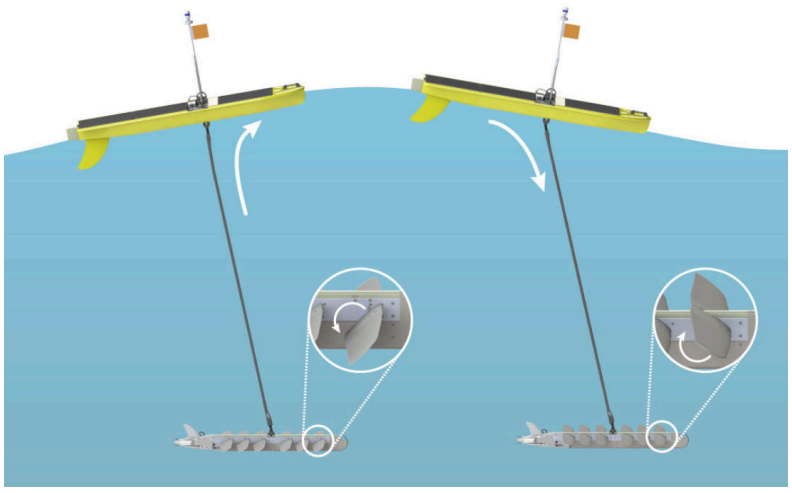

Figure 1.1: A diagram showing the wave glider and its propulsion system [1]

One difficulty in using Wave Gliders for research is around planning routes for them to follow. This is a challenging task due to the nature of their propulsion system. In order to design and test the Wave Glider system, several key systems must work together correctly. Examples of these systems include accurate models of sea conditions, dynamic models which accurately predict the interactions the Wave Glider will have with the sea and the accurate calibration of sensor systems. The University of Cape Town currently has researchers working on several of these systems and is in need of a hardware-in-the-loop (HIL) simulation platform which would be able to mimic the accelerations experienced by the Wave Gliders in the ocean.

1.2 Objectives of this study

1.2.1 Problems to be investigated

This study involves designing and testing a prototype of a Wave Glider HIL simulation platform. This prototype is based on a Stewart platform with the intention of being able to simulate ocean waves. The prototype will not be a full scale platform as there are several limitations such as budget and time for this project. The main question to be investigated is whether or not a Stewart platform is viable for such use. In testing the Stewart platform, it will be required that sensors mounted on the platform measure its positions and orientations and allow for the platform's outputs to be compared to corresponding inputs.

1.2.2 Purpose of the study

The study aims to see if a Stewart platform can be used as a viable simulation platform for a Wave Glider. The testing of embedded systems is most accurate when tested in real environments. In the case of the Wave Glider, this means testing in the ocean. Unfortunately testing the Wave Glider in the ocean is not feasible due to the cost of setting up experiments, waiting for specific conditions for tests, non-repeatability due to the stochastic nature of the ocean, a long development-testing turnaround time and the inability to take lab equipment out to sea. For this reason it is necessary that a realistic HIL simulator be created to emulate the ocean environment for testing. This will mean being able to simulate the necessary movements such that an IMU mounted on the Stewart platform would experience realistic wave motions.

1.3 Scope and Limitations

The project entails the design of a Stewart platform. Many commercially available Stewart platforms are machined out of materials like aluminium. For this project, the materials needed to be cheap and readily available due to the time constraints. The university provided 3D printing and laser cutting services for free, so the design was focused around using materials such as perspex, hardboard and PLA plastic.

Commercial platforms are also often pneumatically actuated. Due to this being an electrical engineering project, the choice of actuators was limited to electrical actuators.

There was a budget of R1600.00 for the project which could be used to purchase items not currently owned by the university. The project also spanned a 12 week semester. This meant that using components which were already owned by the university was preferable as it kept the cost of the project down and meant that time did not have to be wasted in waiting for components to be shipped.

1.4 Plan of development

This report follows the development life-cycle of a Stewart platform. Firstly, a literature review was performed to obtain an understanding of oceans and the mathematics that describe them. Stewart platforms and the literature around them is discussed, highlighting areas which are of relevance to the project.

Following the literature review, the methodology chapter documents the process of planning, creating specifications, designing and prototyping the platform. This chapter gives a detailed description of how the platform was designed and built.

A results chapter follows in which the Stewart platform prototype is tested. This chapter describes how tests were set up, what they aimed to measure and how the prototyped platform performed in these tests. At the end of the chapter there is an analysis of the results.

A conclusions chapter shows the verification process of the Stewart platform and contains concluding remarks on the project. Finally, recommendations are made regarding future work that could be performed.

Chapter 2

Literature Review

2.1 Ocean Modelling

2.1.1 Introduction

In a historical review on the study of ocean surfaces, Mitsuyasu [8] accredits the first pioneering study in the field to Sverdrup and Munk [9]. Sverdrup and Munk had worked on forecasting techniques for wind waves and swell for the United States Navy during the Second World War and the theories developed during this time were published in 1947. Their paper discusses some of the fundamental principles of wave surfaces as well as theory behind observations on the growth and decay of waves. Sverdrup and Munk also made progress towards understanding the transfer of wind energy into wave energy which built on previous work by Jeffreys [10]. Jeffreys had discussed how wind blowing over a wave created pressure differences resulting in work being done by the wind and energy being transferred to the wave.

Holthuijsen [11] describes in his book, "Waves in Oceanic and Coastal Waters", how waves are formed. As wind blows across the ocean's surface, perturbations at the interface between the air and the water cause small waves to form (the cause of which is still debated). The wind interacts with the wave's surface inducing a pressure difference between the windward facing and non-windward facing sides resulting in energy being transferred from the wind to the water. This process causes the wave's size to increase, leading to a positive feedback effect where the interaction between the wind and water's surface perpetuate the continual growth of the wave. The longest distance across an area

of the water's surface where it is exposed to wind is known as the fetch, shown in Fig. 2.1, and is where the growth of wind waves occur.

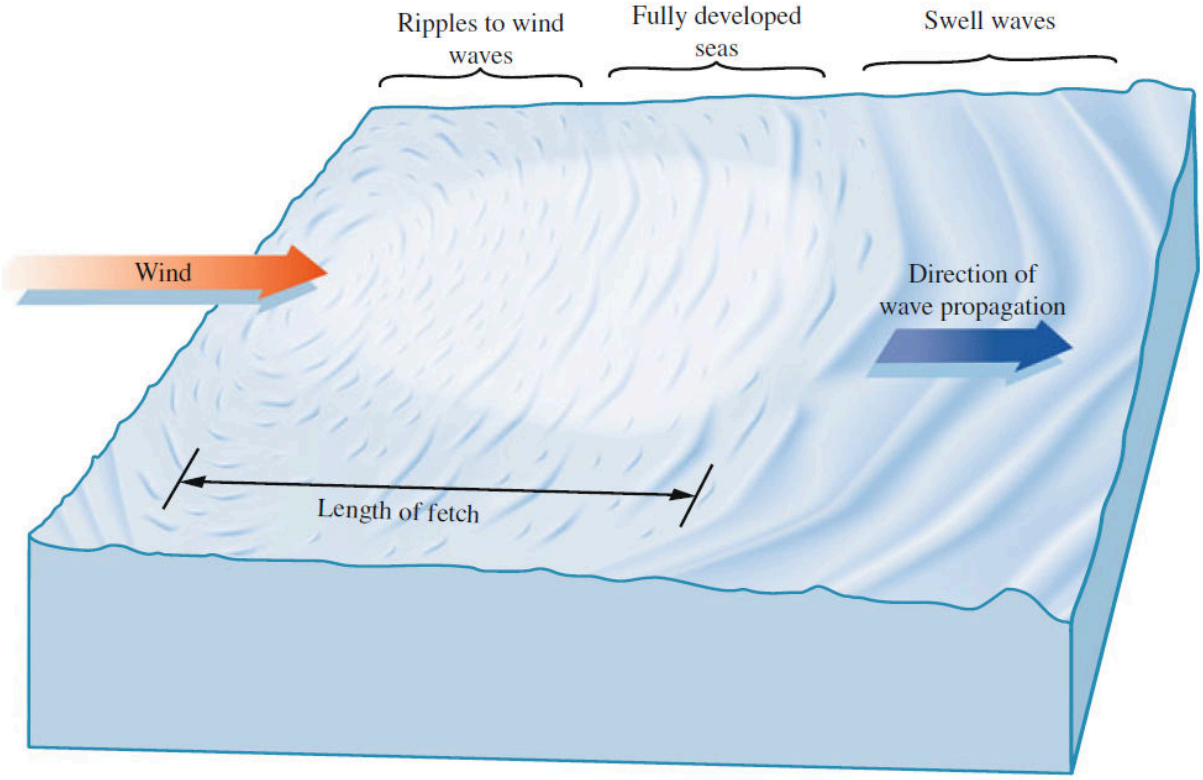

Figure 2.1: A diagram showing the development of sea state due to wind [2]

This growth phase of waves is called the transient state. As the waves grow, they approach a specific speed and wave height. When waves have stopped growing and their speed is constant, the sea is said to be fully-developed or in steady state. Due to irregularities when measuring waves, Sverdrup and Munk proposed a measure of significant wave height which was determined by the mean of the highest third of the waves. A measure of significant wave period could then be determined from the mean period of these waves.

Sverdrup and Munk show that in water where the depth is constant, the wave velocity can be represented by Eq. 2.1 where, v is the wave velocity, g is acceleration due to gravity, λ is the wave's wavelength and h is the depth of the water. The majority of their research at the time focused on deep water waves.

$$v = \sqrt{\frac{g\lambda}{2\pi} \tanh\left(2\pi \frac{h}{\lambda}\right)} \tag{2.1}$$

Deep water waves were categorised as per Eq. 2.2 and shallow water waves as per Eq. 2.3. This allowed for simplifications to be made to Eq. 2.1 such that the speed of a deep water wave can be closely approximated by Eq. 2.4 and a shallow wave by Eq. 2.5.

$$h > \frac{1}{2}\lambda \tag{2.2}$$

$$h < \frac{1}{25}\lambda \tag{2.3}$$

$$v \approx \sqrt{\frac{g\lambda}{2\pi}} \tag{2.4}$$

$$v \approx \sqrt{gh} \tag{2.5}$$

It is important to note that the speed of a wave in deep water is only dependent on its wavelength. Sverdrup and Munk used these, along with other hydrodynamic equations, to describe and model wave generation and decay. Due to the derivation of their solutions involving integration which produced unconstrained constants, these constants were solved for empirically from data they had acquired.

In 1950, Munk [12] investigated the origin and generation of waves. Munk classified waves based on their period, defined as the time taken for successive crests of a wave to pass a fixed point. Munk classified waves into the categories shown in Table 2.1.

Classification	Period
Capillary waves	less than 0.1 sec.
Ultra-gravity waves	from 0.1 sec. to 1 sec
Ordinary gravity waves	from 1 sec. to 30 sec.
Infra-gravity waves	from 30 sec. to 5 min.
Long-period waves	from 5 min. to 12 hours
Ordinary tides	12 hours to 24 hours
Trans-tidal waves	24 hours and up

Table 2.1: Munk's classification table based on wave period

Munk showed that most of the energy contained in the wave could be attributed to both ordinary gravity waves and ordinary tides shown in Fig. 2.2. Wind was the main contributor to the generation of waves with a period of between 5 and 15 seconds. Munk stated that the use of wave energy spectra from observed sea states would have use in engineering as it would enable engineers to "evaluate its effect on engineering structures, harbours, and many other things".

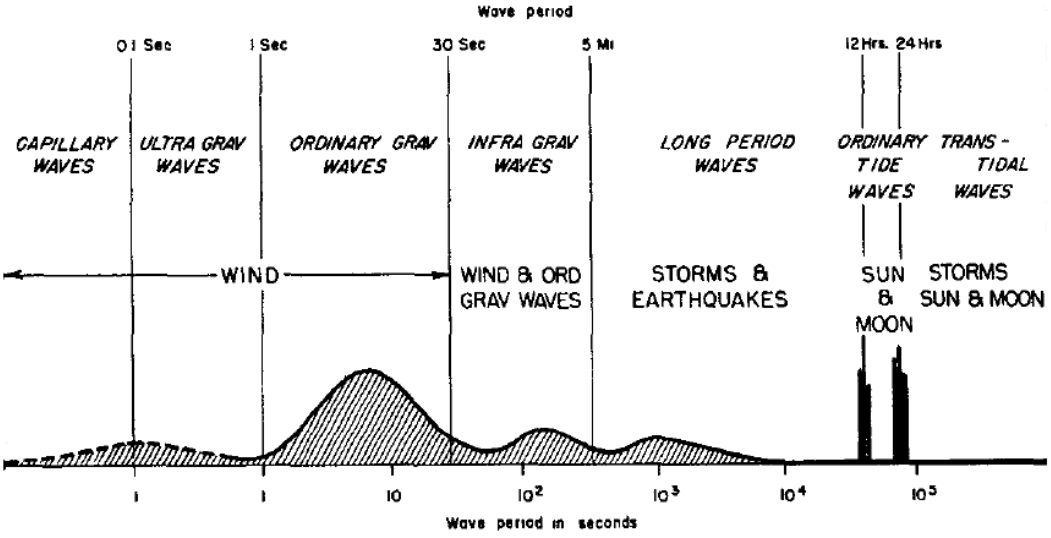

Figure 2.2: Classification of waves and their relative amplitude (shown by the curve) and the forces responsible for the generation of those waves

Bretschneider [13] talks about an acceleration in state of the art research that occurred around 1952 which focused on empirically modelling the power and variance spectra of waves. Bretschneider was able to expand on Sverdrup and Munk's progress by developing wave generation formulae which related wave height and period as a function of wind speed, water depth and fetch length, allowing for wave generation to be modelled in shallow water. This growth in wave spectrum research created a new branch of wave theory and has been an extensive field of research ever since. Cavaleri et al. [14] discuss advancements and pitfalls in the field of wave modelling in a state of the art review based on the previous 60 years of research. Cavaleri et al. discuss the more modern approaches to wave modelling which use fundamental physics to model non-linear interactions; however, the computing power available at the time (2007) prohibited its evaluation. For wave modelling not involving non-linear interactions such as those with the ocean floor in shallow waters, wave spectra provide realistic estimates, especially those modelling wind waves.

2.1.2 Advancements in Wave Spectra

Pierson-Moskowitz / Bretschneider

In 1964, Pierson and Moskowitz [15] proposed a power spectrum for fully developed seas. This spectral form was in agreement with that proposed by Bretschneider. Pierson and

Moskowitz found that given a known wind speed, a power spectrum for the waves could be approximated by Eq. 2.6. In the experiments performed by Pierson and Moskowitz, the wind speed ($U_{19.5}$) was measured by a ship's anemometer approximately 19.5 metres above the sea surface. Dimensionless constants in Eq. 2.6 are: $\alpha = 8.10 \times 10^{-3}$ and $\beta = 0.74$. Eq. 2.6 became a basis for the development of many other spectra.

$$S_{PM}(\omega) = \frac{\alpha g^2}{\omega} \exp\left[-\beta \left(\frac{g}{\omega U_{19.5}} \right)^4 \right] \tag{2.6}$$

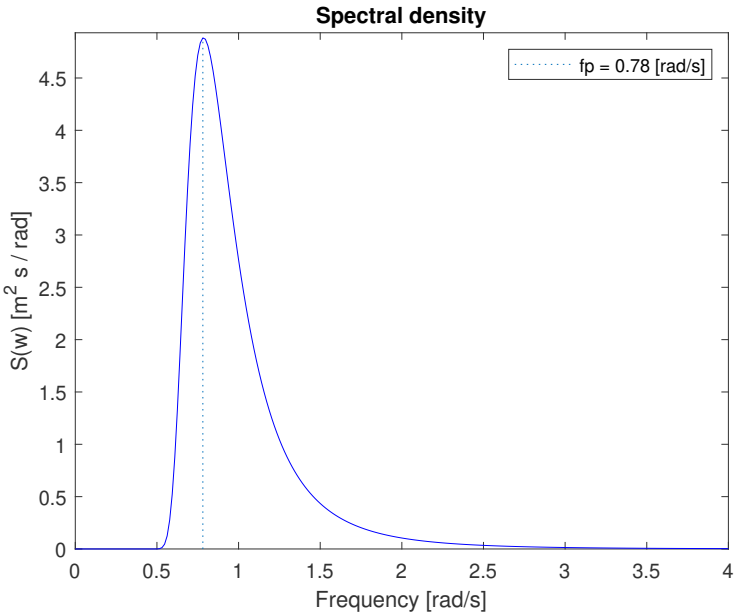

Figure 2.3: A plot from the WAFO MATLAB Toolbox of a Pierson-Moskowitz spectrum with significant wave height $H_m0 = 6m$ and peak period $T_p = 8sec$

JONSWAP

In 1973, Hasselmann et al. [16] made significant developments relating to modelling spectra of waves which had a finite fetch length and were not fully developed, unlike those which the Pierson-Moskowitz spectrum modelled. The JONSWAP spectrum, named after the Joint North Sea Wave Project, was based off the Pierson-Moskowitz spectrum; however, the formula for the spectrum was modified by multiplying the Pierson-Moskowits spectrum with a wave peak enhancing function. This function was added to increase the accuracy of the spectrum in order to better fit the North Sea readings which were obtained during the project.

Ochi-Hubble

In 1977, Ochi and Hubble [17] proposed a six parameter, double peaked spectrum which was the combination of two modified Pierson-Moskowitz spectra. These two spectra modelled both low frequency swell components and higher frequency wind wave components. This allowed nearly all deep water sea states to be modelled using this spectrum. Brodtkorb et al. [18] stated that one of the main advantages of this model was that it allowed for the sea state to be dominated by either swell, wind waves or a mix of the two.

Torsethaugen

In 1993, Torsethaugen developed a spectrum which was created by averaging two JONSWAP spectra and was used to model wave spectra on the Norwegian Continental Shelf. In 2004, Torsethaugen and Haver [19] revisited the formula proposed in order to reduce the number of parameters required for its use. The Torsethaugen spectrum has similar advantages to those of the Ochi-Hubble due to its double-peaked properties.

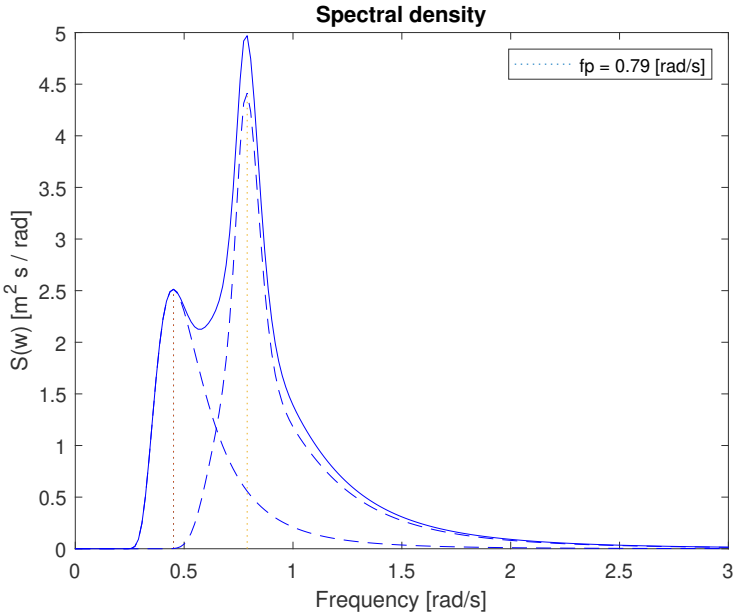

Figure 2.4: A plot from the WAFO MATLAB Toolbox of a Torsethaugen spectrum with significant wave height $H_m0 = 6m$, peak period $T_p = 11sec$

Others

The International Towering Tank Conference (ITTC) [20] lists a range of other spectra including their own ITTC spectrum which is based on the Pierson-Moskowitz spectrum. While certain spectra may realistically simulate a real ocean surface, they state that there is no single model which is suitable for describing all sea surfaces.

2.1.3 Conclusions from Ocean Modelling Review

The main points which were of relevance to the building of the Stewart platform were those concerning the motion of waves and characteristics which could be used as specifications for the platform. In the review, it was determined that the waves which the platform would be required to model were gravity waves. These waves were generated by wind-water interactions and in a developed state had a frequency between approximately $0.03Hz$ and $1Hz$. For fully developed or nearly fully developed seas, the waves' power was mostly contained in this frequency band - this was confirmed when examining some of the common wave spectral models. Ultimately, it was determined that if the Stewart platform was capable of generating movements and rotations covering this frequency band, realistic ocean wave simulations should be possible.

2.2 Stewart Platforms

2.2.1 Introduction

In 1965, D. Stewart [21] published the first paper on what was to become commonly known as the "Stewart platform". Stewart's research paper, titled "A Platform with Six Degrees of Freedom", discussed how a platform or end effector attached to a base using six extendible legs could be configured to be suitable for use as a simulator for helicopter pilots as can be seen in Fig. 2.5. The platform was capable of translation along the x, y and z-axes as well as rotation about the x, y and z-axes relative to the base of the platform and thus achieving six degrees of freedom (DOF). The rotation about the z-axis, y-axis and x-axis are commonly often referred to as the yaw (ψ), pitch (θ), and roll (φ).

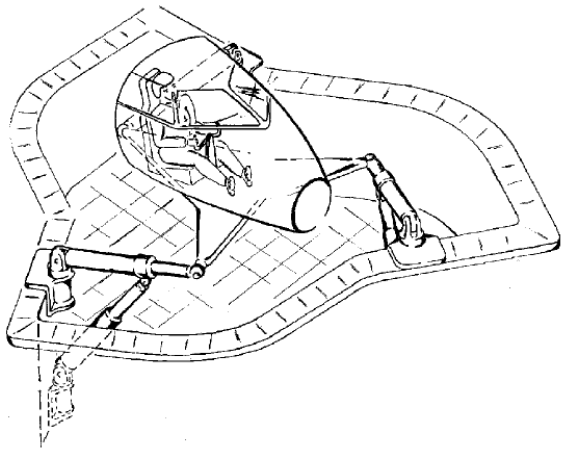

Figure 2.5: D. Stewart's pictorial description of his platform

One of the first recorded uses of this design was in 1947 by an engineer, V.E. Gough, while working on research for Dunlop. Gough [22] devised and implemented a tyre testing machine, shown in Fig. 2.6, which used 6 extendible legs attached to a platform at the one end and fixed to a base at the other. By adjusting the length of the legs, the end effector was able to achieve motion with six DOF similar to that of Stewart's platform.

Stewart includes his communications with reviewers of his paper, one being Gough. Stewart was not aware that Gough had already been using a six DOF platform functionally from 1954. While the platform was developed separately from Gough and both were designed differently, the invention of the platform is usually attributed to both Gough and Stewart.

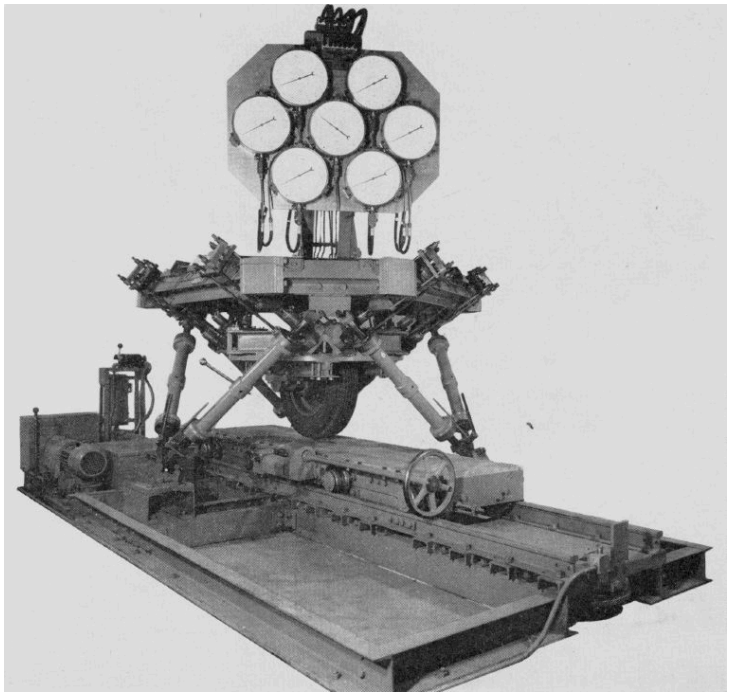

Figure 2.6: A picture of Gough's tyre testing machine

The use of parallel manipulators attracted attention as they were more stable and rigid than other platforms at the time which used sliders and gimbals to achieve the same motions.

Stewart was criticised by one reviewer of the paper regarding the realism of the rotations and accelerations exerted by the platform on the pilot, expressing how they would not be able to exert exactly the same forces as that of a helicopter. Stewart acknowledged the criticism and stated, "Unless one is simulating only the smallest of perturbations from the hover or straight-and-level flight no practical system is capable of meeting this requirement".

2.2.2 Kinematics and Dynamics

Since the 1980s, the field of research relating to parallel manipulators has been growing consistently. Much of the research done with respect to Stewart platforms has been in areas of kinematic and dynamic modelling of the platform. Kinematic and dynamic modelling can be further split into two categories: forward and inverse modelling. Forward modelling is important when one wants to simulate a system, whereas inverse modelling is important for control.

In 1983, studies from Earl and Rooney [23] and Hunt [24] made advancements into understanding the kinematics of in-parallel actuator arrangements with relation to the theory of screw systems. Hunt also provided insight into how different configurations of joints on the Stewart platform's end effector and base as well as linkages could be used to achieve desirable characteristics such as control through the use of rotary actuators shown in Fig. 2.7. Mention was made of how there was more work to be done in investigating the kinematics of parallel manipulators and how future research should explore general principles around their geometries and arrangements.

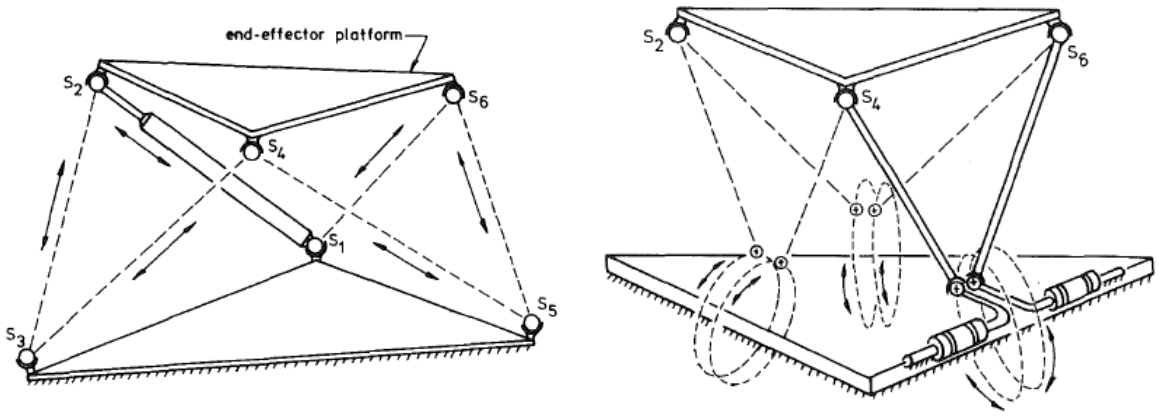

Figure 2.7: Hunt's design of a generalised Stewart platform (left) implemented using rotary actuators (right)

Much of the pioneering research regarding the kinematics and inverse kinematics of the Stewart platform was performed by Fitcher [25] and Merlet [26]. Fitcher performed work looking at the theory around designing a generalised Stewart platform and at what other considerations needed to be made in the practical construction of a platform. One of the main problems Fitcher addressed was that of the inverse kinematics of the platform - this being where one wants to determine the coordinates of the actuators given the orientation and translation of the end effector relative to the base. By determining the coordinates of the joints on the end effector and base relative to either the platform's or base's coordinate system, the leg lengths and directions can be calculated numerically. Fichter also examined the dynamics of a generalised Stewart platform in order to balance torques or forces exerted by the actuators against the inertial forces and the forces exerted on the legs by the end effector and external loads. Fichter noted that his solution to this problem was only partially complete as it assumed that the legs had no mass and that they exerted pure forces on the end effector.

Merlet investigated the problems around the direct kinematics and dynamics of the Stewart platform. He proposed developments around the Jacobian matrix (which relates the velocities of the actuators to the velocity of the platform) as well as developing a

method for determining the workspace of the generalised Stewart platform. Merlet also proposed a simple method which made it possible to determine whether a trajectory could be achieved given constraints from link lengths.

Dasgupta and Mruthyunjaya [27] have shown that using a Newton-Euler approach, the inverse dynamics of the Stewart Platform can be computed efficiently. The Newton-Euler approach requires that all constraint forces and moments between links are computed. Dasgupta and Mruthyunjaya also presented a trajectory planning approach for pre-conditioned paths (where no singularities are present) where a trajectory was planned in Cartesian space. This was then related to the actuator forces required to follow the trajectory path. Dasgupta and Mruthyunjaya also stated the need for an optimal trajectory planning scheme.

In review papers by Merlet [28] as well as Dasgupta and Mruthyunjaya [29], several methods of solving the direct kinematics problem are described. Dasgupta and Mruthyunjaya categorise solutions to these problems into groups of closed-form solutions for special platform configurations, analytic solutions and numerical solutions. The solutions in nearly all cases are both large and complex. In order to be able to compute solutions in real-time, problems were linearised to reduce computational complexity. More recently, neural networks have also been proposed as a means to solve the direct kinematics problem.

Bingul and Karahan [30] presented work on a MATLAB algorithm which was based on a Lagrangian formulation of the dynamic equations. The focus of their work related to the model's accuracy and computational efficiency and proposed two methods for the computation of the Jacobian. In a review by Bingul and Karahan, they state that in solving the inverse modelling problem, screw theory, the Newton-Euler approach and the Lagrangian approach are most commonly used.

A document by the Workingham U3A Math Group [5] describes the process of solving for the inverse kinematics of a Stewart platform. Their work provides derivations for a fixed rotary actuated platform as well as a platform with linearly actuated legs. This provides a mathematical foundation for any projects based on Stewart platforms in their various configurations.

2.2.3 Singular Position Analysis

Singular positions in parallel manipulators refer to certain positions and orientations in which the system gains one more DOF. In the case of Fig. 2.8, if the legs B_3P_{34} and B_4P_{34} are extended, the platform would be forced to rotate about $P_{12}P_{56}$. It can be seen that the direction of this rotation cannot be predicted and both Fitcher and Hunt make mention of this. In Fitcher's analysis, he points out that for the Stewart platform to be fully constrained, the directions of the forces exerted on the end effector must be linearly independent. This led to Fitcher stating: "the search for singular positions has thus been reduced to the search for positions in which the six leg vectors are not linearly independent". Merlet [31] proposed a method of finding singular positions of the platform using a new method based on Grassman line geometry. This was based on an existing method which made use of Plücker coordinates. The determination of singular positions has become an important part of the control of Stewart platforms.

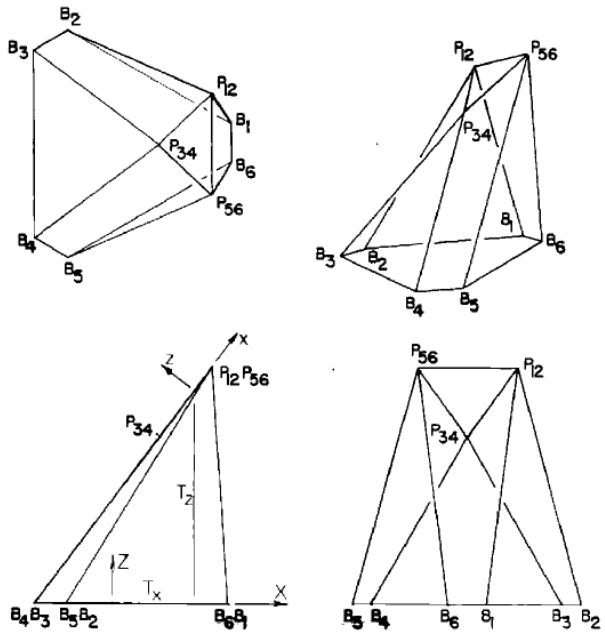

Figure 2.8: A diagram showing a singular position of the Stewart platform where the platform has gained an extra DOF around the line $P_{12}P_{56}$

2.2.4 Control

In 1993, Lebret et al. [32] performed research investigating the controllability of the Stewart platform and was successful in creating a platform which could follow trajectories

and maintain forces in certain directions. Lebret et al. made note of dualities[1] between parallel and serial manipulators and how this related to the duality between the Jacobian and inverse Jacobian. The topic of dualities in parallel and serial manipulators was covered in further depth by Waldron and Hunt [33]. Lebret et al. discussed how a simplified model for the control algorithm had to be used as many non-linearities in the kinematics made it infeasible to use the explicit dynamic model. Their model neglected the Coriolis-centrifugal effects of the model based on assumption of low velocities. Lee et al. [34] also proposed a method for approximating the inverse dynamics model of the platform and using an H_{inf} controller which treated the errors in the model approximation as disturbance errors. The H_{inf} controller was both stable and provided better responses than fully modelling the inverse dynamics and using a PID controller. Lee et al. showed that the proposed controller would be able to run on a single digital signal processor at 50Hz.

Bhattacharya et al. [35] proposed two methods for singular position avoidance in the controlling of parallel manipulator-type robots. The two methods trade-off between being able to track a desired trajectory with minimum deviation from a desired path and the computation time involved. Bhattacharya et al. showed that the first method, which used a quadratic programming model with inequality constraints on the actuator forces, yielded a minimum deviation of path when avoiding singularities; however, this approach was too computationally expensive to be used in practise. Instead, a method which involved manipulating the Jacobian in order to approximate constraints of the actuator forces was used. This method did not always maximise leg forces; however, the algorithm implemented for this method was far more efficient and so this was the method implemented in practice.

2.2.5 Workspace Analysis

While Stewart, Hunt and Fitcher all touched on the topic of the workspace of the Stewart platform, one of the first rigorous attempts at defining the workspace was made by Yang and Lee [36]. Yang and Lee found that the general 6-SPS (where S denotes the use of a spherical joints and P denotes the use of Prismatic joints) mechanism of the Stewart platform was limited in its movement due to the use of the spherical joints. This was the first time that a study of the workspace of the Stewart platform included limitations based on the constraints of the types of joints used. Yang and Lee also concluded from their study that the workspace of the Stewart platform in general was poor.

[1] an instance of opposition or contrast between two concepts or two aspects of something; a dualism.

In 1994, Masory and Wang [37] modelled the workspace of a 6-SPS Stewart platform by slicing the z-axis (which is normal to the base) at increasing heights. A boundary was then computed at each height up until a maximum height where there were no more possible configurations. Masory and Wang's method allowed for boundaries to be computed in relation to constraints on joints used in the platform, the legs interfering with each other, and the configuration of the base and end effector. In 2011, Bohigas et al. [38] presented an elegant solution to the workspace problem whereby slices could be computed for both the orientation and position of the end effector of any arbitrarily arranged Stewart platform. Their approach was also fast to compute and had the ability to include constraints based on the base and end effector configurations, joint restrictions, and leg collisions, similar to that of Masory and Wang.

In order to visualise the problem, it is common to fix three of the six DOF to allow for a 3D volume to be created representing the achievable positions and/or orientations. By fixing the position or orientation of the end effector, one can compute the constant-position or constant-orientation workspaces respectively. These are shown in Fig. 2.9.

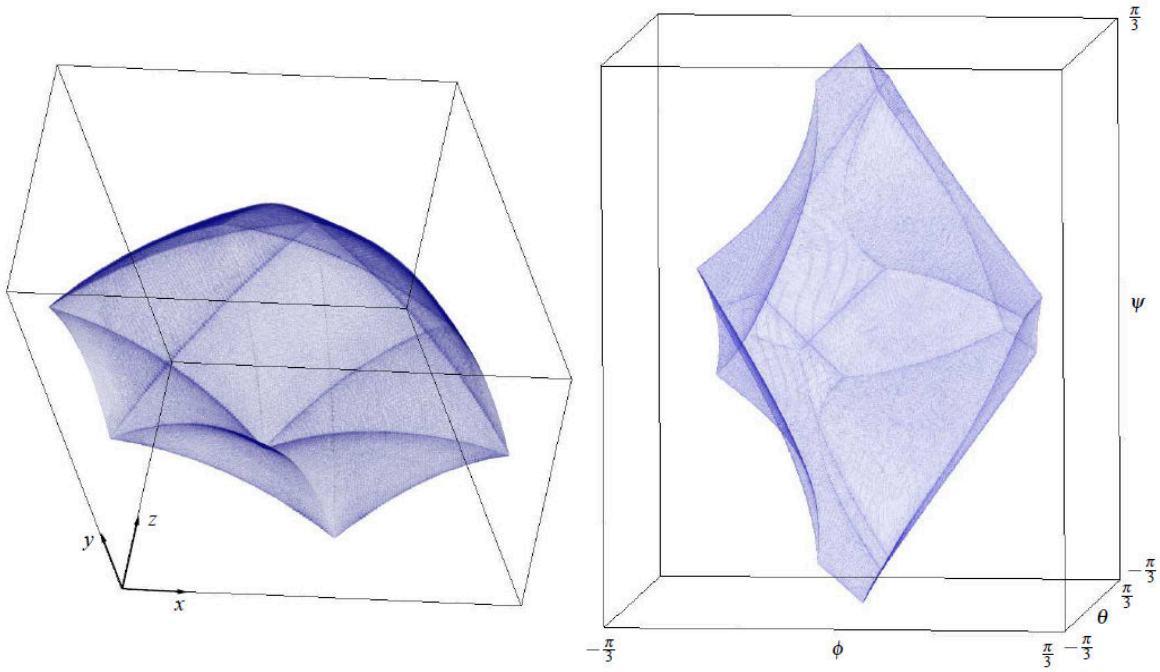

Figure 2.9: A constant-orientation workspace (left) and constant-position workspace (right) for a 6-SPS Stewart platform computed by Bohigas et. al.

2.2.6 Practical Construction

Fichter used a semi-regular hexagon structure to locate the leg joints for the base and end effector of the design, noting that if one uses regular hexagons, the platform is no longer fully constrained which is verifiable using screw theory. His design used electric motors to drive screw jacks and this allowed for the use of shaft encoders and tachometers for position and velocity control. Fichter also noted that the ends of the legs should be mounted using Hooke joints, otherwise known as gimbals, and not mounted using ball-and-socket joints as they limit the range of motion substantially. This was reciprocated by Yang and Lee [36].

Rastegarpanah et al. [39] performed a study where different configurations of platforms, shown in Fig. 2.10, were compared against one another other using a finite elements analysis in SolidWorks. The study showed that platforms of semi-regular hexagon shape were the most stable in comparison to other structures. Rastegarpanah et al. also stated that, "the stability of different structures completely depends on the position of the joints and actuators and the shape of the platforms".

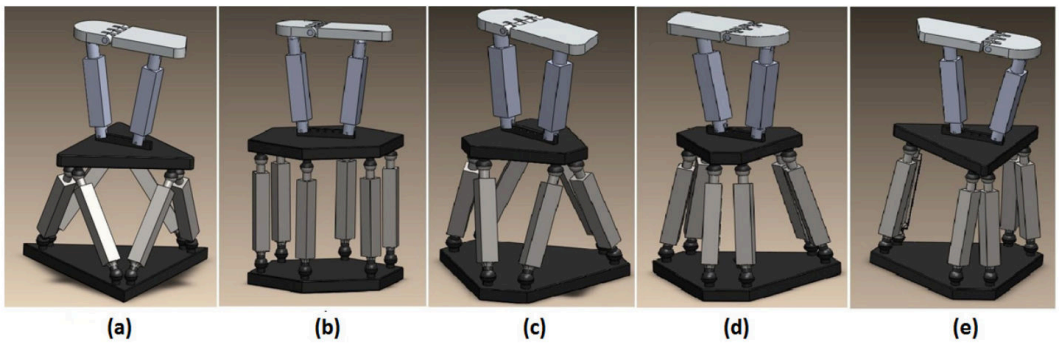

Figure 2.10: A diagram showing the different platform configurations compared by Rastegarpanah et al.

Chapter 3

Methodology

3.1 Project Plan

In building the Stewart platform, the planning, development and evolution of concepts, designs and prototypes all make up important stages of the project design life-cycle. This section aims to provide a detailed understanding of the problem which this project aims to solve and in doing so, sets out the requirements and constraints for the project.

3.1.1 Problem Definition

The aim of this project is to create and evaluate a small Stewart platform capable of mimicking the movement of an ocean wave's surface. The project brief indicates that the final design should be capable of "fooling" an IMU into reading measurements in agreement with those experienced on an ocean wave's surface. Tests should also be performed to compare the movements experienced by the IMU on the platform against input/simulated movements.

3.1.2 Project Requirements

The requirements from the project brief were reviewed along with literature on and around the topics of Stewart platforms and ocean modelling. As well as this, several requirements were created to aid in limiting the scope for the project and to make use of components

already in possession of the university. The following list of requirements was established:

1. Develop and build a controllable, small-scale Stewart platform. The Stewart platform should:
   (a) be a small model with a platform capable of 6-DOF movement,
   (b) use electrical actuators to control the platform's movement,
   (c) make use of a microcontroller to control the platform's hardware,
   (d) be controlled via input from a computer so that its movements can be controlled 'on-the-fly',
   (e) be capable of replicating accelerations/movements similar to those of scaled-down ocean wave surfaces, and
   (f) consist primarily of 3D printed and laser cut components.
2. Develop microcontroller code to control the Stewart platform. The microcontroller code should:
   (a) communicate from the PC to the microcontroller via a USB cable,
   (b) be able to control the positions of the actuators,
   (c) act as an interface for the platform to the PC software,
   (d) abstract away low-level hardware, and
   (e) define a protocol to be used to communicate with the PC software.
3. Develop software to control the Stewart platform from a PC. The software should:
   (a) communicate from the PC to the microcontroller via a USB cable,
   (b) be able to control the positions of the actuators based off a simulated model of the platform, and
   (c) be cross-platform compatible.

3.1.3 Analysis of Constraints

There were several constraints determined at the beginning of the project. These constraints shaped decisions during the conceptual design stages as well as manufacturing stages of the project. These initial project constraints were as follows:

- A budget of R1,600.00 was provided for the purchasing of components which were not already owned by the university
- 3D printing of parts made from PLA (polylactic acid) and laser cutting services were available for free from the university, resulting in a design where the majority of mechanical parts would be created using these methods
- Time constraints meant that a rapid prototyping phase was required and hence a familiar microcontroller was used
- Availability of components due to cost and shipping time

3.1.4 Design Approach

The approach which was taken in planning and carrying out the project was to divide the project into tasks. Each task had specific goals associated with it. For example, the task of developing the microcontroller code had a goal of being able to commuicate with a computer via a USB connection.

The initial phases of the project were focused on determining user specifications and technical specifications for the design. These specifications were created after a review of literature on relevant topics, the project brief, consultations with lecturers and data on ocean waves (positioned off South Africa's coastline) gathered from communications with the Council for Scientific and Industrial Research (CSIR).

Each module of work undertaken was based off the requirements. Different ideas on how to meet these requirements were conceptualised and prototyped where necessary. This was mostly performed when designing the mechanical components of the project. Testing and verification was then done on each module before being integrated with other modules in the system. A V-diagram showing the flow of the project life-cycle is shown in Fig. 3.1.

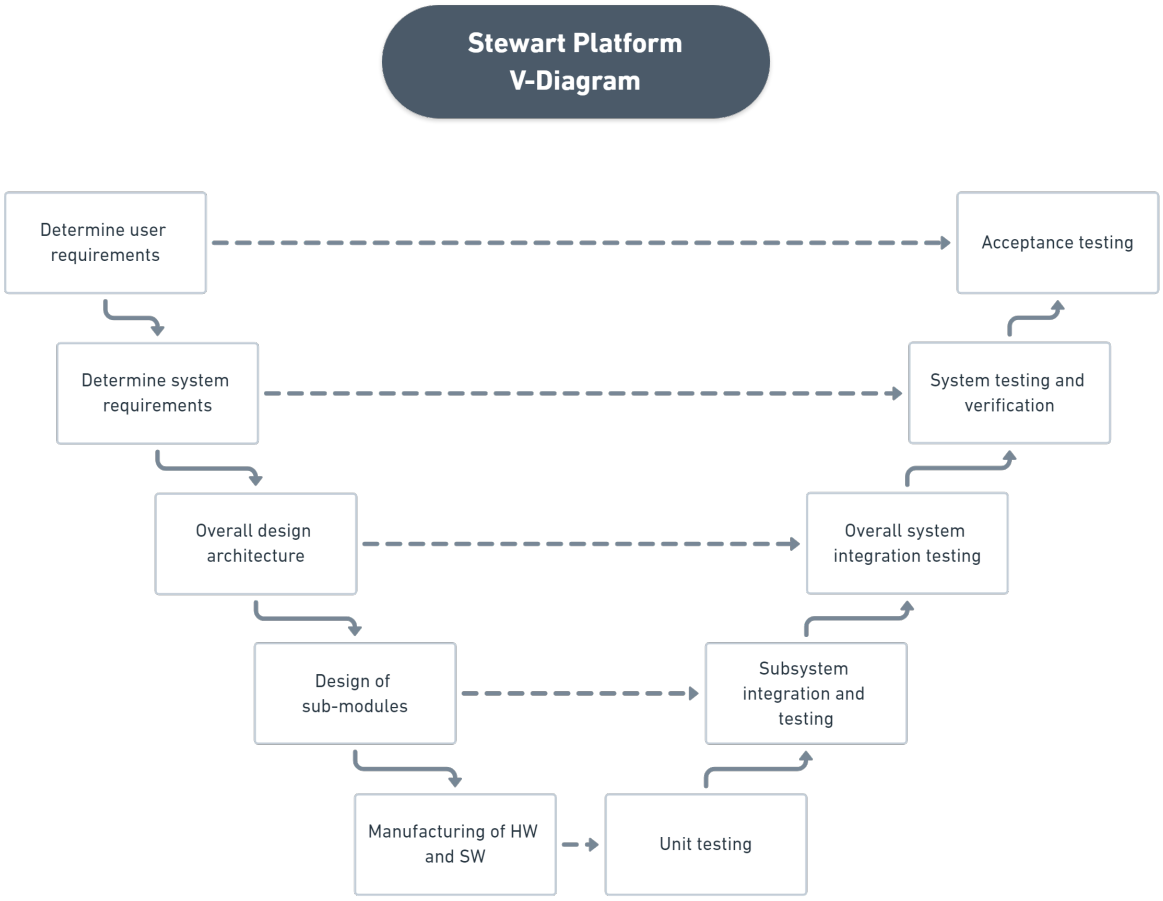

Figure 3.1: A V-diagram showing the flow of work (from left to right) throughout the project

A work breakdown structure (WBS) was used to illustrate what work was required to achieve each goal and is shown below in Fig. 3.2.

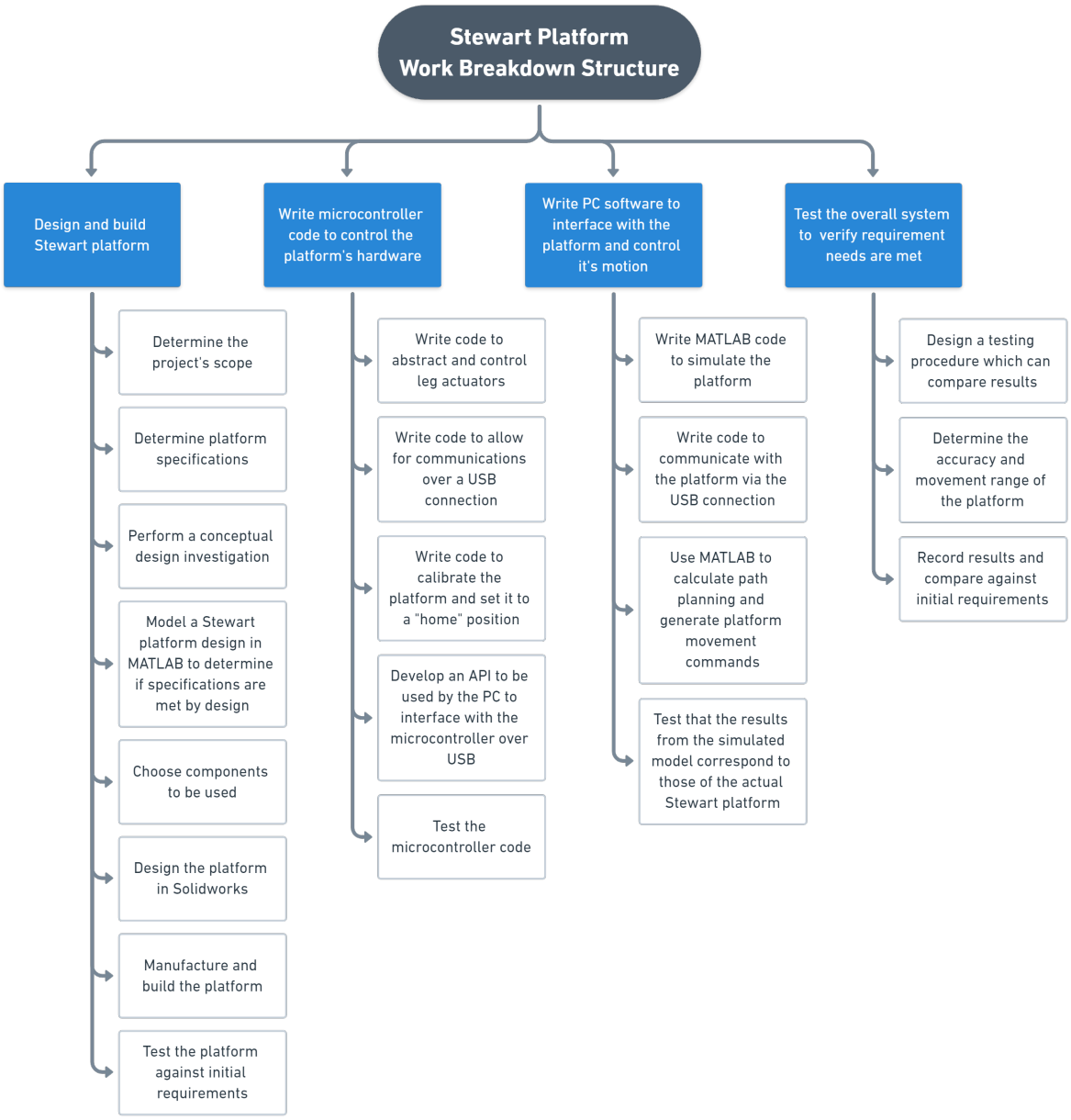

Figure 3.2: A work breakdown structure showing work required for tasks to be completed

3.1.5 Technical Specifications and Requirement Analysis

A requirement analysis was performed based on the approach mentioned above. The requirements were split into groups based on their work module and the practise of engineering under which they fell (electrical, mechanical and software). The requirements are listed below:

General Design Requirements

RG-1: Stewart Platform:

- RG-1.1: The platform should look professional and be a complete product.
- RG-1.2: The platform should be a self-contained unit, not needing to be assembled and disassembled for transportation.
- RG-1.3: The platform should not need any external electronics to function such as a power supply (with the exclusion of the computer which it would be plugged into).
- RG-1.4: The platform should look aesthetically appealing.

Mechanical Design Requirements

RM-1: Stewart Platform:

- RM-1.1: The platform should have an end effector capable of 6-DOF movement.
- RM-1.2: The platform's end effector should be capable of achieving a yaw, pitch and roll of $\pm 15^\circ$ from the horizontal in a centred position (i.e. no translation has occurred).
- RM-1.3: The platform's end effector should be capable of translation along the x, y and z-axes.
- RM-1.4: The platform's end effector should have a maximum error of 5° for given yaw, pitch and roll inputs.
- RM-1.5: The platform's end effector should allow for tests to be repeatable within the accuracy stated above.
- RM-1.6: The platform should have a built-in mechanism which can be used to calibrate the end effector.
- RM-1.7: The platform's legs should be driven by electrical actuators.

RM-1.8: The platform's legs should be designed in a way which attempts to minimise limitations on the end effector's movements.

RM-2: Manufacturing Materials:

RM-2.1: The platform should make use of 3D printed, laser cut and other commonly available parts in order to balance the design's robustness while ensuring it is easily re-creatable and affordable.

RM-2.2: Large flat components such as the frame should be made from commonly accessible materials such as hardboard, wood or perspex.

RM-2.3: Smaller components which are structurally complex should be 3D printed out of PLA plastic.

RM-2.4: The end effector and leg assemblies should be designed in a way which minimises their weight in order to allow for larger loads to be placed on the platform.

Electrical Design Requirements

RE-1: Electrical Safety:

RE-1.1: The entire system should be electrically safe.

RE-1.2: All electrical connections should be insulated to ensure no bare wires are exposed to the user.

RE-1.3: The system should have an external on/off switch in order to be able to turn the unit off at any time in case of malfunction.

RE-1.4: The system should use an IEC60320 C13/C14 coupler as the power inlet for the device (these are commonly referred to as 'kettle plugs').

RE-1.5: The system should have a cooling system implemented into the design to cool electronic components such as motor drivers, power amplifiers or linear voltage regulators.

RE-1.6: All power management and cabling should be contained within the system and be rated correctly for its intended use.

RE-2: Actuators:

RE-2.1: Limits should be integrated into the software to prevent dangerous movements.

RE-2.2: Actuators should be prevented from moving in ways which may cause harm to the user and/or platform.

Microcontroller Code Requirements

RuC-1: Communications Protocol:

- RuC-1.1: The microcontroller code should integrate a protocol which provides an interface to a PC via a USB cable.
- RuC-1.2: The protocol should allow for the platform's movements to be controlled.
- RuC-1.3: The protocol should allow the platforms movements to be halted.
- RuC-1.4: The protocol should allow the platform to be calibrated.

RuC-2: Control:

- RuC-2.1: The microcontroller should abstract all actuator control away from the user.
- RuC-2.2: A user should be able to input actuator position commands and the platform should perform all the necessary hardware actions to move the actuators to these positions.

RuC-3: Testing:

- RuC-3.1: Testing should be performed to ensure that the microcontroller code is stable and operates correctly.
- RuC-3.2: Tests should include software tests as well as hardware tests using lab equipment such as oscilloscopes.

PC Requirements

RPC-1: Computer Software:

- RPC-1.1: The computer software should show a visual representation of the platform through means of graphs and/or a 3D model of the platform.
- RPC-1.2: The computer software should be simple to use and be accompanied by documentation.
- RPC-1.3: The computer software should be cross-platform and run on at least Windows 10, MacOS and Ubuntu Linux to ensure its usability is maximised.

RPC-2: Communications:

- RPC-2.1: The PC should have at least one USB port to allow for communications with the platform.

3.2 Mechanical Design

The first step taken in designing was to gather information on current designs and determine what made certain designs function well or poorly. This research would allow for improvements to be made and hopefully mitigate some of the problems that others have experienced. Besides research papers, open source 'hacker' and 'hobbyist' hardware websites have become a great way to see and learn from what others have made.

Stewart platforms have become quite popular projects on these forums leading to many interesting designs becoming publicly available. The main reason these websites were of particular interest to this project relates to the nature of materials and components used in these projects. Most parts in the online designs were 3D printed or laser cut (due to these manufacturing methods becoming affordable to hobbyists), unlike many of the designs in research papers which were often machined out of materials such as aluminium and used expensive components which would not have been affordable for this project.

These websites along with the literature review made up the basis for the research of the conceptual design investigation. Trade-offs between designs and components are discussed in more detail below, ultimately leading to a final conceptual design.

3.2.1 Legs

One of the main components which influenced the design process of the leg was the choice of actuators. These had a large impact on the size and cost of the design. It was found that there were several popular methods of actuation which were used to extend the length of the legs. These included pneumatic and hydraulic linearly actuated legs, electrical linearly actuated legs and electrical rotary actuated legs. As this was an electrical engineering project, non-electrical actuation methods were not investigated further. Two promising methods of actuation were prismatic or linear leg actuators and fixed rotary leg actuators. These are depicted by C and E respectively in Fig. 3.3.

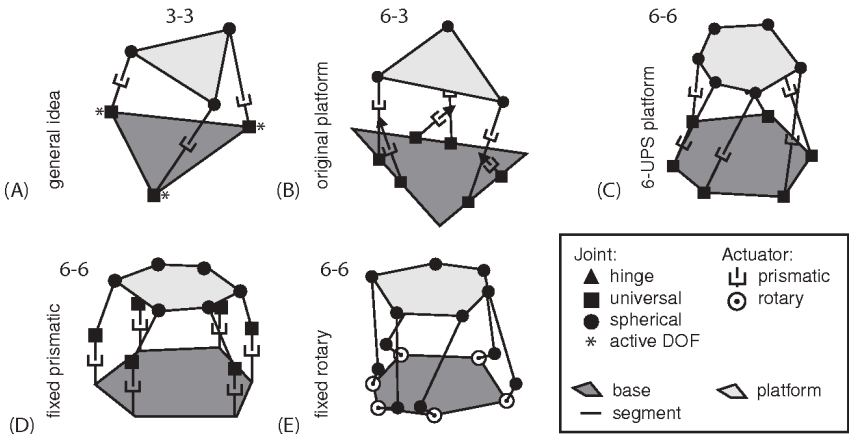

Figure 3.3: A diagram depicting several different configurations of the Stewart platform[3]

Linear Actuators

Electric linearly actuated legs, shown in Fig. 3.4, work by using ball screws to translate the rotational movement of a motor into linear movement. There are several important characteristics to note from this mechanism. Firstly, the ball screw results in a large reduction in movement meaning that for a full rotation of the motors shaft, the linear leg only moves by one tooth's width of the worm gear. This provides very precise control over the length of the leg but results in extensions and retractions being slow. A big advantage of this mechanism is that for a small input torque from the motor, a large force can be exerted by the leg.

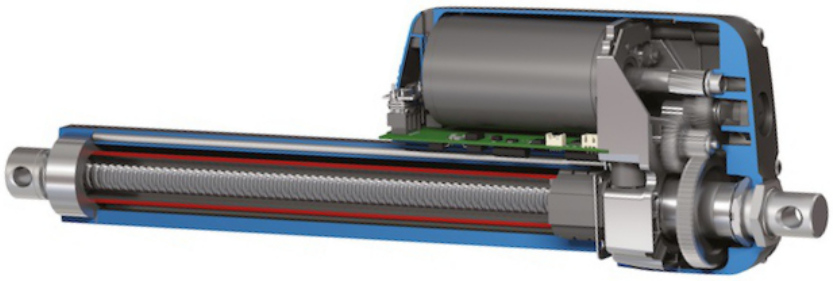

Figure 3.4: A picture showing the internals of an electric linear actuator [4]

Rotary Actuators

Rotary actuated legs are able to extend and retract by adjusting the angle between two linkages shown in Fig. 3.5. As opposed to the linear actuator, larger input torques need to be exerted by the motor to exert the same output forces on the ends of the legs. This design also has the characteristic that a small rotation of the motor results in a significant extension of the leg. This means that extensions and retractions of the leg can be much faster if the motors are able to output the required torque.

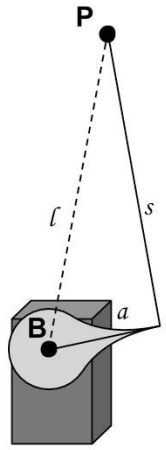

Figure 3.5: A diagram showing the geometry of a rotary leg design

Final Actuator Choice

Rotary actuated legs were chosen for the design for several reasons. Linear actuators were far more expensive than the motors and arms required for the rotary mechanism. The payload for the platform was light as it would only consist of a sensor in the form of either an IMU or a smartphone which would be used to test whether the platform responded correctly to given inputs.

A rule of thumb is that the weight of the platform is distributed approximately evenly over the six legs. This force combined with the weight of the legs contributes to the torque exerted on the motors. Dynamic inertial forces of the platform, sensor and legs contribute to the torque on the motors; however, since they move slowly and have little mass, these effects could be ignored in an approximation at this early stage of the project. The platform and its payload could be made to be light, meaning that the legs didn't need to exert large forces, hence a rotary mechanism was preferable.

Stepper motors or servo motors could be chosen as motors for the legs of the platform. Additionally, brushless DC or synchronous motors could also be used but would require optical encoders to be connected in a feedback loop to achieve angle set-point tracking.

Stepper motors were chosen as they provide more accurate open-loop control than the servo motors and would remove the need for additional closed-loop feedback mechanisms to be built or purchased. Stepper motors provided a balance between simplicity of use, accuracy and output torque. This is also one of the reasons they are commonly found in CNC machines. The stepper motors chosen, shown in Fig. 3.6, were NEMA-17 motors capable of providing a holding torque of $0.65N.m$. The size of these motors was $42 \times 42 \times 60mm$.

Figure 3.6: A picture of A NEMA-17 motor used in the platform's design

3.2.2 Joints

In using a rotary actuated design, two joints are required in each leg; one at the joint between the upper and lower leg linkages and another at the joint between the upper linkage and the platform. The affixture between the motor's shaft and the leg linkage may also be seen as a third driven joint at the base of the lower leg linkage.

Many designs, especially those online made by electronics hobbyists, use ball-and-socket joints between the leg linkages and leg-platform joint; however, the use of these joints can be severely limiting on the workspace (as discussed by Yang and Lee [36]).

A particularly interesting design which demonstrates this is one made by Dan Royer [40]. In his design, shown in Fig. 3.7, the platform's end effector is limited to 20 degrees pitch

& roll and 10 degrees yaw. As well as this, the x, y and z translation is limited to 40mm. In order to try and overcome this problem, it was decided that a new joint mechanism be devised to provide greater manoeuvrability.

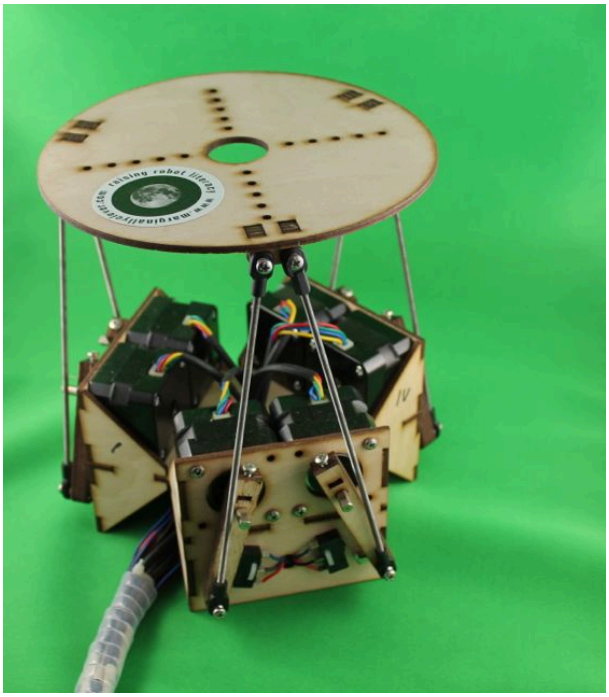

Figure 3.7: Dan Royer's rotary actuated Stewart platform design

Several joint mechanisms were explored which could be used in place of the ball-and-socket mechanism. 3D printable versions of these joints were prototyped and compared and a final design was chosen. In all of these designs, bearings were used to achieve smooth rotation. To ensure that costs were kept down, 608zz bearings were used in the designs. These bearings were cheaper than others due to them being mass produced as they are commonly used as skateboard wheel bearings.

The first design, shown in Fig. 3.8, was a universal joint which allowed for 2-DOF. An additional DOF would have to be obtained by adding bearings to one of the joint's connection points, allowing it to swivel. This design was robust but clunky and had a limited range of motion. Also, this design needed to be bolted onto a flat surfaced leg. This was due to initial concepts for the leg linkages being laser cut out of perspex or hardboard; however, this was soon decided against due to the flexibility of these materials. It was decided that the linkages should be made from aluminium rod to reduce flexibility while still being light.

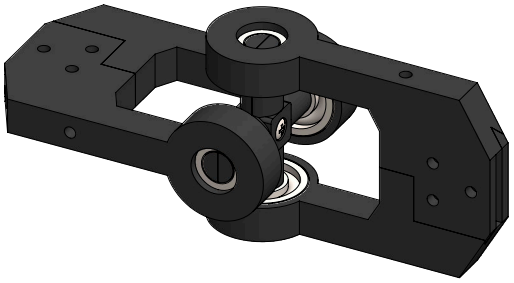

Figure 3.8: An image from SolidWorks of the universal joint design

The next design, shown in Fig. 3.9 aimed to maximise the range of motion and consisted of two rotary joints. It also included a part which allowed for a 6mm aluminium rod to be mounted through the bearings. Although a greater range of motion was achieved, this design was far less stable than the universal joint.

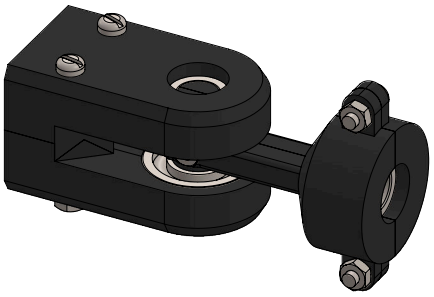

Figure 3.9: An image from SolidWorks of the 1st rotary joint design

A final design, shown in Fig. 3.10, built on the previous parts and aimed to make the second design more stable. The instability in the joint was caused by having the leg linkage mounted through a single bearing which resulted in any wobble in the bearing being exaggerated at the end of the linkage. By adding an extra bearing to the joint, the new design was stable while also providing a better range of movement.

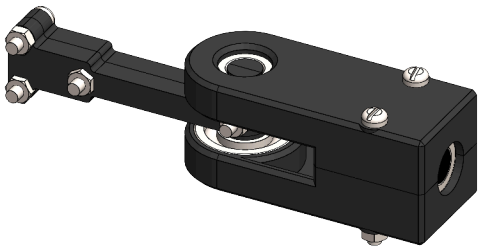

Figure 3.10: An image from SolidWorks of the 2nd rotary joint design

To compare the joints, a table describing their main features is shown below in Table 3.1.

Joint Type	Stability	Range of Motion	Bearings Required	Weight
Universal	Stable	265°	6	$97g$
Rotary v1	Unstable	290°	4	$59g$
Rotary v2	Stable	290°	4	$64g$

Table 3.1: A table showing comparisons between the joint designs

Once the leg joints were designed, connecting mounts were created. The one mount allowed the motor shaft to to be joined to an aluminium rod at 90° to the motor shaft's axis. The other mount attached the top leg linkage to the platform while allowing it to swivel about an axis normal to the platform. The top joint of the leg was allowed for it to be attached to the platform using 3 bolts. By affixing the joints to the leg linkages and mounts, the leg design shown in Fig. 3.11 was created.

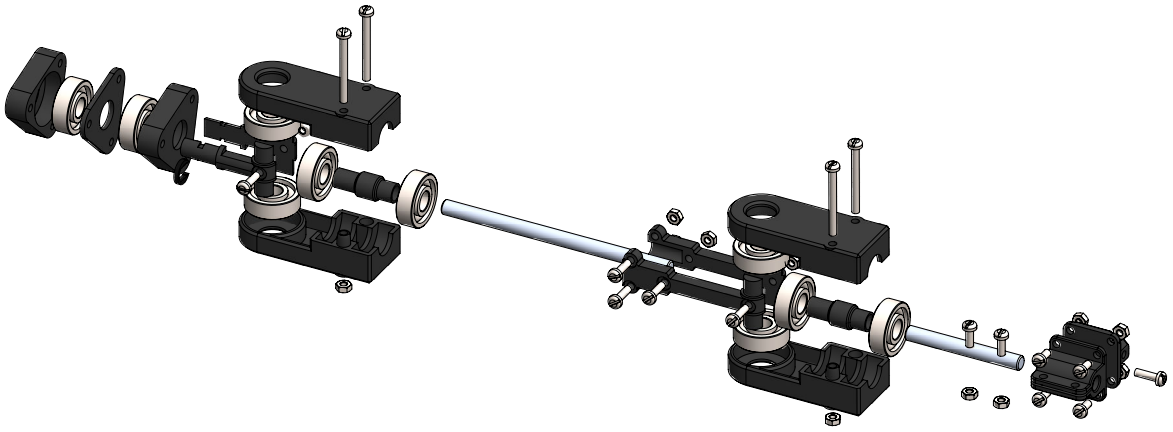

Figure 3.11: An image from SolidWorks showing an exploded view of all the components making up the leg

Figure 3.12: A rendered image from SolidWorks showing the rotary leg design used, attached to a motor pair

3.2.3 Platform and Base

In choosing the scale of the project, a small design was favoured. This was because less resources would be used in building the design and it would be faster to prototype. Parts for this project needed to be 3D printed or laser cut and additional constraints were introduced by the use of stepper motors which had a limited torque output and took up space on the base of the platform. The platform needed to be able to support the IMU sensor and so a minimum size specification for the platform could be obtained from the size of sensor.

For both the platform and the base, a semi-regular hexagon shape was used. This shape was chosen as it provides the most stability to the platform (as discussed in the literature review). The top platform, shown in Fig. 3.13, had mounting holes so that it could attach to the top of the legs and the base had mounting holes for the motor-leg assembly (shown above in Fig. 3.12).

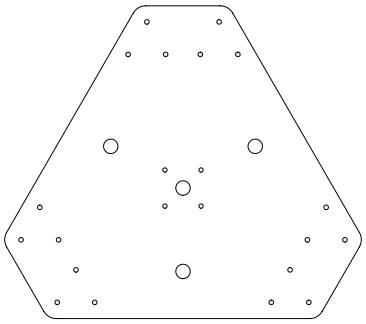

Figure 3.13: A drawing showing the semi-regular hexagon-shaped platform

The base, shown in Fig. 3.14, had cut-outs for support legs to slot into which would raise the base of the platform off the ground giving the actuators the ability to rotate below the base. The base and the platform were laser cut out of $6mm$ and $3mm$ thick hardboard respectively.

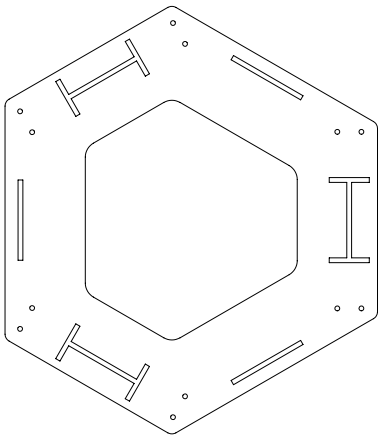

Figure 3.14: A drawing showing the semi-regular hexagon shaped base

The base was originally made out of $3mm$ hardboard but had to be made thicker to provide extra stability as the weight of the legs and platform caused the base to warp. As well as increasing the base thickness, supports were added directly underneath the stepper motors. These two modifications to the design prevented any instability under load. Support legs for the base were also designed which raised it off the ground. This was then mounted on an enclosure for the electronics.

3.2.4 Base Enclosure

An enclosure was designed to fit beneath the Stewart platform to house the electronics. Mounting holes were made so that electronic components such as the power supply and motor drivers could be bolted to the side of the enclosure. Cut-outs were also made for switches, sockets, wiring and ventilation.

Two $80mm$ fan ducts were made to allow the electronics (mainly the stepper motor drivers and power supply) to be actively cooled. Cool air was sucked in through the fan ducts at the rear of the enclosure and blown out through the ventilation holes on the sides to avoid the recirculation of hot air.

The top of the enclosure had several holes made to allow wires which connected to the actuators to be fed into the enclosure. This allowed all the wires to be hidden so that no wires would be exposed to a user. The entire enclosure was made out of $6mm$ hardboard and had internal supports to ensure it was stable for the Stewart platform. The final base and enclosure assembly is shown in Fig. 3.15.

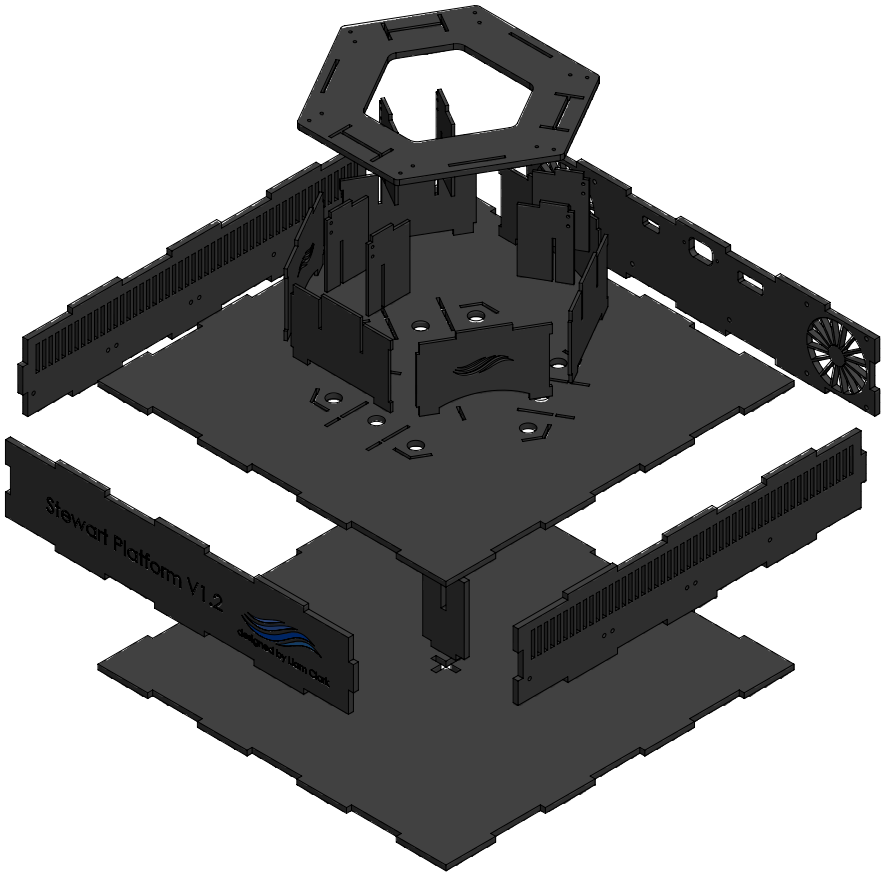

Figure 3.15: An exploded view from SolidWorks of the base enclosure assembly

3.2.5 Calibration Rig

In order to calibrate the Stewart platform, two components were created which would force the platform to be centred above the base with the platform's roll, pitch and yaw being equal to zero. To do this a mount was made for the platform, shown in Fig. 3.16 which would slide through three holes made on the top of the platform.

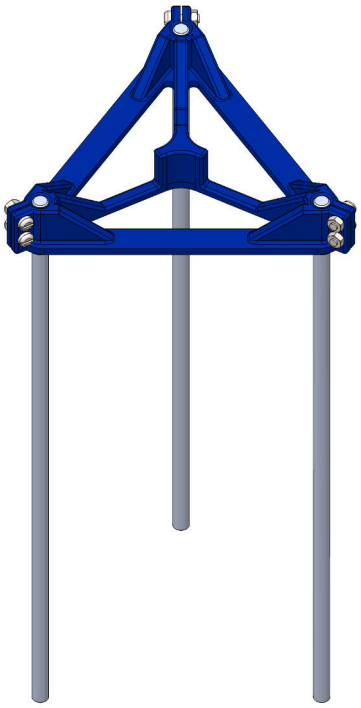

Figure 3.16: A SolidWorks model showing the platform's calibration mount

Another mount was made for the base of the platform, shown in Fig. 3.17, which the rods from the top mount would slide into. The mounts forced the aluminium rods to be normal to both the base and platform and were centred around their origins.

Figure 3.17: A SolidWorks model showing the base's calibration mount

The mounts did not account for z-axis calibration; however, the height required to raise the platform to a home position would only need to be manually determined once. A calibration routine could then be written in software to raise the platform to a home position.

3.2.6 Complete Mechanical Design

The complete mechanical design was assembled in SolidWorks and is shown below in Fig. 3.18. The assembly allowed the movement of the platform to be simulated and checks for obstructions to be performed.

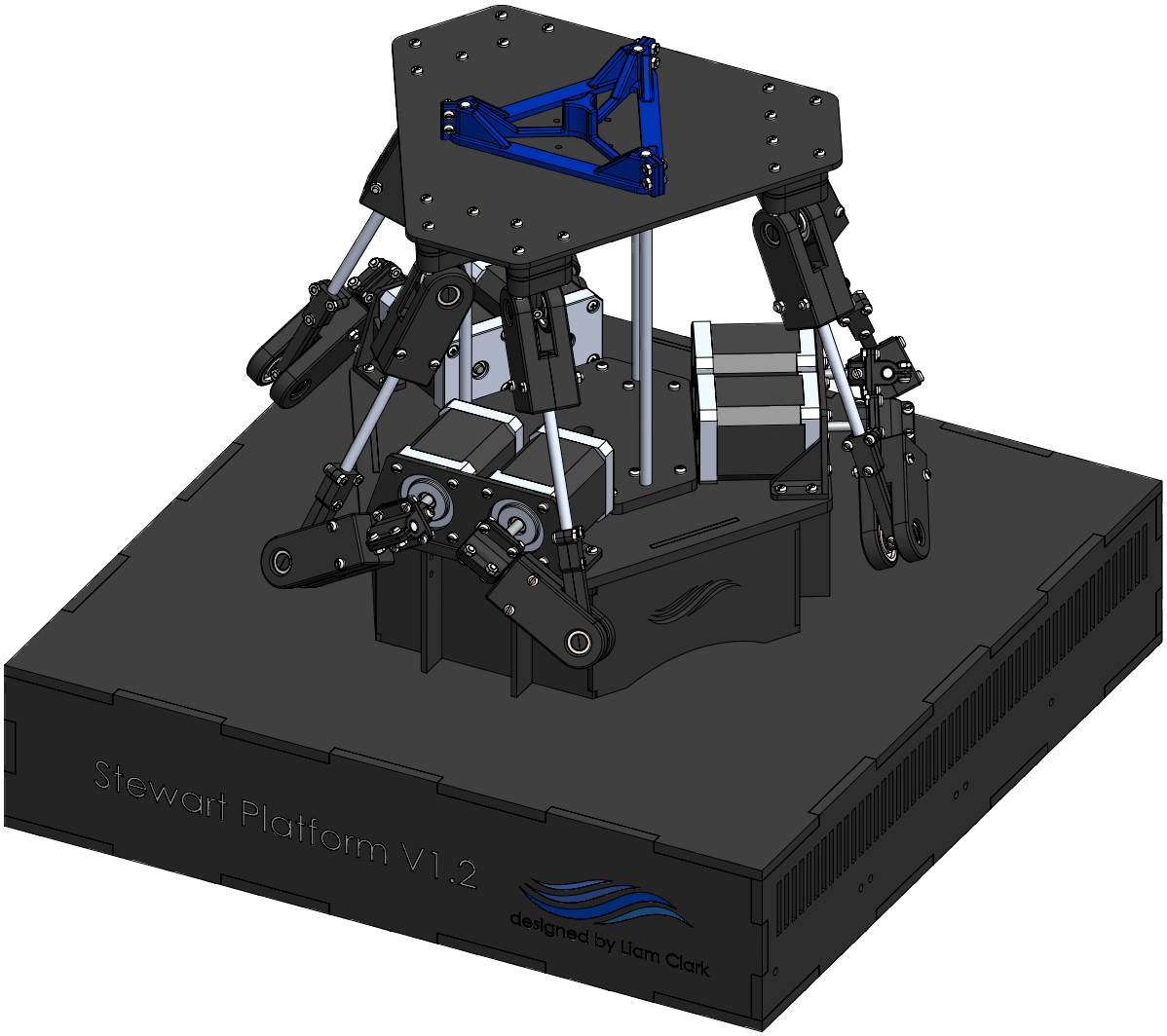

Figure 3.18: An image from SolidWorks showing the full Stewart platform assembly

3.3 Simulation Software Design

To able to control and simulate the Stewart platform, a solution to the inverse kinematics of the platform needed be found. To achieve this, one can view the process of solving the inverse kinematics problem as a function. The input to the function is the translational position and orientation of the platform along with information on the configuration of the platform (which is discussed in further detail later). The resulting output is the angular position of each actuator required to achieve the desired input position.

By using the inverse kinematics, a model of the Stewart platform could be used to determine invalid positions which the platform was incapable of achieving. As well as this, it was desirable to have a 3D visualisation of the platform based on its configuration and current position. In order to achieve this, the mathematics of the platform were first explored. Once a good understanding of the Stewart platform's mathematics was established, a MATLAB implementation of the model would be created. The MATLAB model would serve two purposes. Firstly, it would aid in simulating platforms and their configurations and secondly it would act as a base for the control aspect of the project. By reusing the code, it would mean that there would be no discrepancies between a controllers model of the platform and the simulations model. It also would reduce the time spent coding. Below in Fig. 3.24 is the resulting MATLAB simulator.

3.3.1 Mathematics of the Stewart Platform

The mathematics of the Stewart platform in this project follows the workings of the Workingham U3A Math Group [5]. This document has been used in many projects and the nomenclature used has become commonplace. For this reason, the nomenclature in this project has mostly been kept the same as in their original document to avoid confusion. The workings of several sections required for solving the inverse kinematics are not shown; however, these have all been worked through for this project and were described in MATLAB for simulation and control purposes.

Coordinate Frames

In determining the inverse kinematics of the Stewart platform, two coordinate frames are defined: one relative to the platform or end effector and the other relative to the base. The platform coordinates can be represented, relative to the base, by translational

displacements along the x, y and z-axes. Its orientation can be given by a rotation about the x, y and z-axes more commonly known as Euler angles. The Euler angles which will be used in the derivations are described below:

- Yaw (ψ) - angle of rotation about the z-axis
- Pitch (θ) - angle of rotation about the y-axis
- Roll (φ) - angle of rotation about the x-axis

These rotations can be described in terms of rotation matrices as follows:

$$R_z(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{3.2}$$

$$R_x(\varphi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\varphi) & -\sin(\varphi) \\ 0 & \sin(\varphi) & \cos(\varphi) \end{bmatrix} \tag{3.3}$$

These matrices can then be used to define orientation of the platform relative to the base by a full rotation matrix. This is shown below in Eq. 3.4.

$$^{P}R_B = R_z(\psi) \cdot R_x(\varphi) \cdot R_y(\theta) \tag{3.4}$$

Spatial Representation of the Platform and Base

To define points on the base which intersect with the leg joints, a circle with radius r is defined. Three pairs of points are then distributed along the perimeter of the circle such that each set of pairs are 120 deg apart. The base-motor assembly showing the distribution of motors is shown in Fig. 3.19.

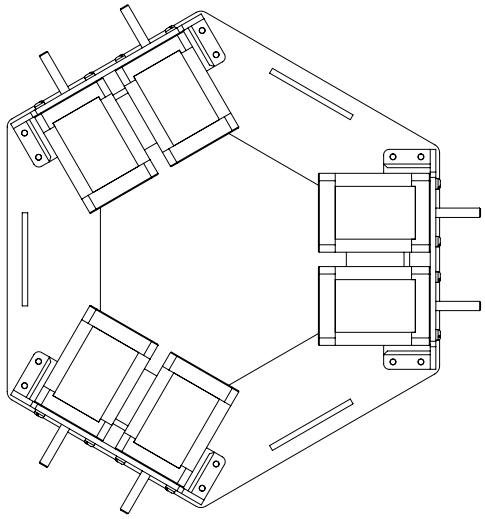

Figure 3.19: A drawing showing the distribution of stepper motors around the base of the Stewart platform

Point pairs relative to the origin of the base can then be defined by a vector b_i. These points are situated at the end of each stepper motor's shaft. The platform's points, defined by p_i when relative to the platform, are situated at the mounting points between the upper leg linkage and the platform.

In order to simplify further equations, the i^{th} point around the base and platform will be labelled as b_i and p_i respectively. Base points will be represented relative to the base coordinate frame and the platform points relative to the platform coordinate frame. Using the translational displacement T and orientation, described by ${}^{P}R_{B}$, of the platform relative to the base, the platform points can be described in the base coordinate system by Eq. 3.5.

$$q_i = T + {}^{P}R_{B} \cdot p_i \tag{3.5}$$

Leg and Actuator Inverse Kinematic Equations

Building on the work in the previous section, a vector describing the legs of the platform can be obtained. The magnitude of this vector describes the length of each leg if linear actuators were used instead of the fixed rotary actuators in this project. This is depicted in Fig. 3.20.

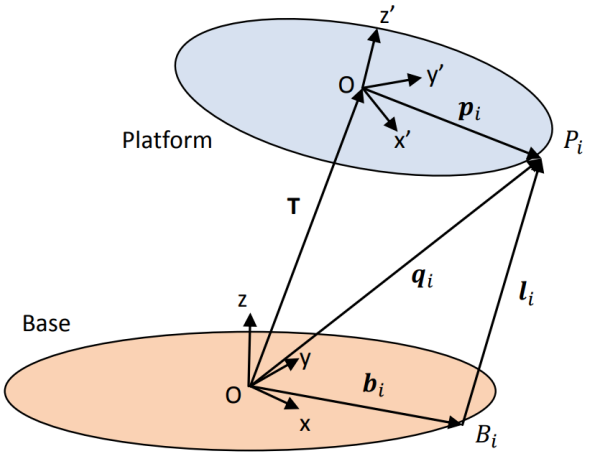

Figure 3.20: A diagram showing the geometric representation of the Stewart platform [5]

The vector describing each linear leg, l_i, is described by Eq. 3.6.

$$l_i = T + {}^{P}R_B \cdot p_i - b_i = q_i - b_i \tag{3.6}$$

To be able to determine the angular positions of the actuators in a fixed rotary design, it is required that the geometry of the actuator's planes of rotation be described. The actuator arm's movement can be described by a circle with a circumference which is the length of the actuator arm, lying in a plane. The axis of the stepper motor's shaft lies in the x-y plane, so the plane containing the actuator arm's movement is perpendicular to the x-y plane. This is shown below in Fig. 3.21. Using the knowledge of the platform's configuration, a point A_i can be defined for each leg which defines the position of the joint between the upper and lower linkage. The upper linkage length is defined as s and the lower linkage length a.

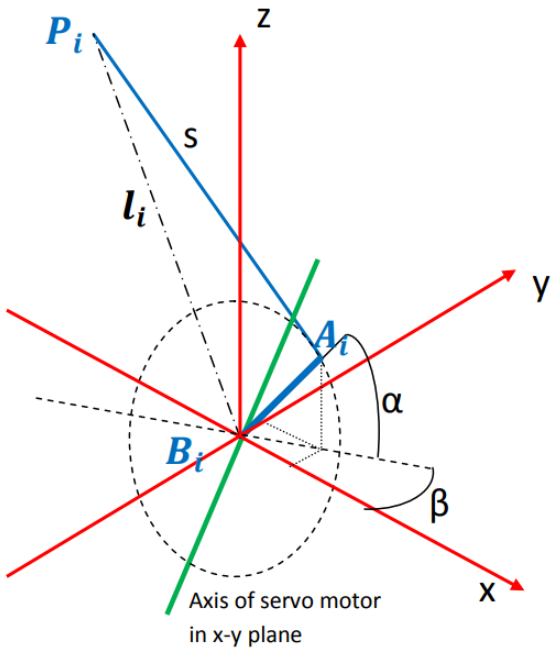

Figure 3.21: A diagram showing the geometric representation of a leg of the Stewart platform [5]

The configuration of the base-motor assembly in this project has the motors positioned as shown in Fig. 3.19. The motor pairs are distributed evenly around the base such that angle $\beta = 0°, 120°, -120°$ for each respective motor pair. This is shown in Fig. 3.21 and Fig. 3.19.

The angle α from Fig. 3.21 is the angular position of the stepper motor relative to the x-y plane. It should be noted that for most positions of the platform, there are two angular positions of the actuator which will result in point p_i being attained for the i^{th} motor. The configuration of the assembly forces the stepper motor arms to always face outwards from the center of the motor pair so that the legs do not collide. This is shown in Fig. 3.22.

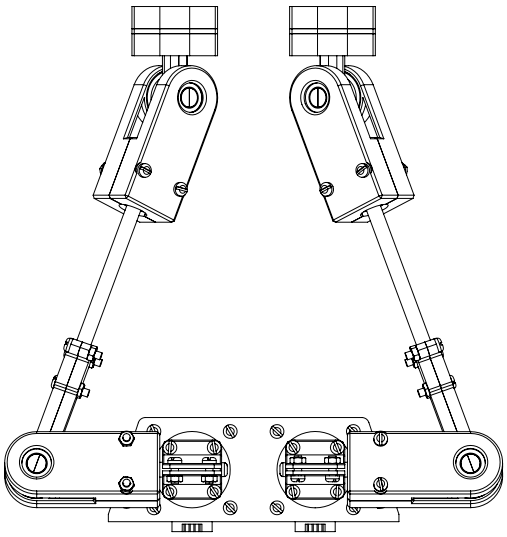

Figure 3.22: A drawing showing a stepper motor pair and leg assembly

The angular position of the i^{th} motor can be described by the Eq. 3.7, Eq. 3.8, Eq. 3.9 and Eq. 3.10. Note that in these equations, the x, y and z coordinates are referenced with respect to the corresponding i^{th} points for the base and platform vectors. The full derivation for this can be found in several papers including that of the Workingham U3A Math Group.

$$\alpha = \arcsin\left(\frac{L}{\sqrt{M^2 + N^2}}\right) - \arctan\left(\frac{N}{M}\right) \tag{3.7}$$

$$L = l^2 - (s^2 - a^2) \tag{3.8}$$

$$M = 2a(z_q - z_b) \tag{3.9}$$

$$N = 2a[\cos(\beta)(x_q - x_b) + \sin(\beta)(y_q - y_b)] \tag{3.10}$$

The next stage was to develop a model in MATLAB based off the mathematics of the inverse kinematics described in this section.

3.3.2 MATLAB Model of the Stewart Platform

Overview

Modelling of the Stewart platform in MATLAB was performed using an object orientated approach allowing for many instances of them to be instantiated at any time, each having their own unique configurations. While this may not have been needed for this project in particular, it allows the code written to scale while still being easy to interface with. The Stewart platform was defined as a class containing properties and methods. The properties included characteristics such as the lengths of the Stewart platform's linkages and motor positions, as well as variables used to store the points and vectors used in determining the inverse kinematics of the platform. The methods section pertaining to the modelling aspects of the Stewart platform included functions which calculated rotation matrices and solved the inverse kinematics problem described in the previous section.

Another special characteristic of the Stewart platform class was that it was defined as a subclass of MATLAB's abstract handle class, making it possible for an instantiated Stewart platform object to be passed between functions as a handle and not copied (i.e. passing objects by reference and not value). This has a performance impact on speed and memory if one later wants to define functions which take in Stewart platforms as arguments and operate on them. Besides this, it is also possible for one to attach event listeners to a handle, opening up many possibilities for future modifications.

The Stewart platform class can also plot a 3D graphical representation of the platform and can determine whether the platform is in an invalid position for the currently defined configuration.

Below in Fig. 3.23 is a graphic representation of the Stewart platform class showing the properties and methods associated with the class. It should be noted that this class also allows one to connect to a Stewart platform via a virtual serial connection; hence, there are methods and objects relating to communications. This will be discussed later.

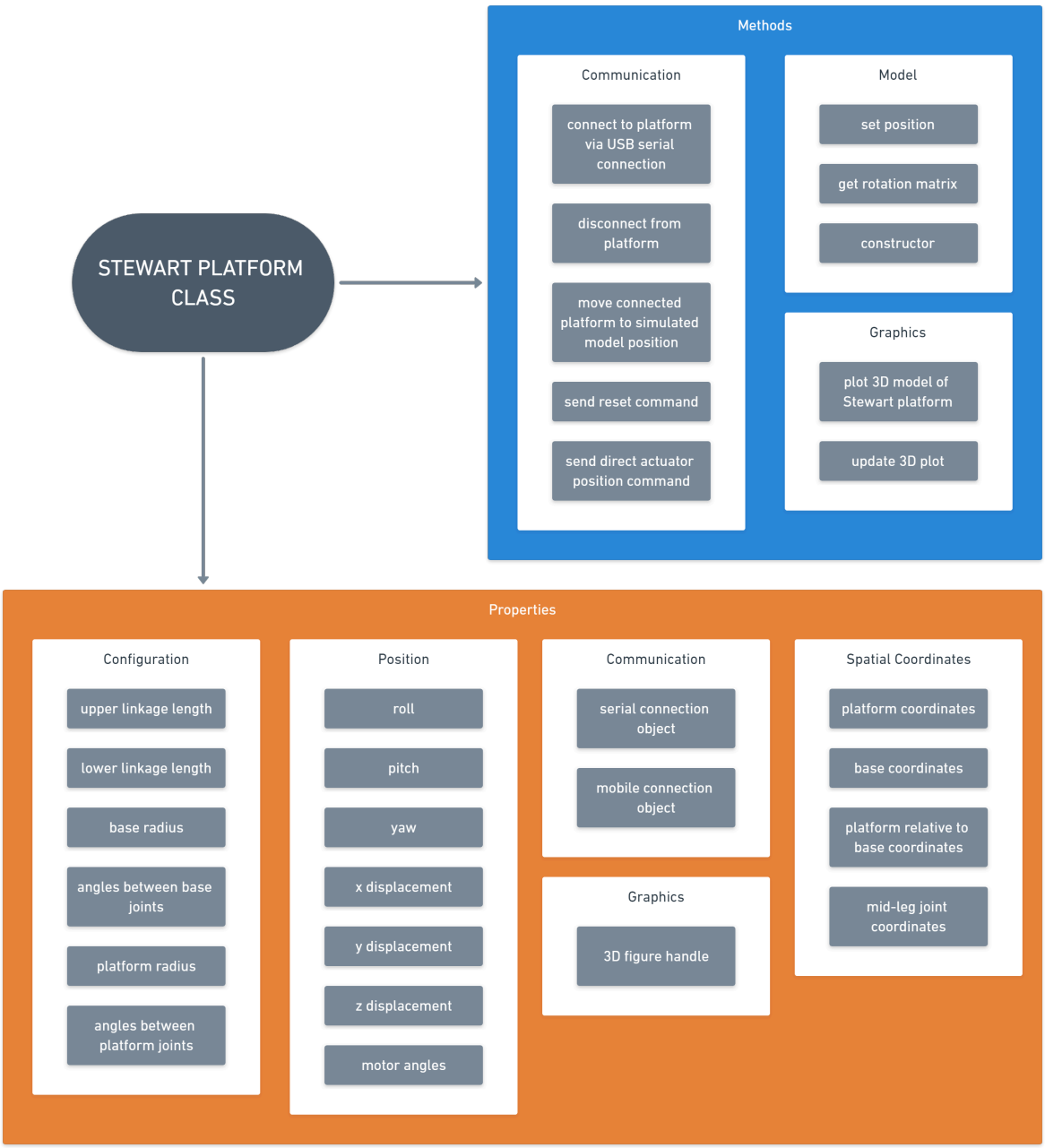

Figure 3.23: A graphic representation of the Stewart platform class

Kinematic Simulation

In order to simulate the kinematics, the inverse kinematic mathematics were implemented using MATLAB's vectorised notation where possible to allow for speed-ups in the code's execution time (such as computing the same functions once for each leg). The reason for this is that MATLAB can execute this code in parallel on multiple CPU or GPU cores if

they are available. This vastly speeds up the process of matrix multiplication as well as operations which are not dependent on one another.

A `set position` function in the class was passed orientation and translational arguments and computed the inverse kinematics of the platform. This resulted in the angular position of each motor in the class being updated. An error message was output to MATLAB's console if the desired position was not achievable. This occurred when a complex number was returned when solving Eq. 3.7, indicating that no real solution to the problem was found.

A plot of the platform could then be created using a `plot platform` method. This resulted in a 3D plot of the platform being produced as shown in Fig. 3.24.

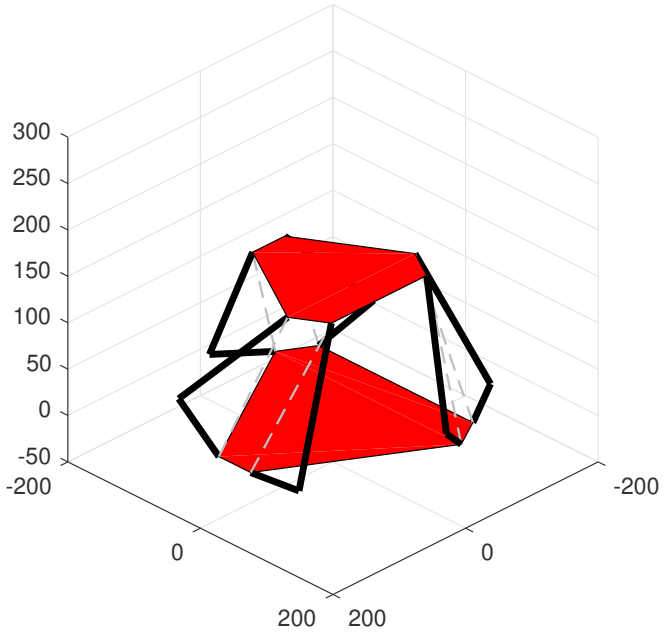

Figure 3.24: A 3D plot of the Stewart platform in MATLAB with the platform at a height of 130mm and yaw of 30°

Motion Simulation

The motion simulation served as a way to visualise the movement of the Stewart platform and to compute paths for the angular positions of the motors to follow. This was done by calculating the angular position of each motor at evenly spaced time intervals and linearly interpolating the motion between these points, similar to one of the methods

used by Bhattacharya et al. [35].

In order to achieve fluid motion simulation in MATLAB, a graphics drawing routine was devised. This consisted of a timer used to call an interrupt routine at fixed time intervals. At each interrupt, the inverse kinematics of the platform would be computed for a specific configuration and then a 3D plot of the model would be drawn, effectively rendering a frame of a video.

Initially, every interrupt routine would re-plot the entire figure and due to MATLAB's graphics drawing algorithm being notoriously slow, a new method of drawing the platform had to be devised in order for the simulation of motion to appear smooth.

To increase the speed of the plotting of the model, handles of the MATLAB figure were passed between interrupts allowing for only portions of the plot to be redrawn. This eliminated the need for MATLAB to recompute components which did not change between frames such as the axes and view perspective. Below is a comparison using MATLAB's `tic` and `toc` functions which recorded the time between interrupts before and after updating the graphics drawing algorithm. From Fig. 3.25 it can be seen that a speed-up of approximately 260% was achieved (average of 0.054 sec to 0.021 sec using a Core i7 - 6700 CPU).

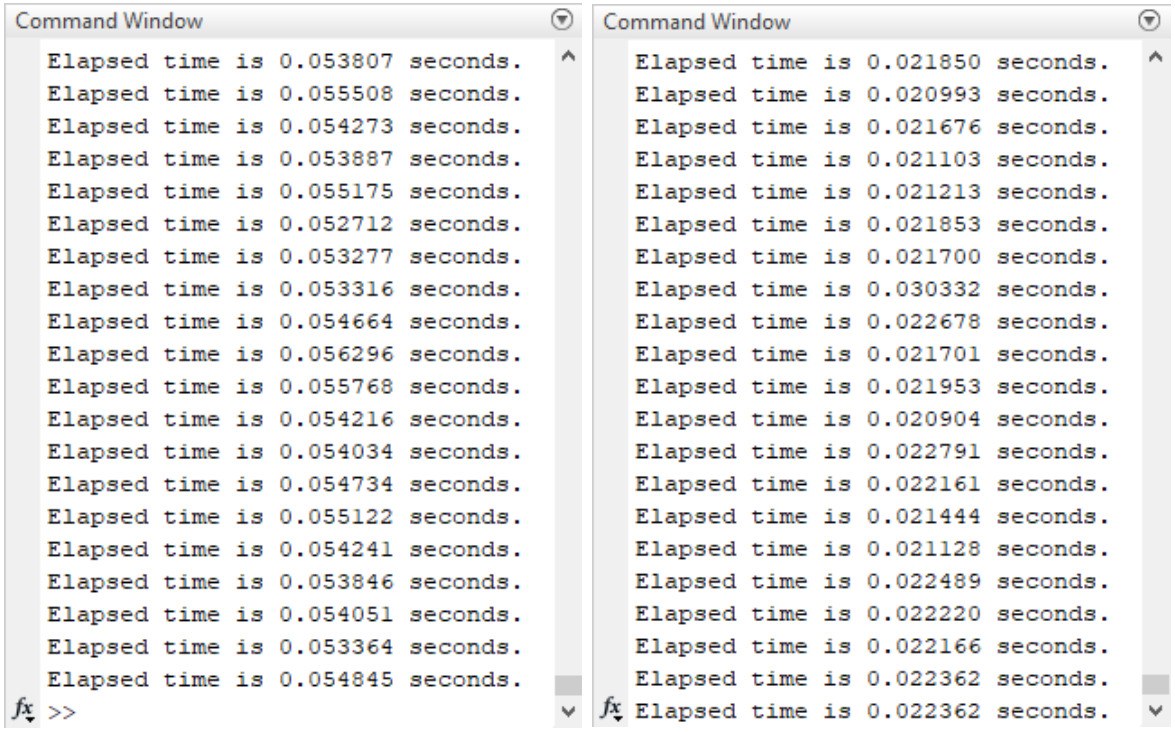

Figure 3.25: A screenshot from MATLAB showing the time taken for the graphics drawing interrupt routine to complete before (left) and after (right) being modified to run faster

In order to illustrate the motion of a video, Fig. 3.26 shows several frames overlapped with one another. The MATLAB algorithm allows for frame updates at approximately 48 fps (frames per second).

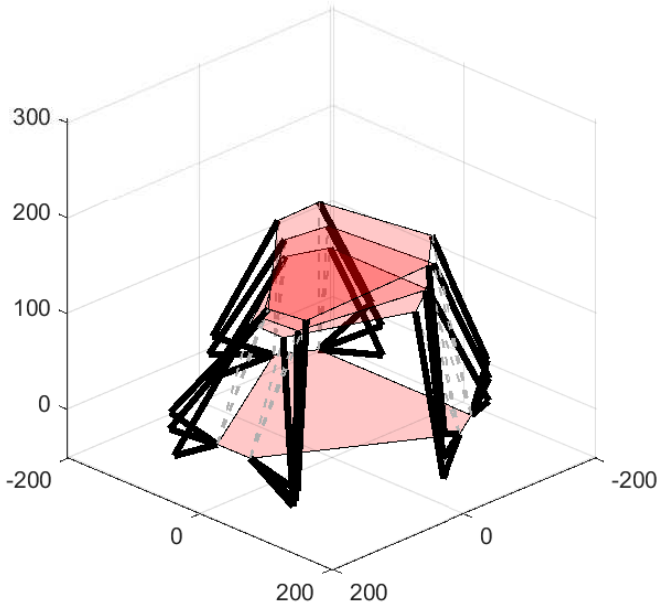

Figure 3.26: An image showing overlapped frames of a motion simulation of the Stewart platform in MATLAB

3.4 Electronic Design

The electronic design component of the project consisted of the integration of several systems shown in Fig. 3.27. Each of these modules (shown by the different colours in the diagram) will be discussed below.

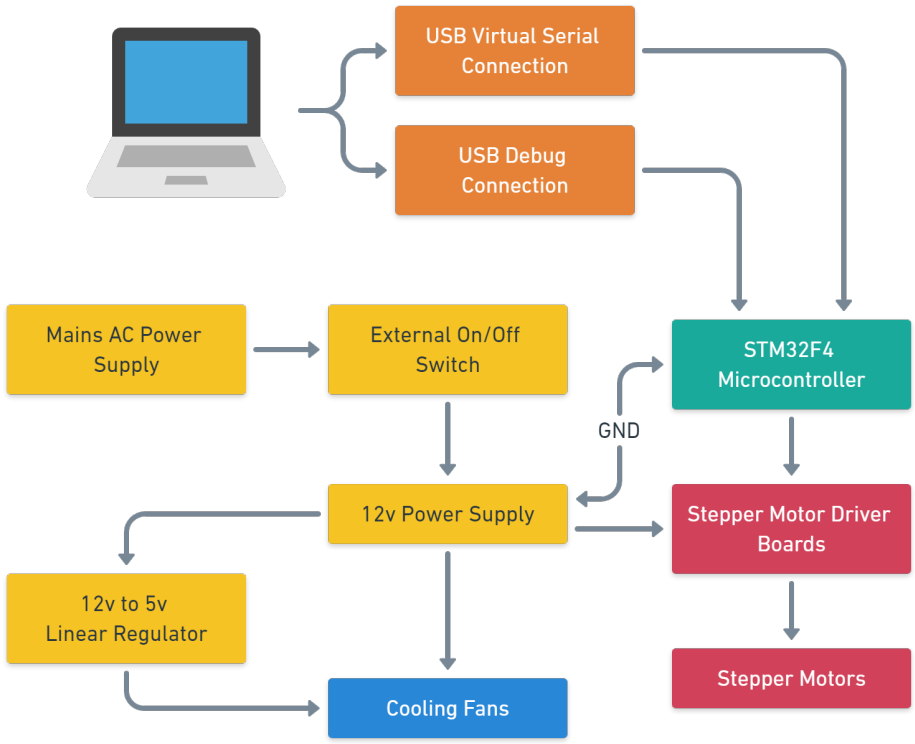

Figure 3.27: A diagram showing the various electronic components in the Stewart platform

3.4.1 Stepper Motors and Drivers

The main electronic component of the project was focused around powering and controlling the stepper motors. In order to allow the stepper motors to be controlled by a microcontroller, six DRV8825 stepper motor driver ICs (integrated circuits) were purchased, shown in Fig. 3.28. The ICs were purchased pre-solderd to breakout boards manufactured by Polulu.

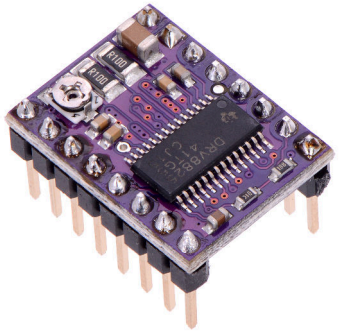

Figure 3.28: A picture of the DRV8825 board from Polulu [6]

The NEMA-17 motors which were purchased were bipolar stepper motors with a step resolution of 1.8° per step equating to 200 steps per revolution; however, the DRV8825 IC allowed for microstepping and was capable of increasing the step resolution to $\frac{1}{32}$ of the original. The DRV8825 boards were wired up in this configuration meaning that there were 6400 steps per revolution of the stepper motor. Using the DRV8825 datasheet [41] and the information on Polulu's website [6], the boards were connected as shown in Fig. 3.29. The components were all soldered onto veroboard with pin headers making them modular. LM7805 linear voltage regulators were added to each board to allow them to be used from a single 12V connection instead of both a 5V and 12V power rail needing to be supplied to the board. Two ceramic capacitors were added to the inputs and outputs of the LM7805 to filter out any noise as per the datasheet [42]. The full wiring diagram for all the stepper motor drivers can be found in Appendix B.

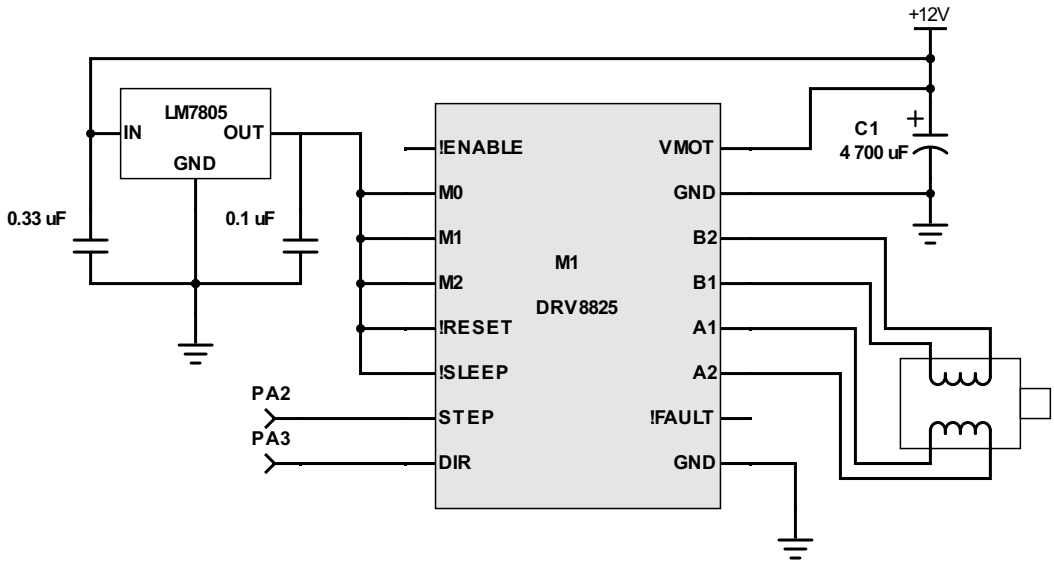

Figure 3.29: A schematic diagram of the DRV8825 wiring

The stepper motors could now be controlled by a `step` and `dir` pin on the board. When a pulse was applied to the `step` pin, the stepper motors would rotate by one step ($\frac{1}{6400}th$ of a revolution) in either a clockwise or anti-clockwise direction based on whether a logic high or low was applied to the `dir` pin. This logic was controlled by the microcontroller.

3.4.2 Microcontroller Choice and Wiring

Controlling the platform's hardware required the use of a microprocessor. The microprocessor was required to perform the task of calculating the timing of pulses to the step pin of the motor driver boards and setting the directions for each motor. The selected microcontroller for this project was the STM32F407VG Discovery development board. This board was chosen as it had a powerful 168MHz ARM Cortex M4 processor on board. This was needed as the software, which will be discussed later, needed to service several interrupt routines at a very fast rate. The development board also provided a mini-USB connector for debugging purposes and a micro-USB connection which was used as a communications port between the microcontroller and the PC. Another advantage of the microcontroller was that it provided an FPU (floating-point unit) - a dedicated, on-board hardware accelerator which sped up any floating-point calculations.

A veroboard circuit was created which allowed for the microcontroller to be mounted as a daughter-board. The veroboard was used to allow for pin headers to be created which would be connected to each of the motor driver boards. It was not possible to directly connect the motor driver boards to the microcontroller development board as the configuration of pins used was spread out over the development board itself. This made the design much neater and removed the need for breadboard jumper cables to be used. Below in Table 3.2 is a list of the pin connections which were used to connect the microcontroller to the stepper motor boards.

Microcontroller Pin	Description
PA3	Motor 1 Direction Pin
PA2	Motor 1 Step Pin
PB1	Motor 2 Direction Pin
PB0	Motor 2 Step Pin
PE8	Motor 3 Direction Pin
PE7	Motor 3 Step Pin
PE10	Motor 4 Direction Pin
PE9	Motor 4 Step Pin
PE12	Motor 5 Direction Pin
PE11	Motor 5 Step Pin
PE14	Motor 6 Direction Pin
PE13	Motor 6 Step Pin
GND	Ground to Power Supply

Table 3.2: A table showing descriptions of the microcontroller pin connections on the veroboard breakout board

3.4.3 Cooling

Components such as the DRV8825 ICs and the power supply needed to be cooled. The power supply had a built-in fan and the DRV8825 boards had heat sinks mounted to them; however, as they were both mounted inside the base enclosure, the air inside the enclosure needed to be circulated. To do this, two $80mm$ fans were used as discussed earlier. The fans required both 12V and 5V to be supplied to them. In each fan, the 12V rail was used for the fan motor and 5V rail was used for the fan's hall effect sensing mechanism which ensured that the fans operated at their rated speed of $2500RPM$.

The fans were both connected directly to a 12V power supply with an LM7805 being used to supply the 5V needed for the hall effect sensor. Again, capacitors were added to filter out noise at the input and output of the voltage regulator.

3.4.4 Power Management

In order to power the Stewart platform, a 12V, 350W power supply was purchased. In order to determine the specifications for the power supply, a table was compiled of all

the components and their respective power usages. In the calculation for power needed by the stepper motors, capabilities for the motor drivers and motors to be upgraded were taken into consideration allowing for each motor to draw currents of up to 4A at 12V. This meant that a total of $288W$ was needed for the motors and drivers. An additional 4W of power was needed for the cooling fans. This resulted in a total power requirement of 292W.

One should also note that the LM7805 linear voltage regulators were only used in circumstances where little power was needed. In circumstances where more power is needed, a different form of voltage regulation should be used such as a buck converter.

3.5 Controller Software Design

To manoeuvre the platform, the stepper motors needed to be moved in sync with each other such that at any time, the angular positions of the motors force the platform into the desired position. There are also sets of motor positions for which no platform position and orientation exists. These need to be avoided to reduce strain on the platform, arms and motors.

In designing the system, the platform's controller was split into two sub-systems. Starting at the highest level and moving down, a computer running MATLAB was used to plot paths or trajectories for the Stewart platform based on a mathematical model of the platform. This could then be used to generate commands which could be issued over a serial communication port to a microcontroller in the base of the platform. The microcontroller decoded these commands and managed the platform's actuators. This involved controlling stepper motor drivers by calculating the timing of pulses sent to the stepper motor drivers. The stepper motor drivers were responsible for providing the necessary voltages to different phases of the stepper motor.

To provide an illustration of the problem, Fig. 3.30 shows an example of how a stepper motor would track a sine wave. It should be noted that in this figure, the effects due to interpolation and step size are exaggerated. In the MATLAB code, new positions for the stepper were sent to the microcontroller every $30ms$ and there were 6400 steps required for a full 360° rotation of the motor, making it much smoother than is apparent in Fig. 3.30.

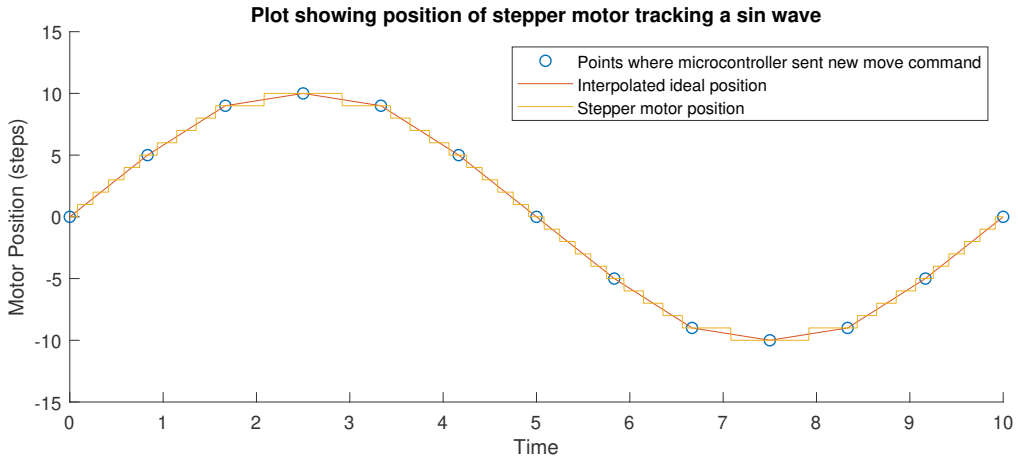

Figure 3.30: A plot from MATLAB demonstrating the interpolation of points and discrete nature of the stepper motor positioning

3.5.1 MATLAB Controller Software

In addition to the simulation code written for the Stewart platform class, several methods were developed allowing MATLAB to connect to the Stewart platform via a serial connection. Similarly to the way the motion simulation worked, MATLAB would issue commands to the microcontroller at short fixed time intervals specifying a position for each actuator as well as the time the microcontroller should take to reach the specified motor positions. The microcontroller would then be expected to move the motors from their current position to the specified position in this time.

As with the simulations, the inverse kinematics would be solved in an interrupt routine. The output of this would provide the angular position of each of the motors required to achieve the input position and orientation. An additional method was then added to the Stewart platform class which used the angular positions of the motors to calculate the required position in steps. This was given by Eq. 3.11.

$$\Delta Pos_{(steps)} = \Delta Pos_{(\theta)} \times 6400/360 \tag{3.11}$$

3.5.2 Microcontroller Software

The microcontroller software was written in C and made use of STMicroelectonics' HAL (Hardware Abstraction Layer) libraries. The main task to be accomplished by the microcontroller was to send pulses to the stepper motor drivers in such a way that the motion was as smooth as possible. The microcontroller stored a `motor` struct and a `controller` struct for each motor in memory. The `motor` struct stored properties associated with the stepper motor while the `controller` struct stored information used in moving the motor between positions, this is shown in Fig. 3.31.

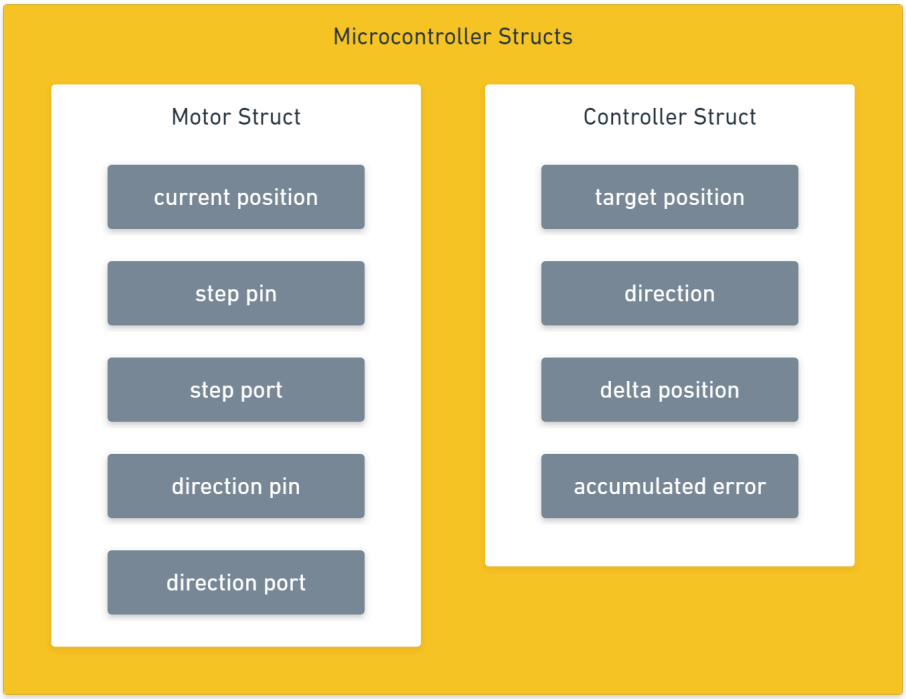

Figure 3.31: A diagram showing the properties of the `motor` and `controller` structs

The underlying problem that needed to be solved was to minimise the error between an ideal position of the motor (a continuous smooth path) and the discrete positions of the stepper motor. This is illustrated in Fig. 3.32. Luckily there was an elegant solution to this problem. Bresenham's line drawing algorithm [43] was originally written to plot graphics on IBM computers in the 1960s. Fundamentally, his algorithm provides a computationally efficient method of plotting a straight line between two points on a screen.

The reason this is important is that much like the discrete nature of a stepper motor's movement, lines drawn on a computer screen are made up of discrete pixels. Bresenham's algorithm effectively chooses which pixels along x and y axes should be used in representing a line to achieve the smoothest look; in other words, it minimises the error between an ideal line and what the computer's screen is capable of displaying. This algorithm was applied in the context of the stepper motor, where instead of discretisation in the x and y-axes, there is discretisation in the position and time-axes.

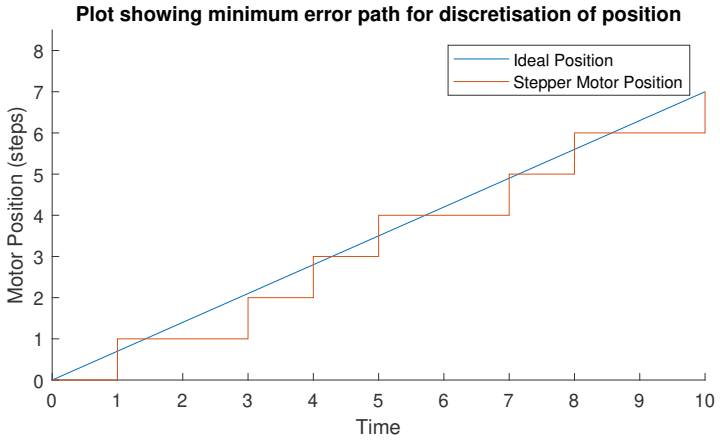

Figure 3.32: A plot from MATLAB demonstrating the minimisation problem of tracking a continuous trajectory with discrete steps

Firstly, a message protocol was developed whereby commands could be sent to the microcontroller over a USB cable. This worked by establishing a virtual serial port connection on top of the USB protocol. From the computer's perspective, the Stewart platform is seen as a serial communication device and is allocated to either a `COM` port for a Windows OS or `/dev/ttyS...` teletype console for Unix variant OSs.

When a command was received by the microcontroller, a callback function was called providing a pointer to the message buffer and the length of the received message. This message was then decoded and the motor and controller structs were modified accordingly. In the case that the command was a `status` command, the current positions of each motor would be returned over the serial connection. This process is shown in Fig. 3.33.

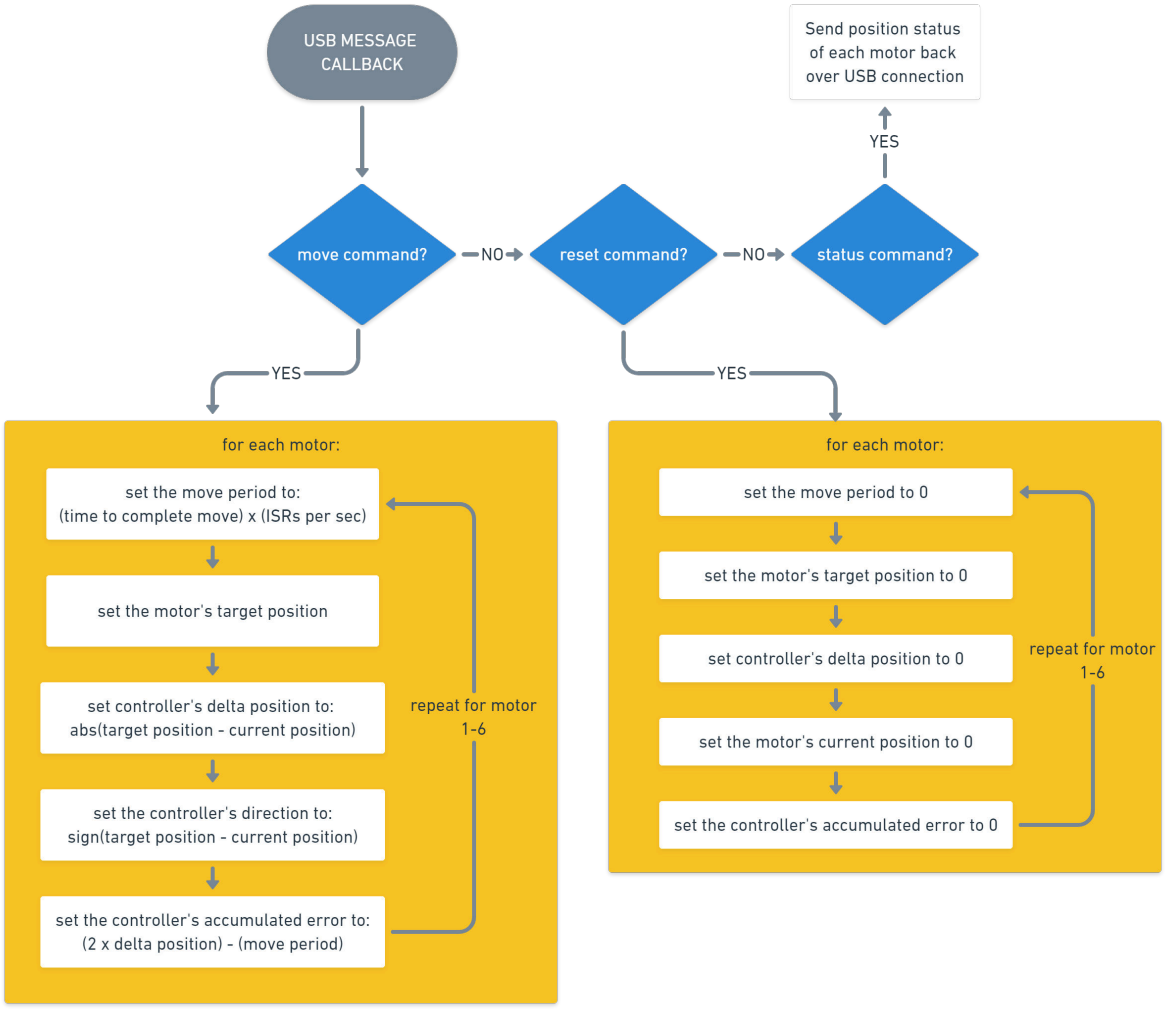

Figure 3.33: A flow diagram showing the USB callback routine for different messages

The next important section of the software was the line algorithm. Bresenham's line algorithm was implemented on the microcontroller as follows. Firstly, a timer interrupt was created which triggered an ISR (interrupt service routine) every $10\mu s$. Within the service routine it would be determined whether or not each motor needed to be stepped and in which direction. If necessary a `step` function was called which toggled the logic level on the step pin of the respective motor ensuring the `dir` pin was set correctly. The following algorithm, shown in Fig. 3.34, was implemented in the microcontroller's ISR.

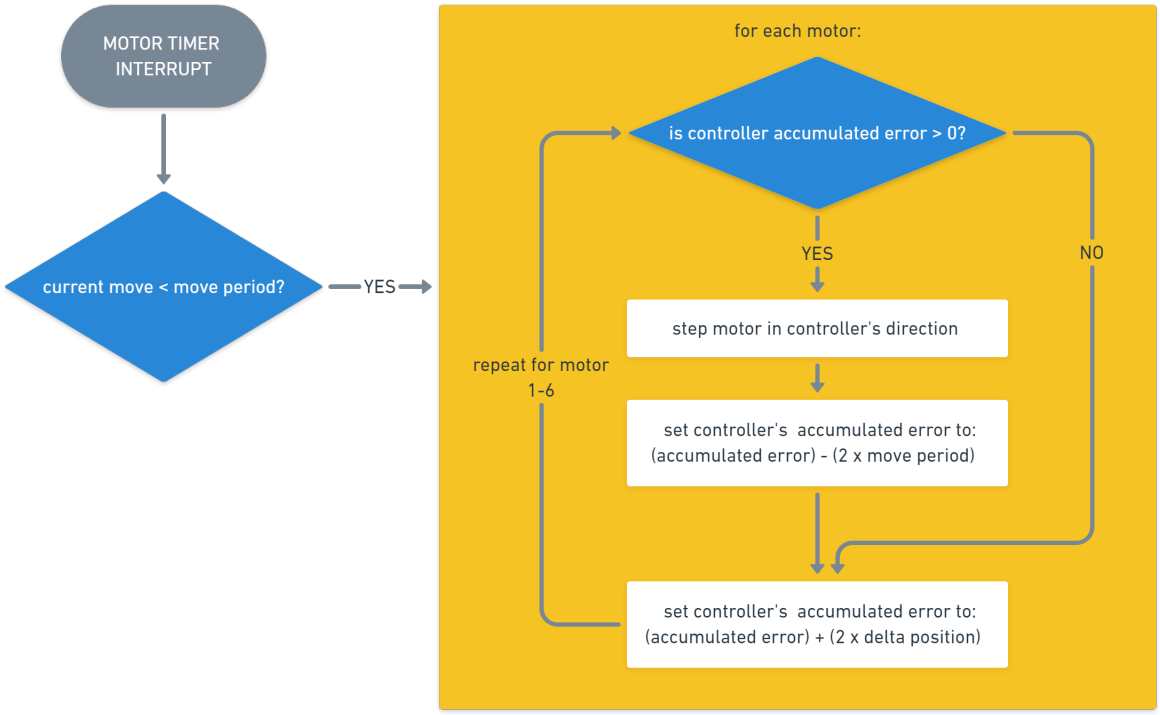

Figure 3.34: A flow diagram showing the ISR used to update the motor's positions according to a modified version of Bresenham's line drawing algorithm

For a mathematical explanation behind Bresenham's algorithm, please see Appendix A.1.

3.5.3 Communication Protocol

The communication protocol only consisted of three commands which are shown in Table 3.3. These were a reset command, a move command and a status command. The reset command set whatever position the Stewart platform was currently in to the home position, this being where each motor's angular position was equal to zero. The move command was used to update the the angular position of each motor and specified a period in which to complete a move to these positions. The status command resulted in the microcontroller sending back a message containing the current angular position of each motor.

Initially, it was expected that more commands would be needed; however, all the necessary control and calibration could take place with just these 3.

Definition	Commands
Move - set motor positions to move to in a specific time period	m <move period><m1 pos>... <m6 pos>;
Reset - set all current motor positions to zero	r;
Status - returns a string of current motor positions	s;

Table 3.3: A table of the commands used to control the Stewart platform

3.5.4 Smartphone as a Controller

An extra feature in the MATLAB code was developed which enabled the platform to be controlled using a smartphone. By adjusting the orientation of the smartphone, the roll, pitch and yaw of the platform could be changed in real-time to align itself to the orientation of the phone. The smartphone used MATLAB's mobile application to stream orientation data to the PC over a WiFi connection, provided that both were on the same local area network.

The smartphone app also had the ability to log orientation data to a file. This presents a useful way to record simulation data. This data can then be played back as a simulation at a later stage.

3.5.5 MATLAB Debugging and Calibration Application

An interface for sending commands to the Stewart platform was made to allow a user to interface with the platform using a GUI (graphical user interface) instead of having to manually entering commands over a serial terminal. This was not intended to be used for motion simulation purposes but as a helpful tool for debugging or calibrating the platform. The GUI is shown in Fig. 3.35.

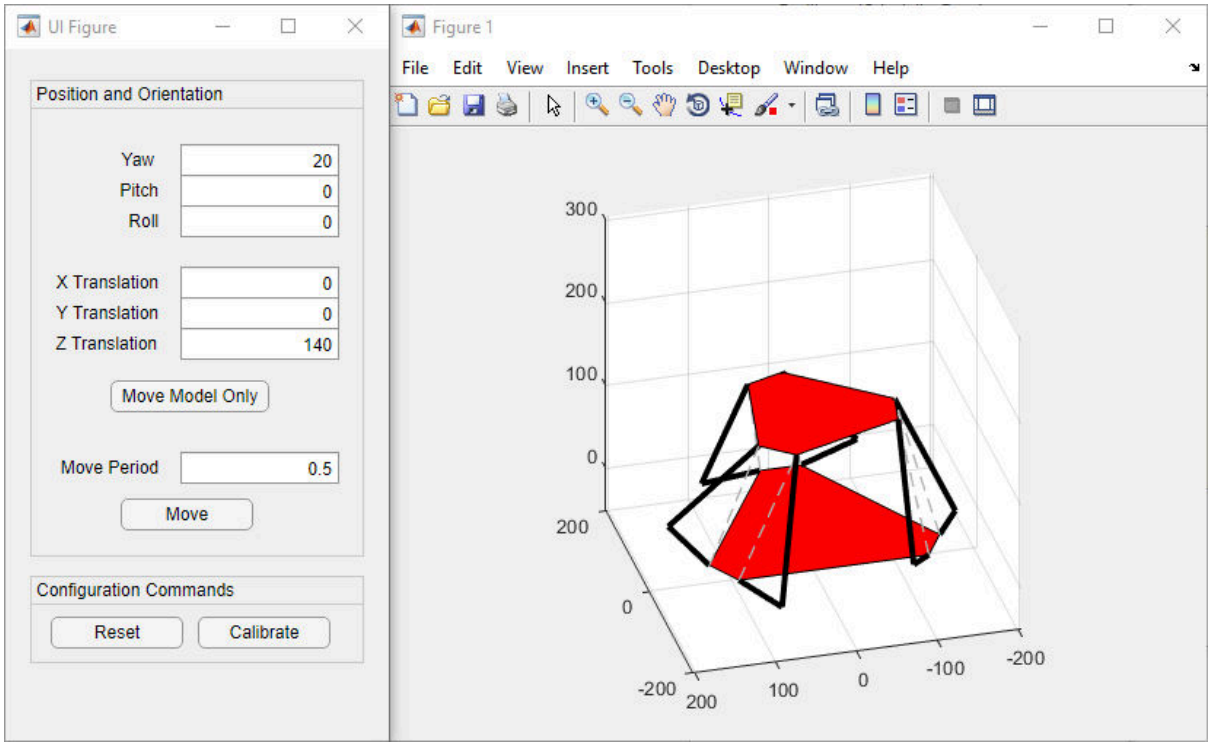

Figure 3.35: A screenshot of the MATLAB application used in debugging and calibrating the platform

It should be noted that the intended use of the platform is for a user to write their own code which sends commands to the platform. A MATLAB code template has been provided with the project allowing for anyone to easily make their own platform simulation routines. Please refer to the build guide in Appendix B for links to the necessary files and a guide on how to use the template.

3.6 The Building-Development Process

This section describes the process taken in building the platform after the initial design and explains the different problems that were faced and how they were overcome. This chapter **does not** explain how every part fits together nor does it provide links to the resources required to build the platform. For this, please see the **Build Guide** in Appendix B which describes each step required to build the platform, connect up the electronics, flash code to the microcontroller and get started with running simulations on the platform. The report was split up this way to make it easy for anyone wanting to build their own Stewart platform to follow a simple guide rather than having to read the entire design section of the report.

3.6.1 Assembly

The first parts created for the platform were the legs. This required the joints to be 3D printed and then assembled. There were several problems with the first iterations of some of the parts. For example, the motor shaft mount which connected leg to the motor was originally printed as a hollow shell with small supports inside for support. This was not strong enough and resulted in one of the parts breaking as shown in Fig. 3.36. While attempting to glue the part back together, the super glue which was used dissolved part of the plastic. This is shown in Fig. 3.37. Eventually, the problem was solved by printing each mount as one solid piece of plastic and not a hollowed out shell.

Figure 3.36: A picture showing the broken motor shaft mount

Figure 3.37: A picture showing the dissolved motor shaft mount

Another problem with 3D printing is that the parts are not always printed exactly to size. It was necessary for tolerances to be introduced to account for slightly oversized or undersized parts. The 3D printer used for this project was an Ultimaker 2+ and it was determined that a tolerance of $0.2mm$ was needed for all holes or slots in parts that were made.

The results of not taking tolerances into account can be seen in the design of the first aluminium rod sleeve. The rod sleeve which was designed to fit between the bearings and aluminium rod, and had an inner diameter that was too small for the rod and an outer diameter that was too large to fit into the bearing. When trying to force the sleeve onto the rod, the plastic cracked, as shown in Fig. 3.38. This was fixed by adjusting the inner diameter to be wider and outer diameter smaller, both by $0.2mm$.

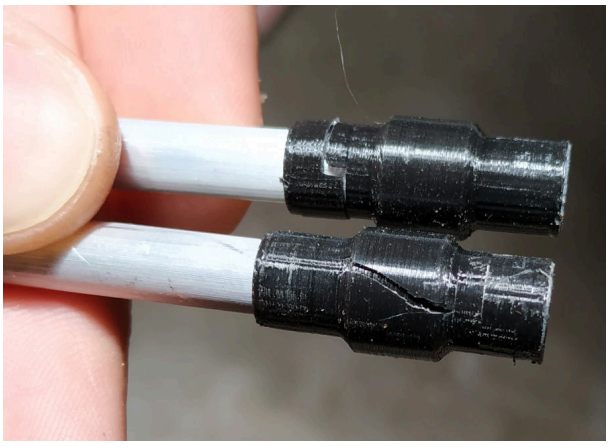

Figure 3.38: A picture showing the cracked rod sleeve

After these problems had been solved, a first prototype of the platform was built, shown in Fig. 3.39. This prototype had smaller base, longer leg linkages and used smaller

motors than those which were used in the final design. The stepper motors were then connected up to the stepper motor drivers via a breadboard. The stepper motor drivers were wired up to a modified ATX computer power supply and the STM32F4 Discovery microcontroller. The wiring during the this stage of the project is shown in Fig. 3.40

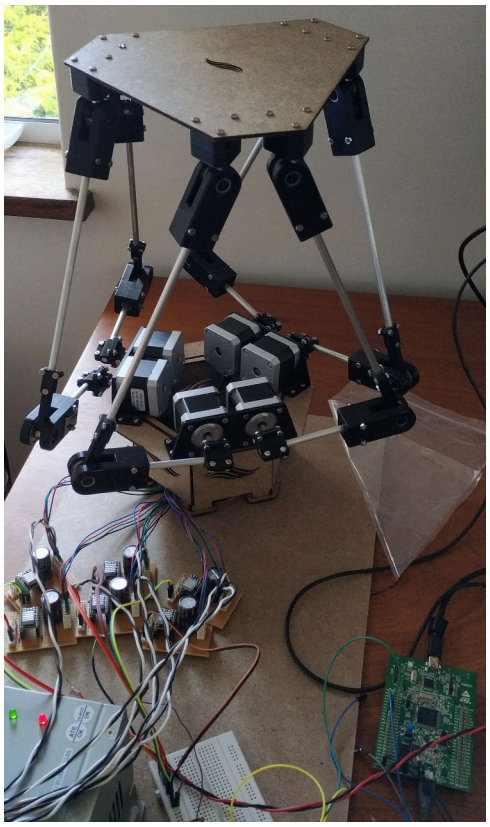

Figure 3.39: A picture showing the first Stewart platform design

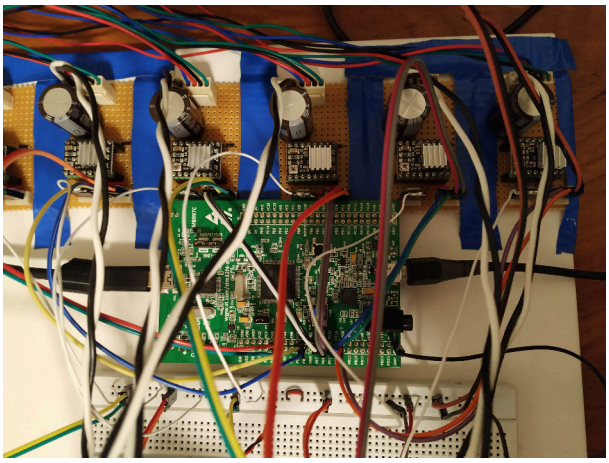

Figure 3.40: A picture showing the wiring for the Stewart platform electronics

This was when the first design changes were made. It was apparent that the small stepper

motors, while being able to provide enough torque to the platform while in a stationary position or when moved slowly, sometimes slipped when the platform's weight was shifted onto them and they were required to move quickly. To remedy this, larger stepper motors were used. As well as this, the length of the lower leg linkages was reduced in order to reduce the torque on the motors.

It was at this stage when a bug in the software was also noticed causing the 6th motor to move to random positions whenever a move command was issued to the platform. The cause of this was due to carriage return and line feed characters which were appended to serial commands by MATLAB's serial connection interface, being interpreted as a motor position for the 6th motor. Once this was found, it was easily remedied by adding a delimiting character at the end of all the commands sent to the microcontroller from the PC.

Besides the adjustments to the leg linkages and stepper motors, the next iteration of the project included a base enclosure which housed the electronics. This is shown in Fig. 3.41 and Fig. 3.42.

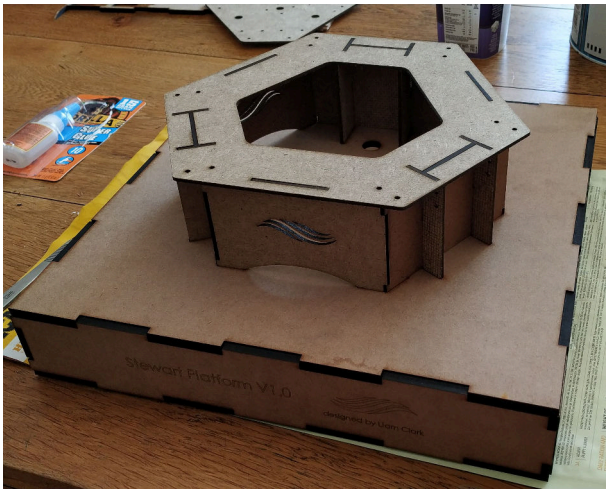

Figure 3.41: A picture showing the second Stewart platform design being built

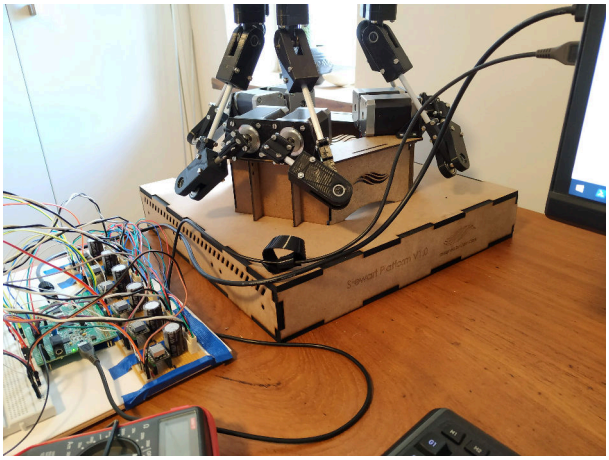

Figure 3.42: A picture showing the second Stewart platform design being tested

After testing the second platform, it was time to enclose all the electronics and add some finishing touches. When the power supply was bought, it was taller than expected and so a final design modification required that the sides of the base enclosure be heightened. Once the heightened sides of the enclosure had been laser cut, laser cut parts were spray painted matte black, shown in Fig. 3.43. After the spray paint had dried, the platform was assembled and where necessary, parts were glued together. The finished assembled enclosure is shown in Fig. 3.44.

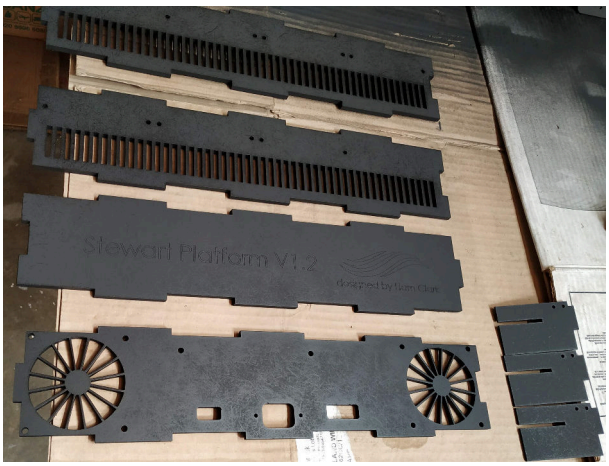

Figure 3.43: A picture showing the laser cut parts being spray painted

Figure 3.44: A picture showing the final design being assembled

Electronics were then wired into the base enclosure ensuring that the cabling was neat and safe. Subsystems were tested individually, ensuring that each subsystem worked correctly before being integrated into the overall system. This first systems that were tested were the power supply and cooling systems, shown in Fig. 3.45. Thereafter, the stepper motor drivers and microcontroller were integrated, shown in Fig. 3.46.

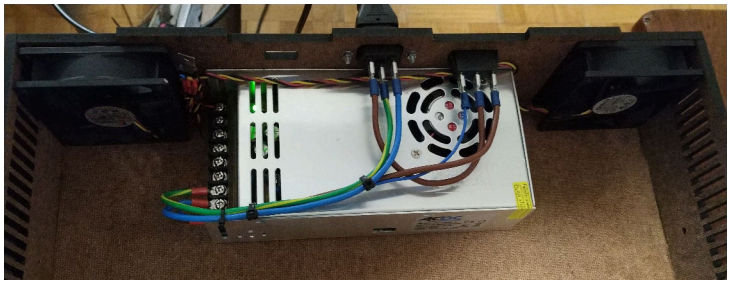

Figure 3.45: A picture showing the power supply and cooling fan wiring inside the base enclosure

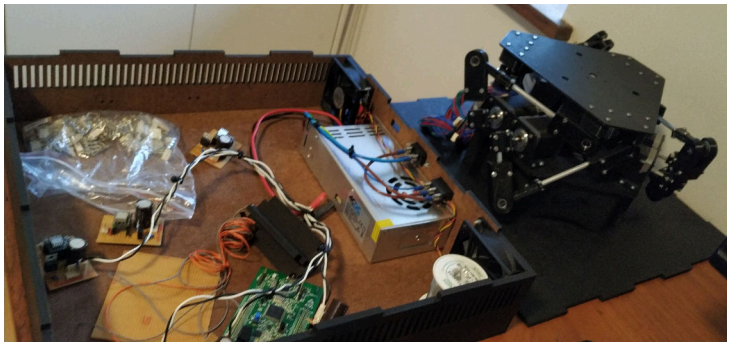

Figure 3.46: A picture showing the stepper motor drivers being connected to the power supply and microcontroller

The final assembled design is shown in action in Fig. 3.47 and 3.48.

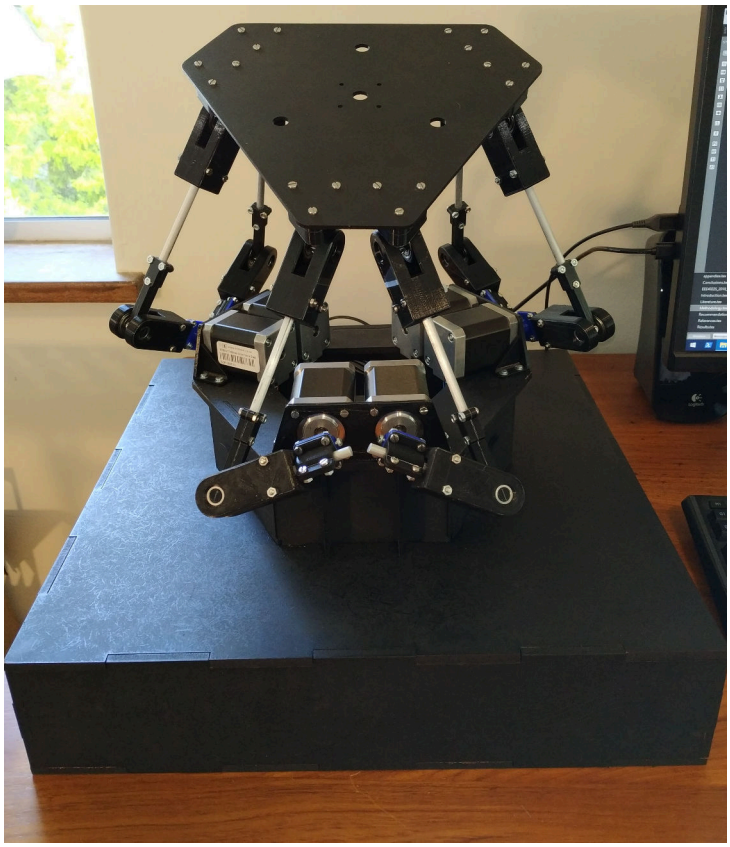

Figure 3.47: A picture showing the front of the final Stewart platform

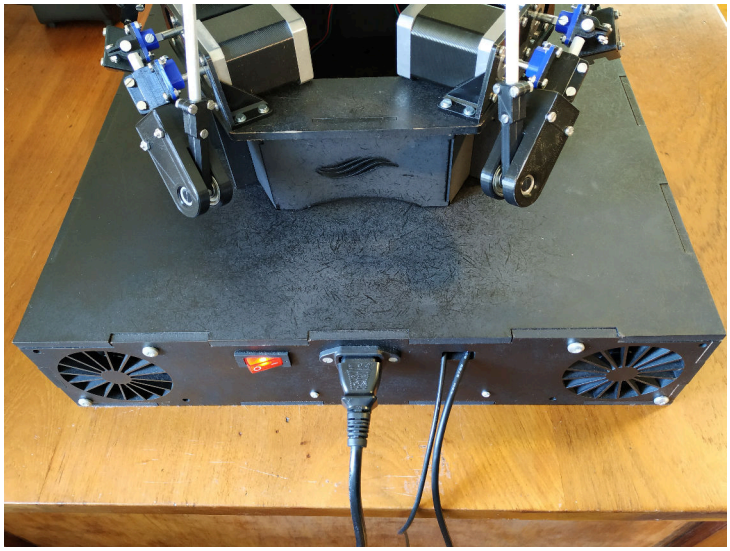

Figure 3.48: A picture showing the back of the final Stewart platform

3.6.2 Bill of Materials

Item	Cost	Quantity	Total Cost	Supplier
Stepper Motors (NEMA-17, 60mm length)	R298.20	6	R1789.20	UCT
Power Supply (350W, 12V)	R799.48	1	R799.48	ACDC Dynamics
Microcontroller (STM32F4 Discovery)	R379.29	1	R379.29	UCT
Fans (12V, 80mm x 80mm, 2500RPM)	R32.00	2	R64.00	UCT
Stepper Motor Drivers (Polulu DRV-8825)	R38.16	6	R76.32	Banggood
Bearings (608zz)	R10.87	60	R652.20	Silkuni
3D Printed Parts	R710.00	$710g$	R710.00	UCT
Hardboard (3mm x 1220mm x 620mm)	R96.00	1	R96.00	Mica
Hardboard (6mm x 1220mm x 620mm)	R156.00	2	R312.00	Mica
Matte Black Spray Paint	R128.00	1	R128.00	Mica
Voltage Regulators (LM7805)	R3.64	7	R25.48	UCT
Capacitors (4700uF, electrolytic)	R2.31	6	R13.86	UCT
Capacitors (0.33uF, ceramic)	R0.12	7	R0.84	UCT
Capacitors (0.1uF, ceramic)	R0.19	7	R1.33	UCT
Molex Connector (2 pin, male)	R0.56	18	R10.08	UCT
Molex Connector (2 pin, female)	R0.56	12	R6.72	UCT
Molex Connector (4 pin, male)	R0.56	6	R3.36	UCT
Molex Connector (4 pin, female)	R0.56	6	R3.36	UCT
Pin Header (1 x 8 pins, female)	R1.17	12	R14.04	UCT
Pin Header (2 x 25 pins, female)	R2.07	2	R4.14	UCT
Veroboard (100mm x 300mm sheet)	R60.00	2	R120.00	UCT
Kettle Plug Port	R14.50	1	R14.50	Mica
On/Off Switch (230V, 10A)	R12.50	1	R12.50	Mica
Crimp Spade Connector	R1.50	5	R7.50	Mica
Terminal Spade Connector	R1.50	6	R9.00	Mica
M3 Bolts (32mm)	R0.40	24	R9.60	Bolt It
M3 Bolts (25mm)	R0.40	18	R7.20	Bolt It
M3 Bolts (16mm)	R0.40	12	R4.80	Bolt It
M3 Bolts (10mm)	R0.40	36	R14.40	Bolt It
M3 Bolts (8mm)	R0.40	36	R14.40	Bolt It
M3 Bolts (6mm)	R0.40	24	R9.60	Bolt It
M4 Bolts (8mm)	R0.50	2	R1.00	Bolt It
M4.5 Screw (16mm)	R0.50	8	R4.00	Bolt It
M3 Hex Nuts	R0.20	126	R25.2	Bolt It
Total Cost			R4964.11	

Chapter 4

Results

In simulating the platform, the MATLAB model was used. The mechanical design allowed for the leg linkage lengths to be adjusted by changing the length of the aluminium rod in the linkage; however, only a single design was simulated and tested. The design which was simulated and tested had the following configuration shown in Table 4.1.

Property	Value
Base radius	$152mm$
Angle between actuators in a pair	$18.92°$
Platform radius	$103mm$
Angle between platform-leg joints	$28.07°$
Lower leg linkage length	$73mm$
Upper leg linkage length	$165mm$

Table 4.1: A table showing the properties of the platform which was designed, simulated and tested

For this configuration of the platform, a home position was defined such that the roll, pitch, yaw, x and y translation of the platform were all zero. The height of the platform was such that the angle of each of the lower leg linkages was $0°$ relative to the base's x-y plane. This corresponded to an approximate height $z = 138mm$.

4.1 Simulation Results

4.1.1 Workspace Analysis

In order to determine a theoretical workspace for the platform, an approach similar to that of Bohigas et al. [38] was taken where discrete vectors containing the roll, pitch, yaw, x, y and z translation were tested to see if they were valid or not for a specific platform configuration. By having a large number of test points and plotting a boundary around all of the valid points, an approximate multi-dimensional "volume" representing the achievable workspace could be computed.

Commonly, three of the six DOF would be fixed, allowing for a 3D volume to be created representing achievable positions and/or orientations. The two most useful workspaces to compute were the constant-position and constant-orientation workspaces as they are intuitive when viewed as a 3D volume.

To determine the constant-orientation workspace, the yaw, pitch and roll of the platform were all kept equal to zero degrees while the position of the platform was varied. A MATLAB script was written to test whether a set of different positions represented as points in a 3D plot were achievable by the platform. Out of a set of test points, those which were valid (were achievable) were kept whilst the others were discarded. Once a set of valid points was determined, a boundary was plotted around the points encompassing them and creating a boundary volume. This was then plotted and is shown in Fig. 4.1 and Fig. 4.2. It should be noted that only values for which $z \geq 0$ were plotted as the platform configuration would not allow for the platform to be lower than the base.

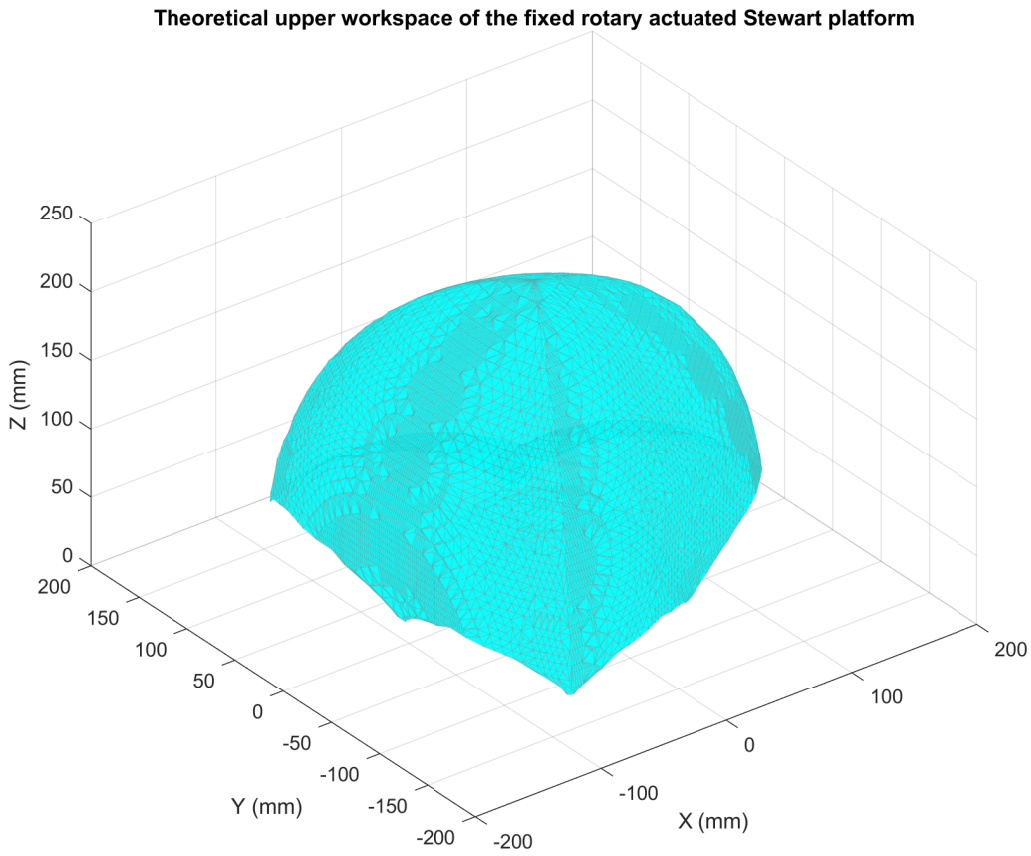

Figure 4.1: A plot from MATLAB showing the upper constant-orientation workspace of the platform

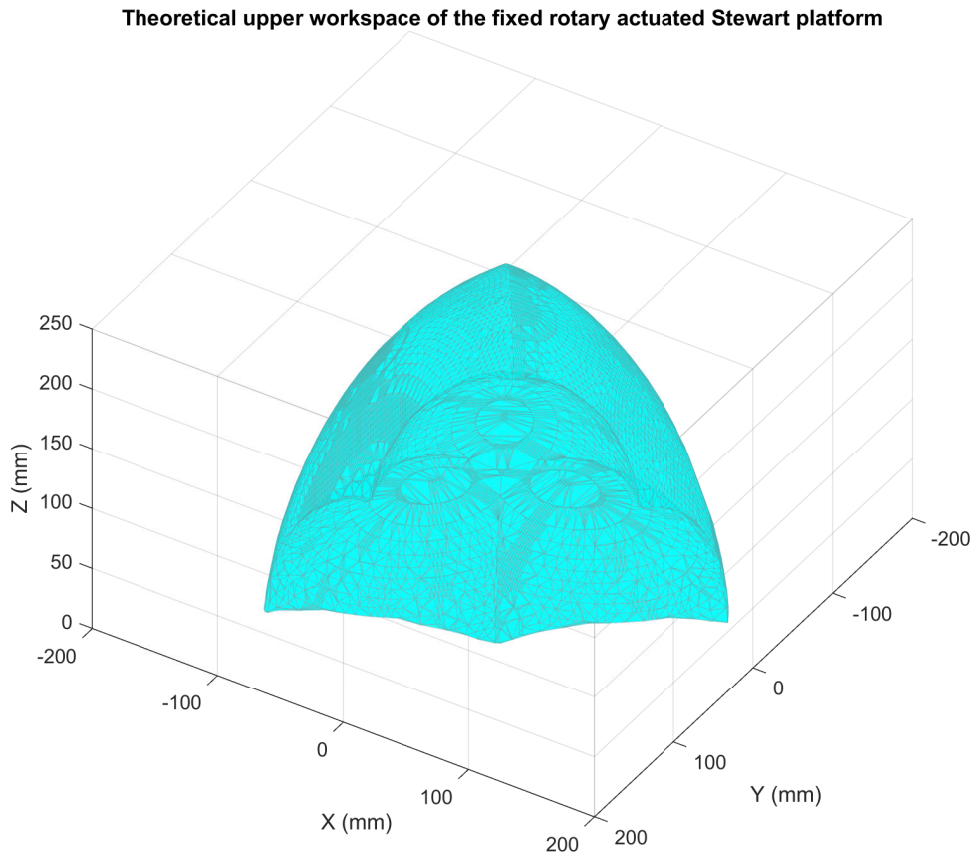

Figure 4.2: A plot from MATLAB showing the upper constant-orientation workspace of the platform from underneath

A constant-position workspace was also plotted. This was done by setting the x and y coordinates to 0 and the z coordinates to $138mm$ above the base such that the lower leg linkages would be horizontal when the roll, pitch and yaw were all zero (the home position of the platform). The resulting constant-position workspace is shown in Fig. 4.3 and 4.4.

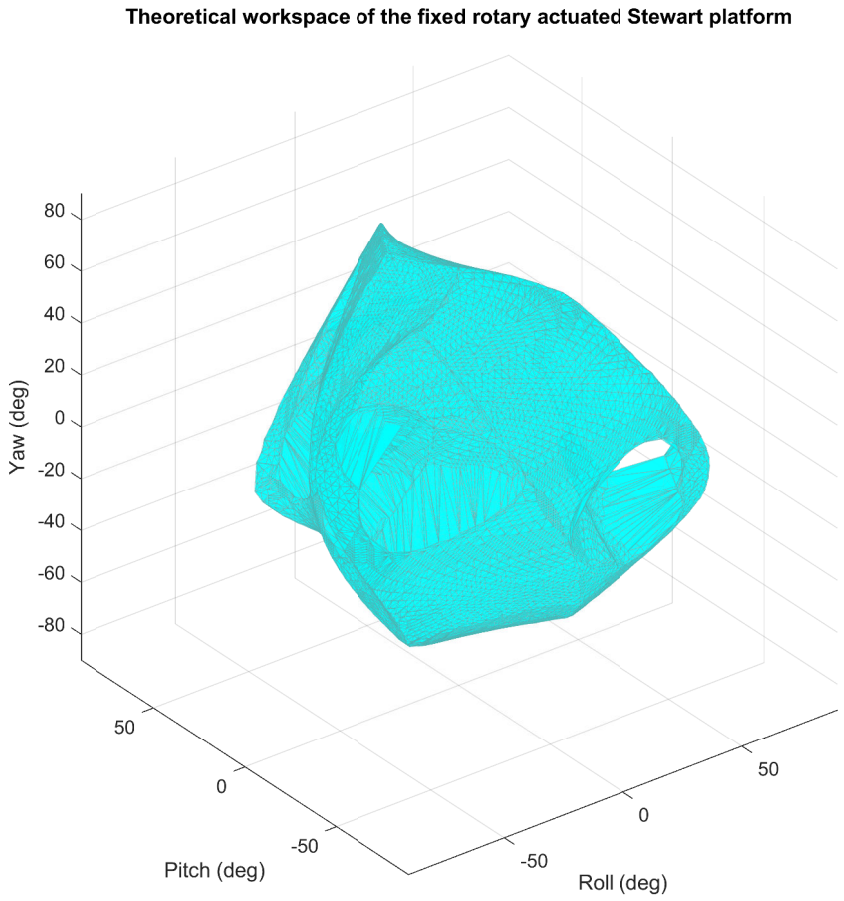

Figure 4.3: A plot from MATLAB showing the constant-position workspace of the platform

An interesting effect was noticed in the constant-position workspace plots where it can be seen that the central section of the volume is mostly contained within a 80° to 100° range for the roll, pitch and yaw; however, three outer loops can be seen containing orientations which are theoretically achievable but have large pitch and roll values. In order to reach these orientations in the loops, one would have to navigate the platform into the orientation and out of it while avoiding invalid positions.

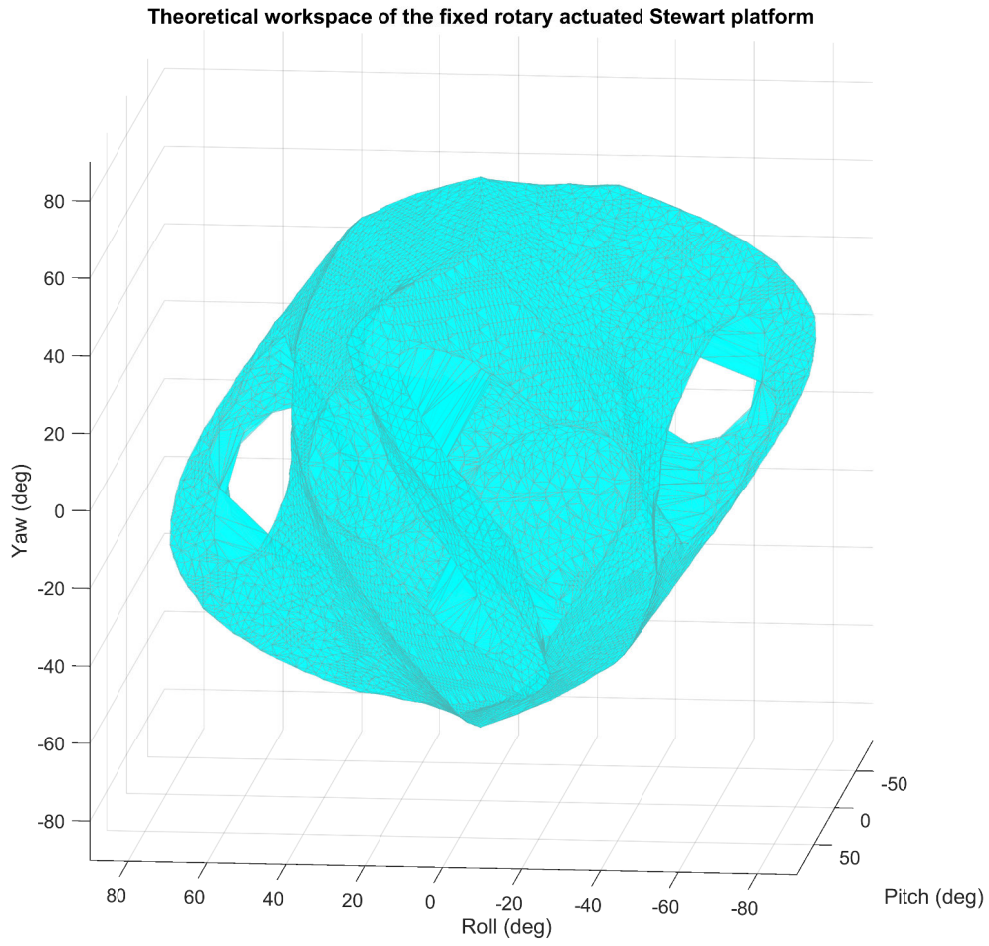

Figure 4.4: A plot from MATLAB showing the constant-position workspace of the platform from the side

4.2 Experimental Results

4.2.1 Types of Tests

In order to test the platform, tests were separated into two categories, namely static and dynamic tests. Static tests refer to tests in which the performance of the platform was measured while in a stationary position and orientation. Dynamic tests were used to evaluate the platform's performance while in motion.

There were several metrics which were to be determined by the tests. These were:

- The accuracy of the platform which would determine how closely the platform's measured position or orientation output was for a corresponding input
- The precision of the platform which would determine the platform's ability to accurately repeat movements
- The robustness of the platform which would determine how the platform's response changed when it was placed under load

4.2.2 Testing Apparatus Setup for Static Tests

Testing the platform's static position and orientation required a testing rig to be built. A smartphone was placed $30cm$ away from the Stewart platform and used to capture images of the platform in various positions and orientations. In order to be able to determine the platform's orientation or position, calibration cards (shown in Fig. 4.5) were printed. A 360° compass calibration card was used in orientation tests and a $1mm$ spaced grid was used in position tests.

The smartphone used in these tests was a Xiaomi MiA2 which had a 12MP front facing camera.

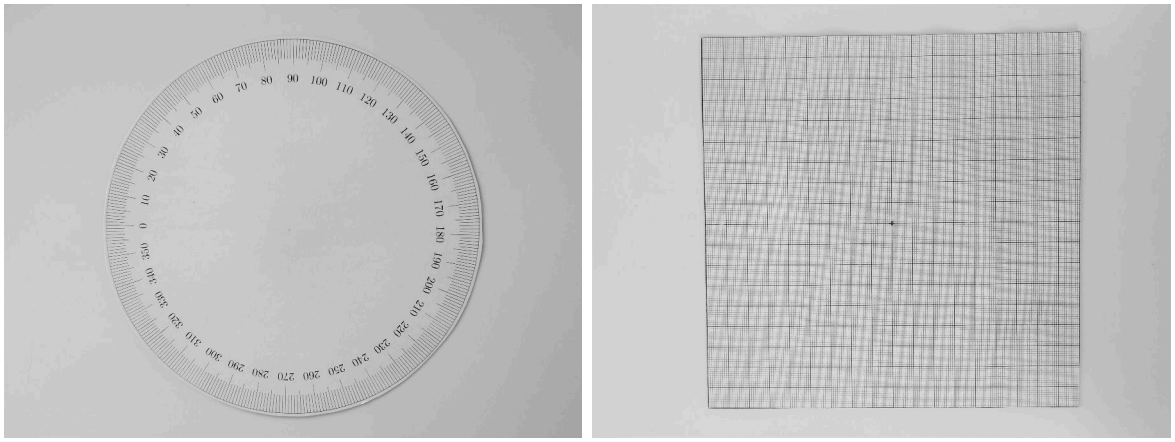

Figure 4.5: A picture of the calibration compass (left) and grid (right)

The calibration cards were mounted on the platform with the smartphone's camera pointing directly at the calibration card. A stand was used as a mount for the smartphone. The smartphone was connected to a computer via a USB cable allowing for it to be controlled remotely through the Android Debug Bridge (ADB). Shown in Fig. 4.6 is the yaw testing rig and the setup used for the roll testing rig.

Figure 4.6: Pictures showing the setup for the yaw testing rig (left and center) and the roll testing rig calibration card configuration (right)

There were 5 separate testing rigs used. Single tests were used for roll, pitch and yaw as well as the z translation. The x and y translation test was combined as data could be gathered for both tests with the same testing rig setup.

A MATLAB script was written allowing for tests to be automated. For each test, the script would calibrate the platform and move it into a home position. Thereafter, the script would move the platform into various positions or orientations and trigger the smartphone's camera to take a picture. After a set of photos had been taken, the MATLAB script would rename all the images according to the orientation or position which the platform was in for that specific picture. This is shown in Fig. 4.7. The automation of this process allowed for many tests to be performed as well as ensuring that no movement in the testing rig occurred during a test.

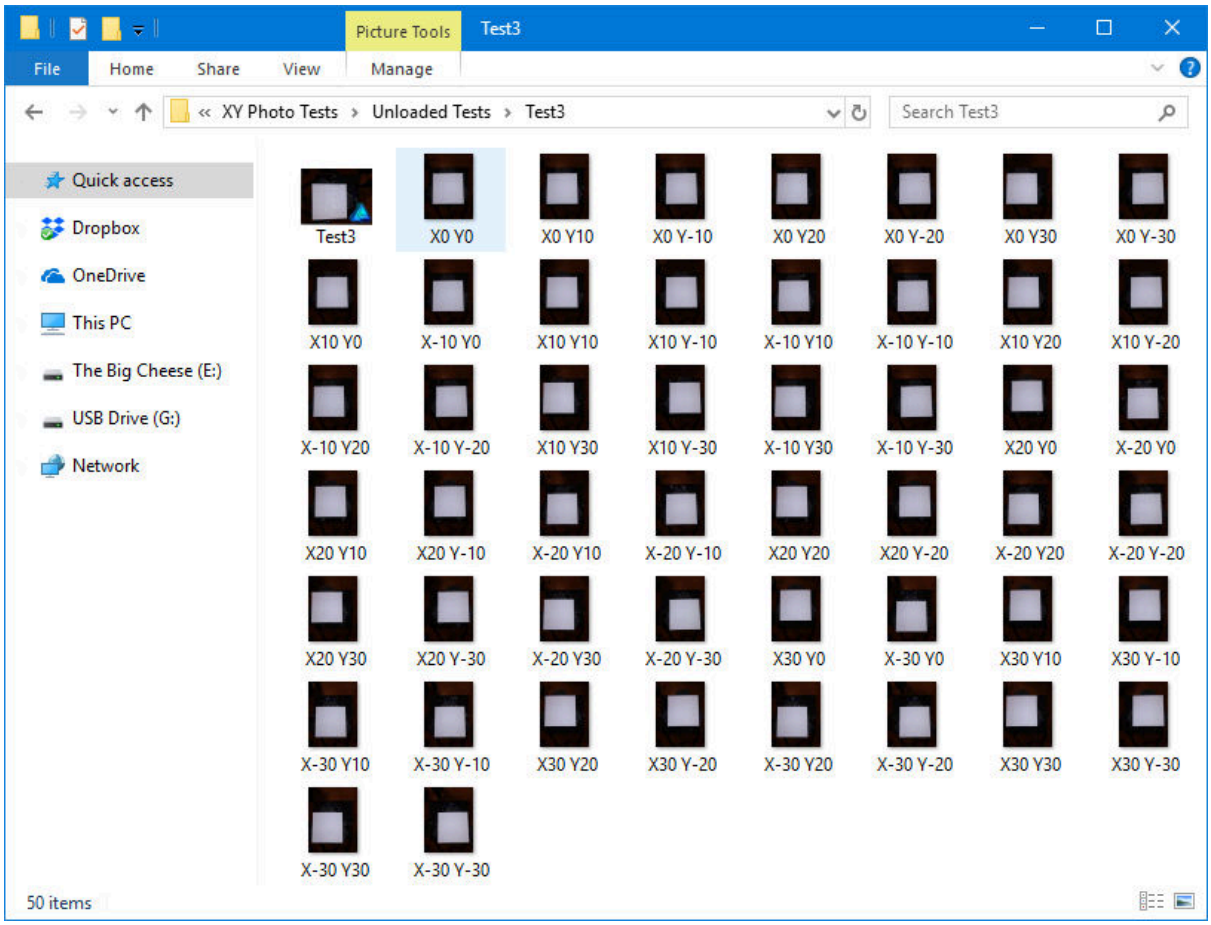

Figure 4.7: A screenshot showing the images captured after an automated test

After the images were captured, they were brought into an image editing program called Affinity Designer. For each test, every photo taken was imported into the program as a separate layer and stacked on top of each other. For the position tests, a grid was drawn and aligned to the home position image. For the orientation tests, a line was drawn representing the 0° mark with all other images taken being centred around the compass's origin. This is shown in Fig. 4.8.

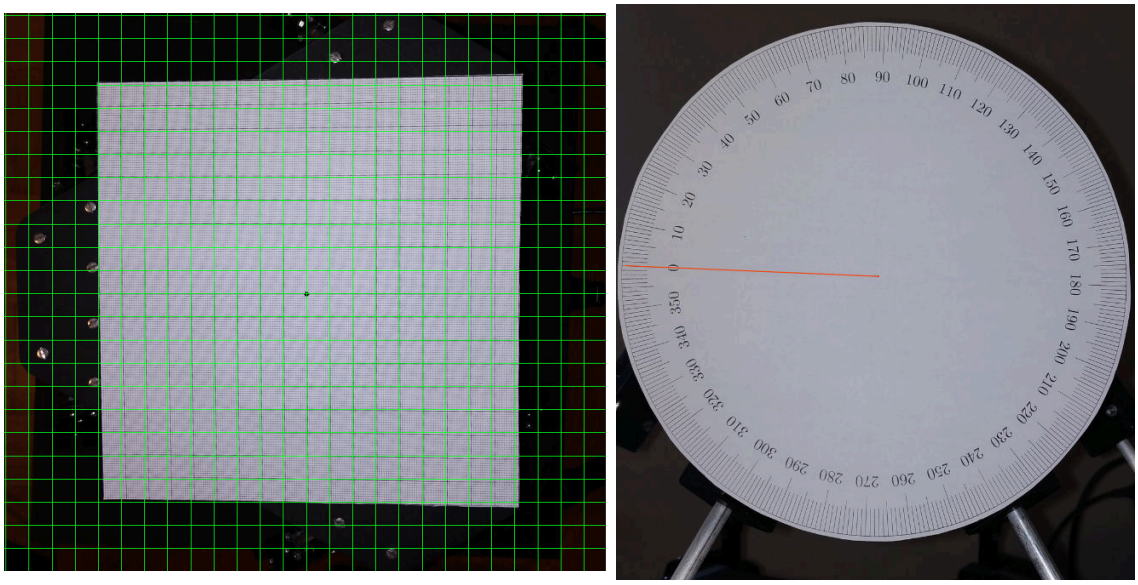

Figure 4.8: A picture of the x-y grid (left) and compass (right) in Affinity Designer

By viewing the various layers representing different positions or orientations while keeping the grid or 0° line visible on top, it was possible to read off the various orientations and positions of the platform. For translational positions, the grid was made up of $1mm$ spaced lines. For rotational orientations, the compass provided 1° accuracy. Each measurement was associated with the nearest grid line or compass line meaning that the uncertainty associated with these measurements was $\pm 0.5mm$ for position measurements and $\pm 0.5°$ for orientation measurements.

Examples showing how measurements were determined from tests are shown below in Fig. 4.9 and 4.10.

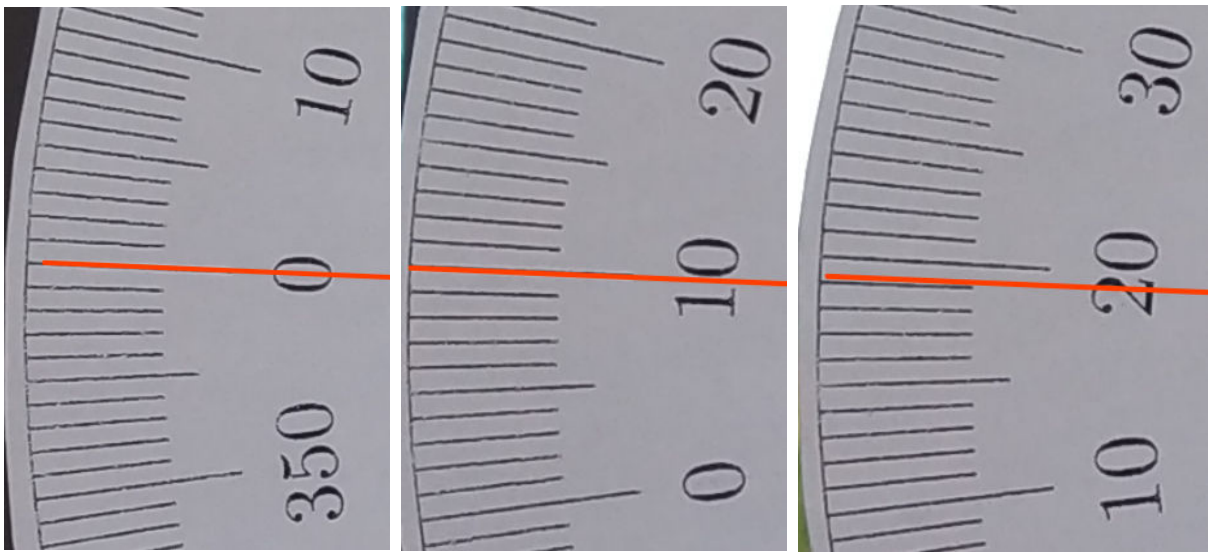

Figure 4.9: A screenshot showing measurements for 10° increases in roll from left to right

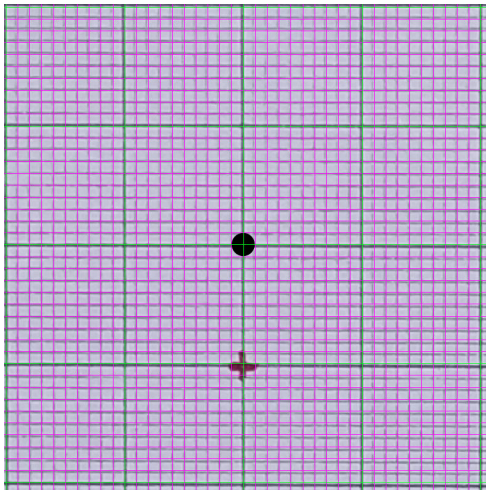

Figure 4.10: A screenshot showing a measurement for a coordinate $x = 0$ and $y = -10$

4.2.3 Static Tests Results

For all of the static tests performed, in order to measure the precision of the measurements, each test was repeated three times. To determine the robustness of the platform, loaded tests were also performed where a 588g weight, shown in Fig. 4.11, was placed on the platform.

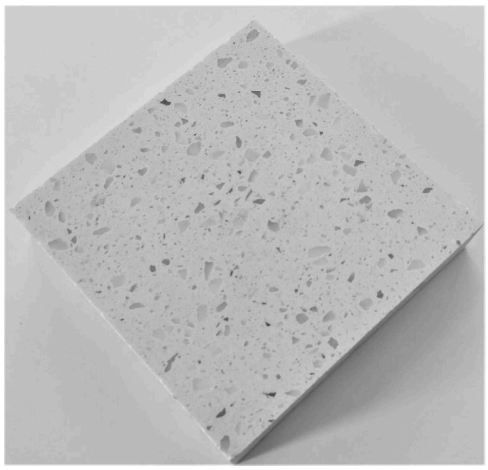

Figure 4.11: A picture of the 588g load which was placed on the platform for loaded tests

For the static tests, the following tests were performed:

- Roll (total of 36 discrete measurements)
  - Unloaded tests: $\times 3$ tests, $-30°$ to $30°$ in $10°$ increments

  - Loaded tests (588g load): $\times 3$ tests, $-20°$ to $20°$ in $10°$ increments

- Pitch (total of 30 discrete measurements)
  - Unloaded tests: $\times 3$ tests, $-20°$ to $20°$ in $10°$ increments
  - Loaded tests (588g load): $\times 3$ tests, $-20°$ to $20°$ in $10°$ increments
- Yaw (total of 36 discrete measurements)
  - Unloaded tests: $\times 3$ tests, $-30°$ to $30°$ in $10°$ increments
  - Loaded tests (588g load): $\times 3$ tests, $-20°$ to $20°$ in $10°$ increments
- X-Y translation (total of 222 discrete measurements)
  - Unloaded tests: $\times 3$ tests, $-30mm$ to $30mm$ in $10mm$ increments for both x and y including all permutations (total of 49 measurements per test)
  - Loaded tests (588g load): $\times 3$ tests, $-20mm$ to $20mm$ in $10mm$ increments for both x and y including all permutations (total of 25 measurements per test)
- Z translation (total of 54 discrete measurements)
  - Unloaded tests: $\times 3$ tests, $120mm$ to $200mm$ in $10mm$ increments
  - Loaded tests (588g load): $\times 3$ tests, $120mm$ to $200mm$ in $10mm$ increments

Static Roll Test Results

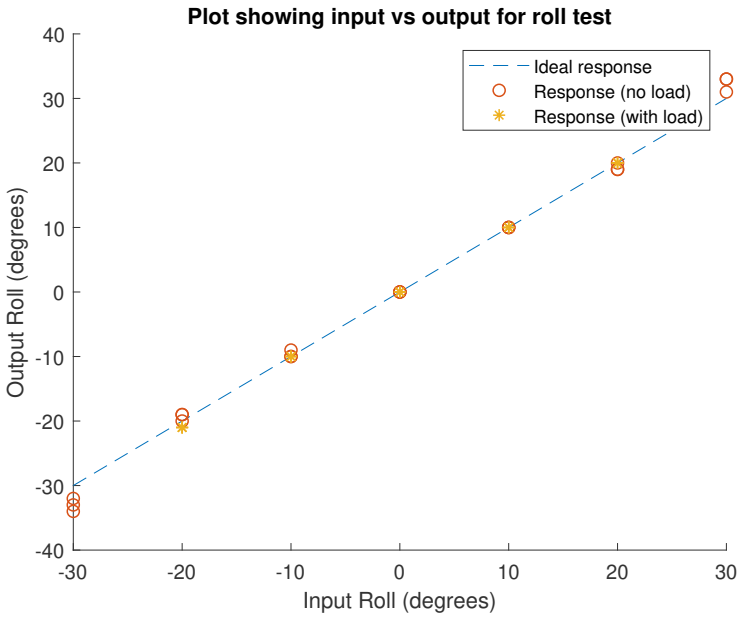

Figure 4.12: A MATLAB plot showing the roll test results

Static Pitch Test Results

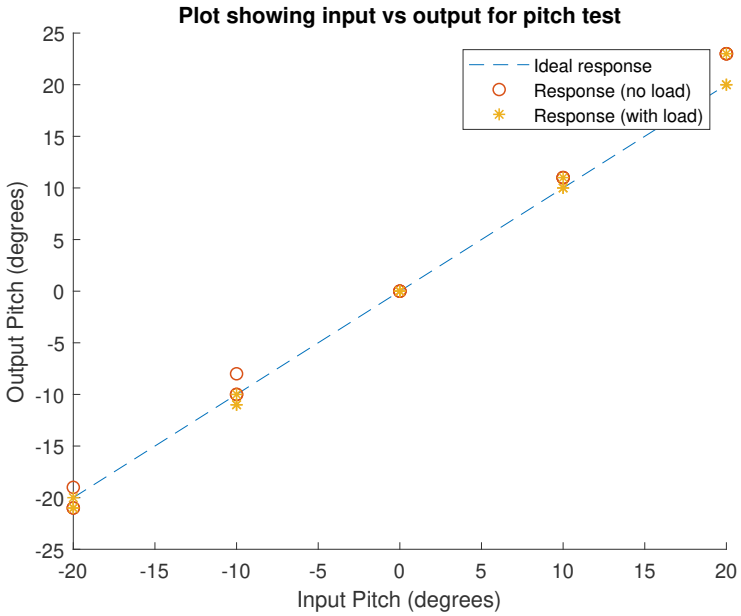

Figure 4.13: A MATLAB plot showing the pitch test results

Static Yaw Test Results

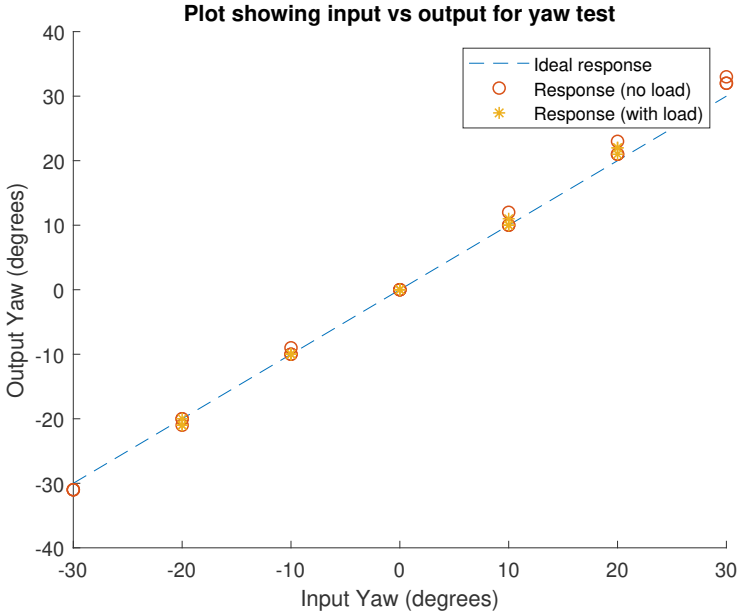

Figure 4.14: A MATLAB plot showing the yaw test results

Static X-Y Translation Test Results

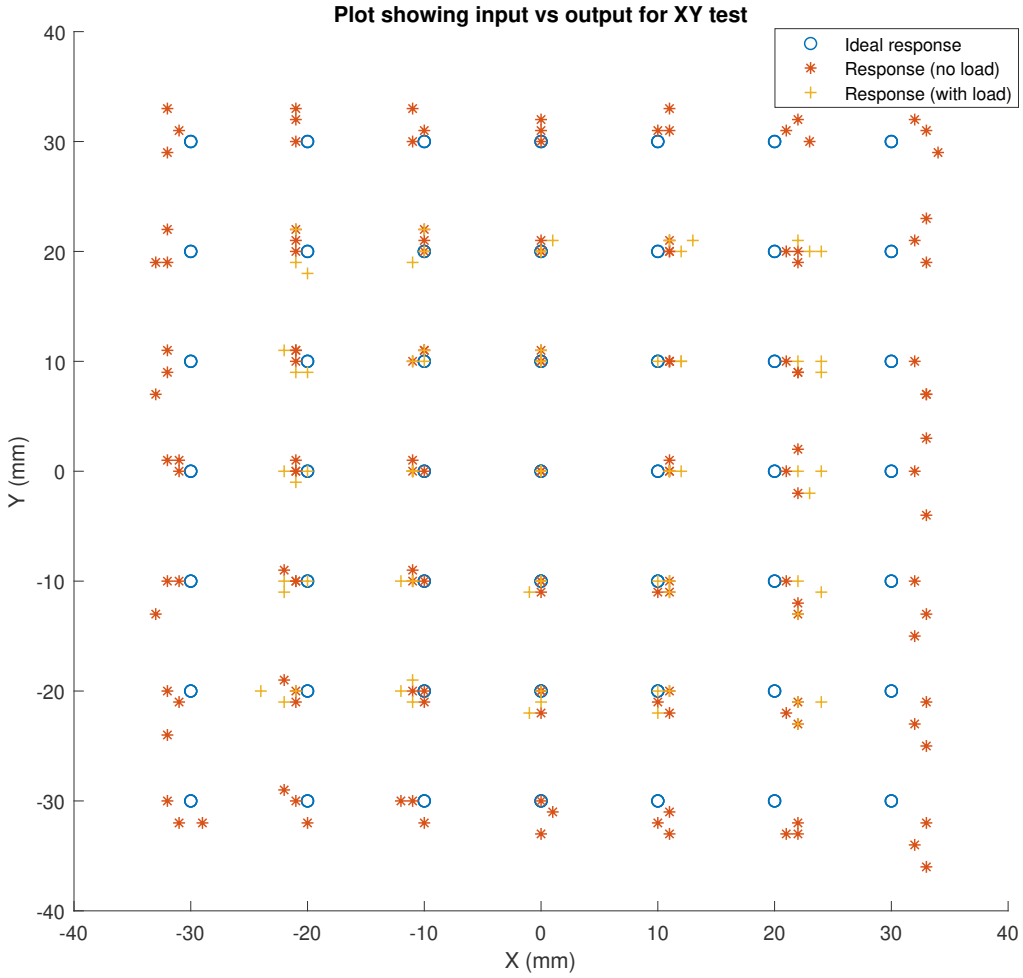

Figure 4.15: A MATLAB plot showing an X-Y grid superimposed with the X-Y translation test results

Static Z Translation Test Results

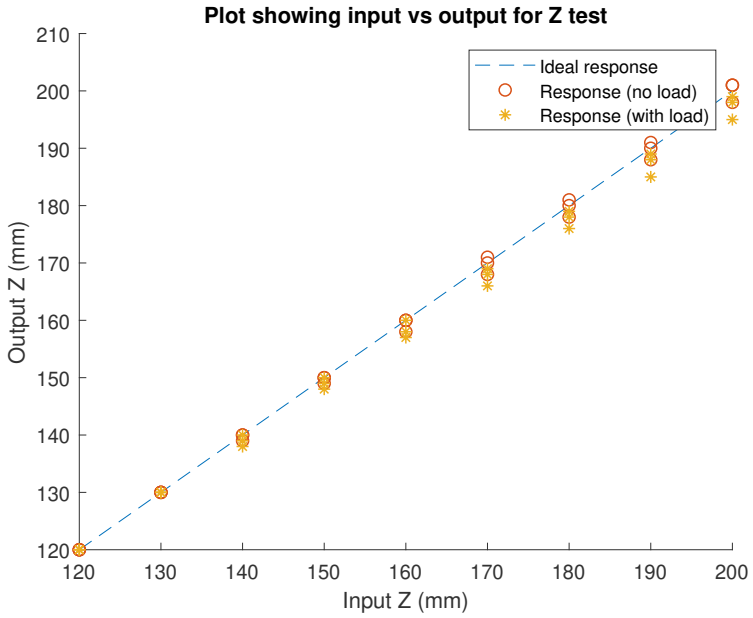

Figure 4.16: A MATLAB plot showing the Z-translation test results

4.2.4 Testing Apparatus Setup for Dynamic Tests

Due to the Stewart platform's main purpose being focused on simulating ocean waves, it was important to test the platform's ability to replicate motion, specifically the roll and pitch of the platform. In order to do this, a smartphone was used in conjunction with MATLAB's mobile application. This would allow the smartphone's sensors' data to be logged. The sensors available for use were the accelerometer, gyroscope, magnetometer and GPS. Initially, it had been planned that the platform's roll, pitch and yaw data would be logged using the smartphone's gyroscope; however, MATLAB's mobile application uses the smartphone's magnetometer to determine the yaw which the smartphone is experiencing. Due to having six stepper motors mounted below the platform, the magnetic field during operation disrupted the magnetometer, preventing any reasonable measurements being taken for the yaw readings. Again, the smartphone used was a Xiaomi MiA2.

In order to test the platform's ability to replicate wave-like motion, tests were developed which would make the platform's pitch and roll track sinusoidal waveforms. The waveforms would increase in frequency such that tests would cover a certain bandwidth. The tracking performance of the platform could then be observed for ranges of frequencies.

The bandwidth of frequencies that the platform needed track would be between $0Hz$ and

$1Hz$. While this would ensure that the platform could track all low frequency waves such as tidal waves and infra-gravity waves, these would not be easily visible in a short period test. The main type of waves the platform would simulate were gravity waves which were generated by wind. These range from approximately $0.03Hz$ to $1Hz$. Ultra-gravity waves have a bandwidth ranging from approximately $1Hz$ to $10Hz$; however, ultra-gravity waves are far smaller in magnitude than the wind generated gravity waves and are not required for simulation purposes. For more information regarding waves please refer back to Fig. 2.2.

The following dynamic tests were performed:

- Pitch
  - Loaded test (198 g load): $-10°$ to $10°$ oscillation, $0.1Hz$ to $1Hz$ linear sweep.
- Roll
  - Loaded test (198 g load): $\times 3$ tests, $-10°$ to $10°$ oscillation, $0.1Hz$ to $1Hz$ linear sweep.
- Pitch and Roll
  - Loaded test (198 g load): $\times 3$ tests, $-10°$ to $10°$ oscillation, $0.1Hz$ to $1Hz$ linear sweep, X-Y phase difference of $0°$, $90°$ and $-90°$.

For all tests, the yaw, x and y translation were all zero. The height of the platform was set to 150mm above the base. Due to the smartphone being placed on the platform, all tests performed were under a load of 198g (the weight of the smartphone and its case).

4.2.5 Dynamic Test Results

Dynamic Roll Test Results

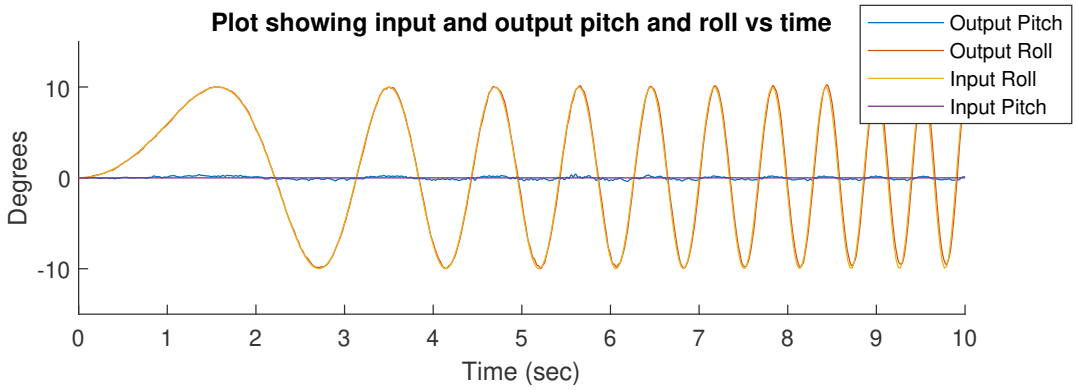

Figure 4.17: A MATLAB plot showing the dynamic roll test 1 results

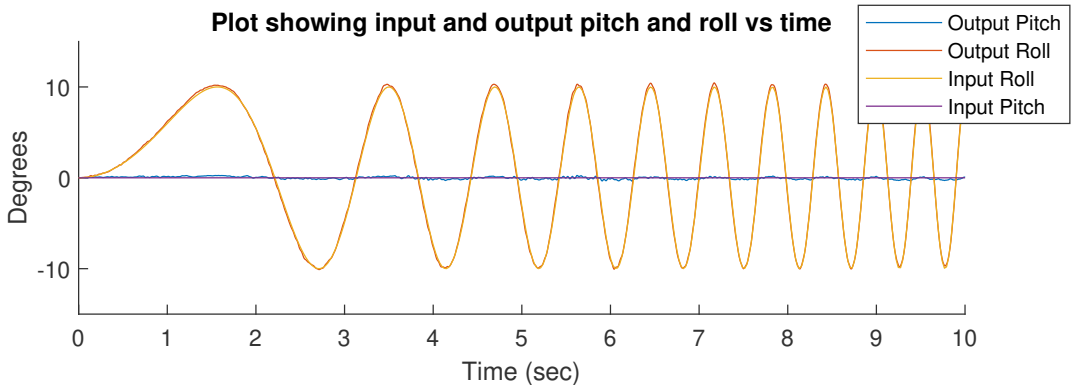

Figure 4.18: A MATLAB plot showing the dynamic roll test 2 results

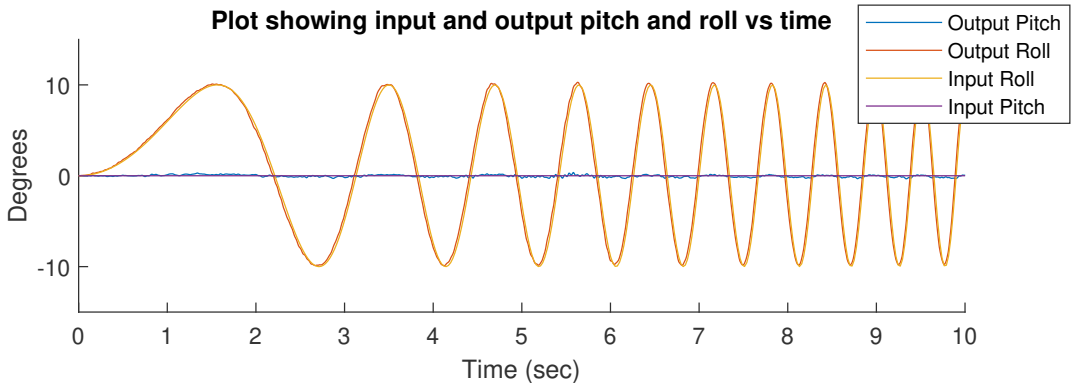

Figure 4.19: A MATLAB plot showing the dynamic roll test 3 results

Dynamic Pitch Test Results

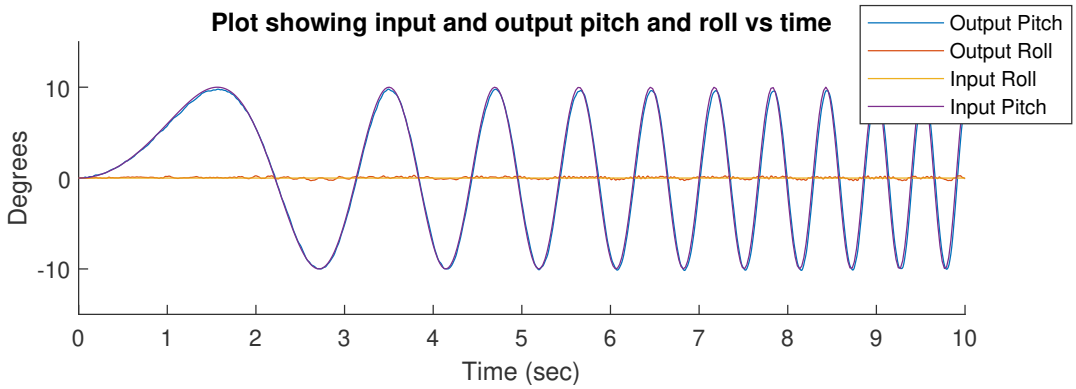

Figure 4.20: A MATLAB plot showing the dynamic pitch test 1 results

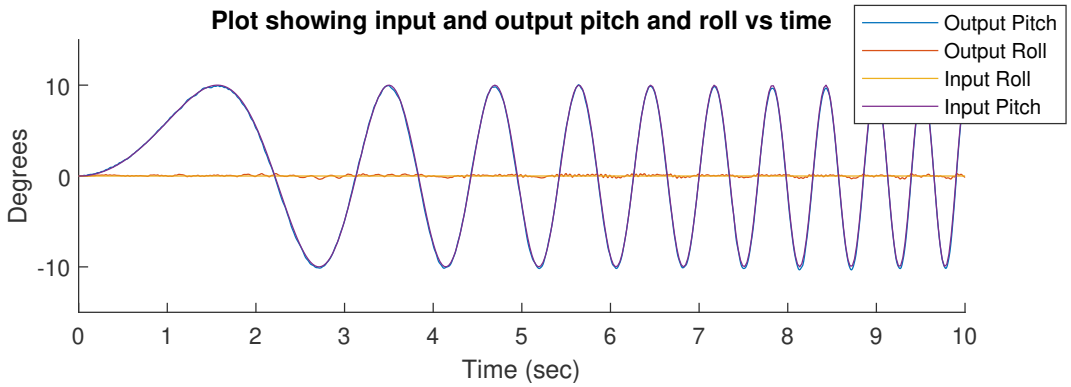

Figure 4.21: A MATLAB plot showing the dynamic pitch test 2 results

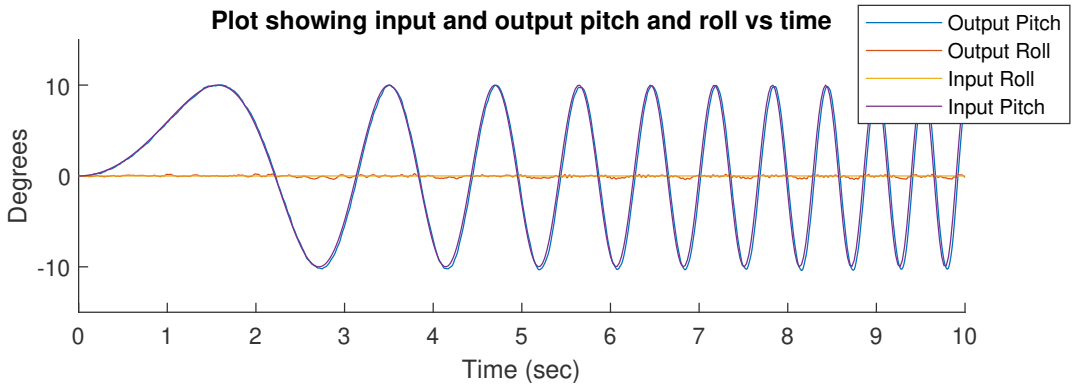

Figure 4.22: A MATLAB plot showing the dynamic pitch test 3 results

Dynamic Roll and Pitch Test Results

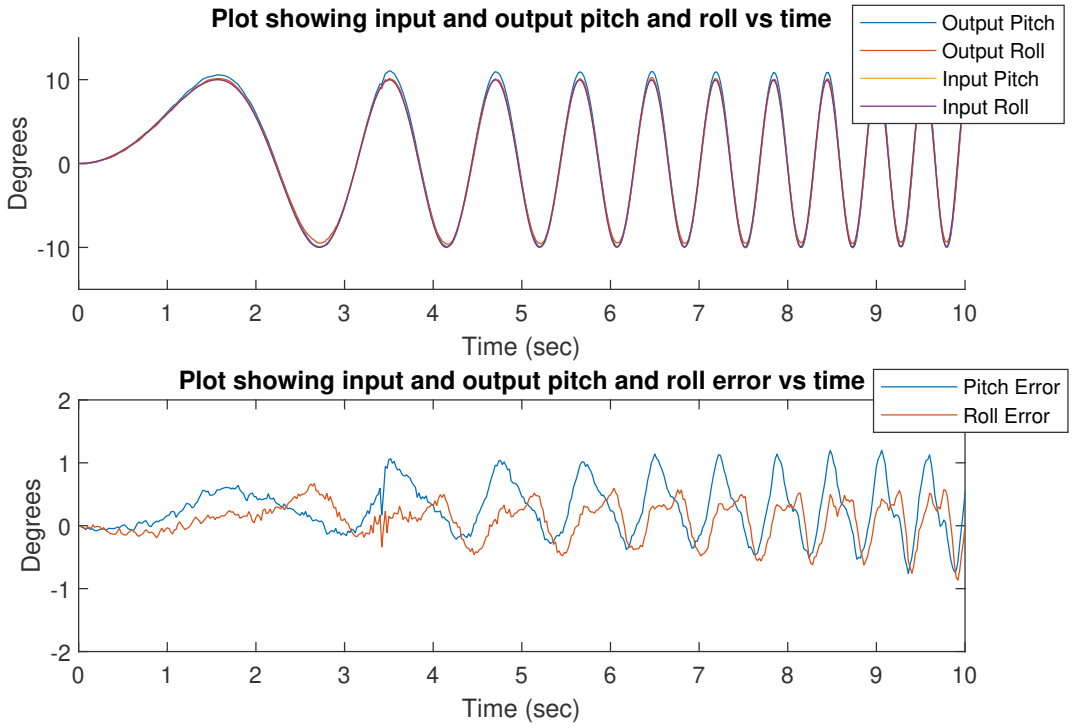

Figure 4.23: A MATLAB plot showing the dynamic roll and pitch test 1 results

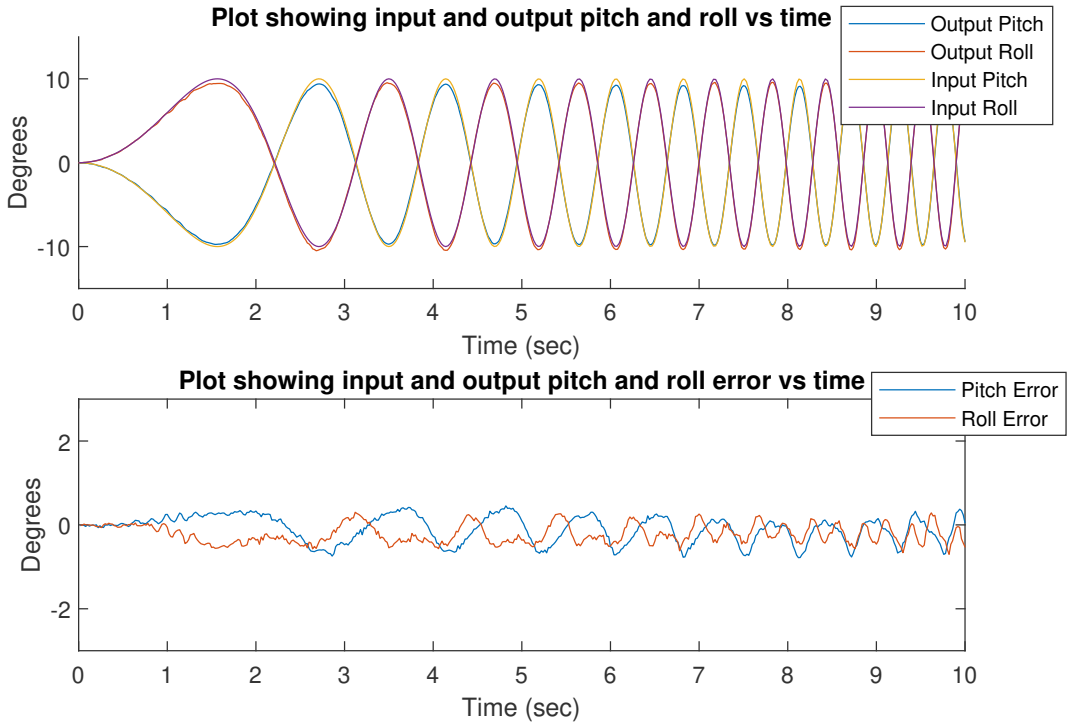

Figure 4.24: A MATLAB plot showing the dynamic roll and pitch test 2 results

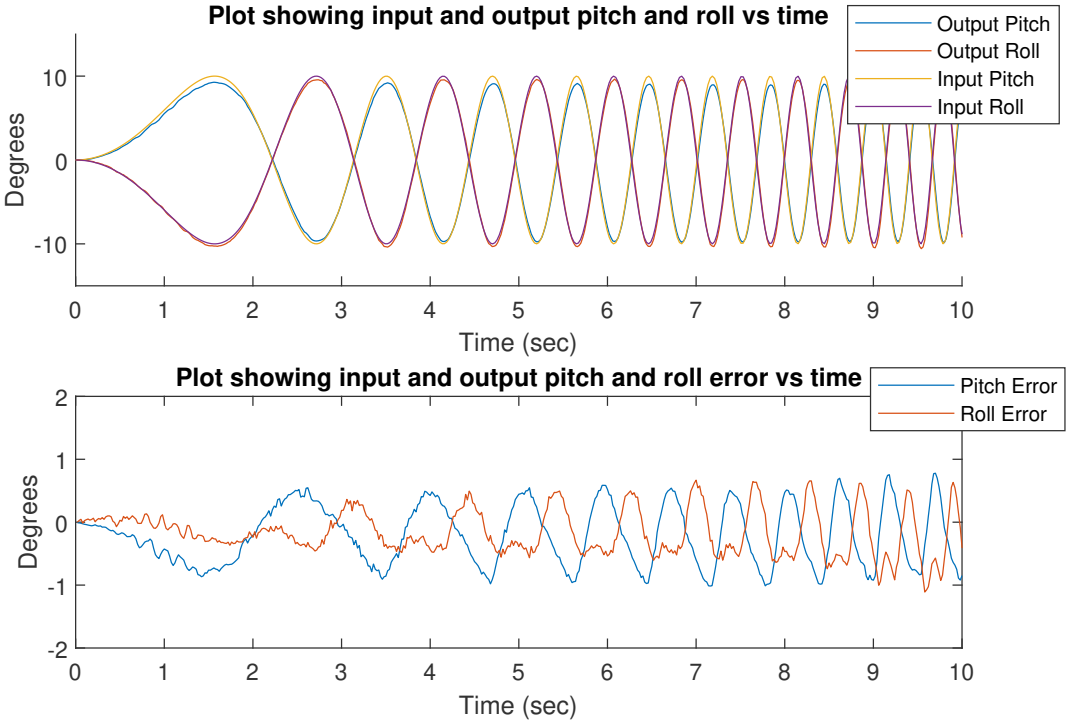

Figure 4.25: A MATLAB plot showing the dynamic roll and pitch test 3 results

4.2.6 Testing Apparatus Setup for Limit Tests

For the testing of the platform's limits, the platform's orientation and position were incrementally adjusted until a point of failure. Initially large increments were made until a point of failure occurred where the platform would reach a singular position and become unstable. After a rough estimate of where the point of failure occurred was determined, smaller and smaller searches were performed in a binary search fashion until the point of instability was known to the nearest millimetre or degree.

4.2.7 Limit Tests

Measurement	Maximum (mm)	Minimum (mm)
X - No Load	125	-61
Y - No Load	95	-95
Z - No Load	210	105

Table 4.2: A table showing the position stability limits of the Stewart platform for no load

Measurement	Maximum (mm)	Minimum (mm)
X - 588g Load	117	-45
Y - 588g Load	75	-75
Z - 588g Load	205	105

Table 4.3: A table showing the position stability limits of the Stewart platform for a 588g load

Measurement	Maximum (degrees)	Minimum (degrees)
Roll - No Load	33	-33
Pitch - No Load	37	-26
Yaw - No Load	45	-45

Table 4.4: A table showing the orientation stability limits of the Stewart platform for no load

Measurement	Maximum (degrees)	Minimum (degrees)
Roll - 588g Load	18	-18
Pitch - 588g Load	36	-24
Yaw - 588g Load	34	-34

Table 4.5: A table showing the orientation stability limits of the Stewart platform for a 588g load

4.3 Analysis of Results

As described earlier, the three metrics used to judge the performance of the platform were the following:

- Accuracy
- Precision
- Robustness

Each of the metrics will be described below together with the necessary formulae and calculations used in order to obtain them. This is followed by a discussion of the results. For a comprehensive list of all the results in tabular form, please refer to Appendix C. It should be noted that the static tests were used as the test data for determining these metrics; however, dynamic test results will also be discussed.

4.3.1 Description of Metrics

The accuracy metric describes the Stewart platform's ability to move to a position or orientation which it was given as an input. The input is seen as a reference measurement and the output of the platform is then compared to that reference, with the difference between the two providing an error value. The mean of these error values could then be calculated. The mean error value was used as the measurement for accuracy and was measured in millimetres for position and degrees for orientation measurements. The equation used to calculate the error value is shown below in Eq. 4.1 where x_i is the i^{th} input value and y_i is the i^{th} measured output value in a set of N measurements.

$$e = \frac{1}{N} \sum_{i=1}^{N} |x_i - y_i| \tag{4.1}$$

The precision metric describes the Stewart platform's ability to replicate similar output values for a given input value. For a set of outputs produced from the same input value, a mean output value is determined. This allows us to compute the standard deviation of a set of measurements. This was used as the measure for precision and was measured in millimetres for position and degrees for orientation measurements. Generally, a high

standard deviation means that there is low precision and a low standard deviation means there is high precision for any given set of measurements. The equation used to calculate the standard deviation is shown below in Eq. 4.2 where y is the set of measured output values and N is the size of the set.

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(y_i - \overline{y})^2} \tag{4.2}$$

A graphical depiction of accuracy vs precision is shown below in Fig. 4.26.

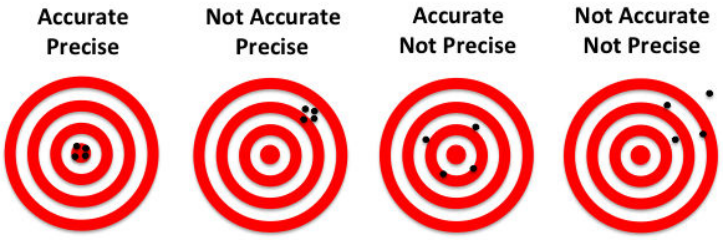

Figure 4.26: A picture illustrating the difference between accuracy and precision [7]

Robustness defines the Stewart platform's ability to replicate output values for a given input value under different stresses. In order to quantify this, the precision calculated from the loaded and unloaded tests was compared. The difference between the two precision values gave a measure of how precision changed under load. The smaller the difference between the two, the more robust the platform.

4.3.2 Discussion of Testing Results

Static Tests

The results from the static tests provided a large set of data from which different characteristics of the platform could be observed. The platform proved to be accurate on the whole, achieving an average error of 1.01° over all the orientation tests and 1.00mm over all the position tests. The platform was most accurate when centred around the home position. This is portrayed well in Fig. 4.15 where it can be seen that the accuracy of the platform decreased as the platform's origin was moved further from the home position.

The Stewart platform showed that it was capable of precise movements with a mean

standard deviation of 0.36° for all the orientation tests and $0.72mm$ for all the position tests. Similarly to the accuracy, the precision of the platform decreased as the platform's origin was moved further from the home position.

Dynamic Tests

The dynamic tests also showed that the platform is capable of tracking movement well. In all of the movement tests performed, the tracking error remained below 1.5° for all frequencies between $0Hz$ and $1Hz$. This means that the platform is capable of simulating a scaled-down ocean surface within these frequency ranges as long as the required positions and orientations are limited to the platform's workspace.

Platform Robustness

It was interesting to note that when placing a load on the platform, the accuracy increased substantially for orientation tests. The average error for the orientation tests is shown in Table 4.6. When looking at the position tests, the load had little impact on the X translation accuracy; however, the Y translation accuracy saw an increase in accuracy while the Z translation decreased in accuracy. This can be seen below in Table 4.7 which shows the average error for the position tests.

Measurement	Average Error - No Load (degrees)	Average Error - With Load (degrees)
Roll	1.00	0.20
Pitch	1.13	0.53
Yaw	0.9	0.53

Table 4.6: A table showing the accuracy of the platform for orientation tests with and without a load

Measurement	Average Error - No Load (mm)	Average Error - With Load (mm)
X	1.24	1.28
Y	1.15	0.55
Z	0.63	1.52

Table 4.7: A table showing the accuracy of the platform for position tests with and without a load

The changes in accuracy due to changes in load could possibly be attributed to the weight distribution on the legs and joints of the Stewart platform. When the weight was evenly distributed over the legs and a load was placed upon the platform, the platform was noticeably sturdier. Additionally, when the weight was not distributed evenly, certain joints were subject to larger forces acting upon them. If there was play in joints due to misalignment of bearings or bending in the legs, the effects may have been exaggerated under higher stresses causing inaccuracies to be larger. The end effector platform would flex slightly when under load - this may have also had an impact on the accuracy of the platform.

Workspace

The workspace of the built platform was smaller than the theoretical workspace when testing the limits of the platform. This was expected due to the joints and platform configuration limiting the motion. When a load was placed on the platform, the translational position and orientation workspaces were smaller. When the platform neared a singular position, the added weight often caused the platform to slip into the singular position. However, the range of motion was considerably larger than that achieved by Dan Royer's design [40] discussed in the design section. It should be noted that the custom joint design improved the range of motion when compared to designs using ball-and-socket joints.

Unfortunately, it was not possible to fully characterise the workspace of the designed platform unlike the MATLAB simulated model. The tests were time-consuming even with sections of them being automated and time was the limiting factor in not being able to perform more tests.

General Observations

Apart from the above results, several other characteristics were noticed. There were slight vibrations when moving the platform which were most likely due to stepper motors being used as the actuators for the legs. The discrete stepping motion, without any gearing or damping, resulted in the legs vibrating. This was most noticeable when all six legs were moving together such as in the dynamic tests.

The stepper motors get warm after extended periods of use. This does not affect the

motor's ability to operate; however, due to the stepper motor mounts being made out of PLA plastic which has a melting point of $60°C$, there were worries that the heat generated by the motors would cause the mounts to deform.

Chapter 5

Conclusions

5.1 End of Project Requirement Verification

After the Stewart platform had been built and tested, the design requirements were verified. All the design requirements are listed below along with the outcome of the verification procedure, stating whether the requirement was fully satisfied, partially satisfied or not satisfied. Where requirements were not fully satisfied, an explanation is provided.

General Design Requirements

RG-1: Stewart Platform:

- RG-1.1: **Fully Satisfied** - The platform should look professional and be a complete product.
- RG-1.2: **Fully Satisfied** - The platform should be a self-contained unit, not needing to be assembled and disassembled for transportation.
- RG-1.3: **Fully Satisfied** - The platform should not need any external electronics to function such as a power supply (with the exclusion of the computer which it would be plugged into).
- RG-1.4: **Fully Satisfied** - The platform should look aesthetically appealing.

Mechanical Design Requirements

RM-1: Stewart Platform:

- RM-1.1: **Fully Satisfied** - The platform should have an end effector capable of 6-DOF movement.
- RM-1.2: **Fully Satisfied** - The platform's end effector should be capable of achieving a yaw, pitch and roll of $\pm 15^{\circ}$ from the horizontal in a centred position (i.e. no translation has occurred).
- RM-1.3: **Fully Satisfied** - The platform's end effector should be capable of translation along the x, y and z-axes.
- RM-1.4: **Fully Satisfied** - The platform's end effector should have a maximum error of 5° for given yaw, pitch and roll inputs.
- RM-1.5: **Fully Satisfied** - The platform's end effector should allow for tests to be repeatable within the accuracy stated above.
- RM-1.6: **Fully Satisfied** - The platform should have a built-in mechanism which can be used to calibrate the end effector.
- RM-1.7: **Fully Satisfied** - The platform's legs should be driven by electrical actuators.
- RM-1.8: **Fully Satisfied** - The platform's legs should be designed in a way which attempts to minimise limitations on the end effector's movements.

RM-2: Manufacturing Materials:

- RM-2.1: **Fully Satisfied** - The platform should make use of 3D printed, laser cut and other commonly available parts in order to balance the design's robustness while ensuring it is easily re-creatable and affordable.
- RM-2.2: **Fully Satisfied** - Large flat components such as the frame should be made from commonly accessible materials such as hardboard, wood or perspex.
- RM-2.3: **Fully Satisfied** - Smaller components which are structurally complex should be 3D printed out of PLA plastic.
- RM-2.4: **Partially Satisfied** - The end effector and leg assemblies should be designed in a way which minimises their weight in order to allow for larger loads to be placed on the platform.
  Note: The majority of weight in the leg assemblies can be attributed to the 608zz bearings which were used. A total of 60 bearings were used, each weighing approximately 10g. If alternative bearings are used, the weight may be significantly reduced; however, this may increase the cost of building the platform.

Electrical Design Requirements

RE-1: Electrical Safety:

- RE-1.1: **Fully Satisfied** - The entire system should be electrically safe.
- RE-1.2: **Fully Satisfied** - All electrical connections should be insulated to ensure no bare wires are exposed to the user.
- RE-1.3: **Fully Satisfied** - The system should have an external on/off switch in order to be able to turn the unit off at any time in case of malfunction.
- RE-1.4: **Fully Satisfied** - The system should use an IEC60320 C13/C14 coupler as the power inlet for the device (these are commonly referred to as 'kettle plugs').
- RE-1.5: **Fully Satisfied** - The system should have a cooling system implemented into the design to cool electronic components such as motor drivers, power amplifiers or linear voltage regulators.
- RE-1.6: **Fully Satisfied** - All power management and cabling should be contained within the system and be rated correctly for its intended use.

RE-2: Actuators:

- RE-2.1: **Partially Satisfied** - Limits should be integrated into the software to prevent dangerous movements.
  Note: It was decided during the project that it would be easier to test the platform by removing the limits integrated into the microcontroller code. This allowed the user complete freedom to move the platform's actuators as they wished and meant that different platform configurations could be tested with the same microcontroller code; however, limits can easily be integrated in the MATLAB software when needed.
- RE-2.2: **Partially Satisfied** - Actuators should be prevented from moving in ways which may cause harm to the user and/or platform.
  Note: As noted above, the decision to limit user's input was decided against and instead, limits should be implemented in the user's PC software.

Microcontroller Code Requirements

RuC-1: Communications Protocol:

- RuC-1.1: **Fully Satisfied** - The microcontroller code should integrate a protocol which provides an interface to a PC via a USB cable.

RuC-1.2: **Fully Satisfied** - The protocol should allow for the platform's movements to be controlled.

RuC-1.3: **Fully Satisfied** - The protocol should allow the platforms movements to be halted.

RuC-1.4: **Fully Satisfied** - The protocol should allow the platform to be calibrated.

RuC-2: Control:

RuC-2.1: **Fully Satisfied** - The microcontroller should abstract all actuator control away from the user.

RuC-2.2: **Fully Satisfied** - A user should be able to input actuator position commands and the platform should perform all the necessary hardware actions to move the actuators to these positions.

RuC-3: Testing:

RuC-3.1: **Fully Satisfied** - Testing should be performed to ensure that the microcontroller code is stable and operates correctly.

RuC-3.2: **Fully Satisfied** - Tests should include software tests as well as hardware tests using lab equipment such as oscilloscopes.

PC Requirements

RPC-1: Computer Software:

RPC-1.1: **Fully Satisfied** - The computer software should show a visual representation of the platform through means of graphs and/or a 3D model of the platform.

RPC-1.2: **Fully Satisfied** - The computer software should be simple to use and be accompanied by documentation.

RPC-1.3: **Fully Satisfied** - The computer software should be cross-platform and run on at least Windows 10, MacOS and Ubuntu Linux to ensure its usability is maximised.

RPC-2: Communications:

RPC-2.1: **Partially Satisfied** - The PC should have at least one USB port to allow for communications with the platform.

Note: - The Stewart platform makes use of a USB cable which sticks out the back of the platform instead of a USB port. This was due to not having a micro USB extension cable; however, there is a hole on the back of the enclosure which would allow for the port to be added.

5.2 Conclusions from Testing

This design of the Stewart platform has shown that it is capable of accurately simulating motions in the $0.03Hz$ to $1Hz$ frequency range. This means that it is suitable for simulating ocean waves based off the models described in the literature review. As well as this, the Stewart platform was robust when tested under loads of $588g$ in the static tests and $198g$ in the dynamic tests.

The Stewart platform had a high precision, meaning that it would make a suitable HIL simulator. Its robustness would allow for simulations to be run multiple times without the platform's output changing between each simulation.

The workspace of the platform was smaller than that of the theoretical workspace. More advanced techniques would need to be used in modelling the workspace which could account for the collisions between legs and the base and limitations of joints if one wanted to model the workspace more accurately.Performing a motion study in SolidWorks is one possible way of determining this.

The use of stepper motors with the current stepper motor drivers leads to small vibrations in the motion of the platform. The motors also heat up which means that to avoid the PLA plastic motor mounts melting, the device should not be used for more than an hour at a time without a cool-down period. This could be easily remedied by 3D printing the motor mount in a material such as PETG or ABS plastic.

5.3 General Conclusions

The project was ultimately successful. This project has shown that it is possible to develop an accurate and robust Stewart platform for a relatively low cost when compared with commercially available products. While this was only a prototype, it may still have use as a small HIL simulator for devices with on-board gyroscopes and accelerometers.

Chapter 6

Recommendations

6.1 Materials

Several materials used in the design of the Stewart platform may be structurally inadequate. The end effector platform, which is made from $3mm$ thick hardboard has a slight flex in it when placed under load. Ideally, the platform should be completely rigid ensuring its shape is maintained. It would be recommended that it be machined out of aluminium or reinforced somehow. This platform needs to be both light and rigid.

The motor mounts which are made from PLA (with a melting point of $60°C$) may be a risk as the heat from the stepper motors may cause them to deform. These should be printed from a material which can withstand the heat of the motors such as PETG or ABS plastic.

6.2 Bearings

The 608zz bearings added significant weight to the legs of the platform. It would be recommended that these joints be redesigned using much lighter bearings to reduce the load on the actuators.

6.3 Calibration Mechanism

Ideally, the Stewart platform should have a built-in calibration mechanism. This problem could be approached by including limit switches which are triggered when the platform's actuators are in a zero position or using optical encoders to feed the motor shaft's angular position back to the microcontroller.

6.4 Bolts

It is highly recommended that flat-head bolts are not used in future designs. This may seem like a small issue; however, when assembling the platform, tightening screws was the most time-consuming part of the assembly. Flat-head bolts do not hold a screwdriver in place while tightening, often leading to the screwdriver slipping off the bolt and scratching parts of the project. It is recommended that hex-head bolts be used.

6.5 Dynamics Modelling

This project focused on the kinematic model of the Stewart platform and did not bring the dynamics of the platform into consideration. When a load is placed on the platform and it is subject to fast accelerations, the dynamics should be taken into account to maintain the platform's accuracy.

6.6 Control

While this project used open loop control, it would be interesting to see if improvements to the platform's accuracy and precision could be achieved through the use of closed loop control. This would require a feedback mechanism. There are two approaches that might work in this regard. Firstly, optical encoders could be mounted to the actuators to determine their angular position. In addition to this, a gyroscope could be mounted underneath the end effector and used to send orientation data about the platform back to the controller. It may also be possible to use both feedback mechanisms in conjunction with one another to fine-tune the platform.

Bibliography

[1] "The Wave Glider | How It Works," 2017. Available at https://www.liquid-robotics.com/wave-glider/how-it-works/.

[2] M. Folley, *Handbook of ocean wave energy*. Cham: Springer International Publishing, 2016.

[3] F. Szufnarowski, "Stewart platform with fixed rotary actuators: a low cost design study," p. 11, 2012.

[4] "Linear Actuators," 2016. Available at https://www.design-engineering.com/products/linear-actuators-4/.

[5] W. U. M. Group, "Mathematics of the Stewart Platform," 2013.

[6] "Pololu - DRV8825 Stepper Motor Driver Carrier, High Current," 2014. Available at https://www.pololu.com/product/2133.

[7] "QRT PCR Data Analysis | Precision and Accuracy." Available at https://www.dnasoftware.com/our-products/copycount/precision-and-accuracy/.

[8] H. Mitsuyasu, "A Historical Note on the Study of Ocean Surface Waves," *Journal of Oceanography*, vol. 58, pp. 109–120, Feb. 2002.

[9] H. Sverdrup and W. Munk, "Wind sea and swell: Theory of relations for forecasting, Publ. 601, US Navy Hydrogr," *Off., Washington, DC*, 1947.

[10] H. Jeffreys, "On the Formation of Water Waves by Wind," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 107, pp. 189–206, Feb. 1925.

[11] L. H. Holthuijsen, "Waves in Oceanic and Coastal Waters," p. 405, 2007.

[12] W. H. Munk, "Origin and Generation of Waves," *Coastal Engineering Proceedings*, vol. 1, no. 1, p. 1, 1950.

[13] C. L. Bretschneider, "Generation of waves by wind. State of the art," tech. rep., National Engineering Science Co Washington DC, 1965.

[14] L. Cavaleri, J.-H. Alves, F. Ardhuin, A. Babanin, M. Banner, K. Belibassakis, M. Benoit, M. Donelan, J. Groeneweg, T. Herbers, P. Hwang, P. Janssen, T. Janssen, I. Lavrenov, R. Magne, J. Monbaliu, M. Onorato, V. Polnikov, D. Resio, W. Rogers, A. Sheremet, J. McKee Smith, H. Tolman, G. van Vledder, J. Wolf, and I. Young, "Wave modelling – The state of the art," *Progress in Oceanography*, vol. 75, pp. 603–674, Dec. 2007.

[15] W. J. Pierson and L. Moskowitz, "A proposed spectral form for fully developed wind seas based on the similarity theory of S. A. Kitaigorodskii," *Journal of Geophysical Research*, vol. 69, pp. 5181–5190, Dec. 1964.

[16] K. Hasselmann, T. Barnett, E. Bouws, H. Carlson, D. Cartwright, K. Enke, J. Ewing, H. Gienapp, D. Hasselmann, P. Kruseman, and others, "Measurements of wind-wave growth and swell decay during the Joint North Sea Wave Project (JONSWAP)," *Ergänzungsheft 8-12*, 1973.

[17] M. K. Ochi and E. N. Hubble, "Six-Parameter Wave Spectra," in *Coastal Engineering 1976*, (Honolulu, Hawaii, United States), pp. 301–328, American Society of Civil Engineers, Nov. 1977.

[18] P. Brodtkorb, P. Johannesson, G. Lindgren, I. Rychlik, J. Rydén, and E. Sjö, "WAFO - a Matlab Toolbox for the Analysis of Random Waves and Loads," in *Proc. 10'th Int. Offshore and Polar Eng. Conf., ISOPE, Seattle, USA*, vol. 3, pp. 343–350, 2000.

[19] K. Torsethaugen and S. Haver, "Simplified double peak spectral model for ocean waves," p. 9, 2004.

[20] C. Stansberg, G. Contento, S. W. Hong, M. Irani, S. Ishida, R. Mercier, Y. Wang, J. Wolfram, J. Chaplin, and D. Kriebel, "The specialist committee on waves final report and recommendations to the 23rd ITTC," *Proceedings of the 23rd ITTC*, vol. 2, pp. 505–551, 2002.

[21] D. Stewart, "A Platform with Six Degrees of Freedom," *Proceedings of the Institution of Mechanical Engineers*, vol. 180, pp. 371–386, June 1965.

[22] V. E. Gough, "Contribution to discussion to papers on research in automobile stability and control and in tyre performance, by Cornell staff," *Proc. Auto. Din. Insrn mech. Engrs*, vol. 10, p. 392, Jan. 1956.

[23] C. F. Earl and J. Rooney, "Some Kinematic Structures for Robot Manipulator Designs," *Journal of Mechanisms Transmissions and Automation in Design*, vol. 105, no. 1, p. 15, 1983.

[24] K. H. Hunt, "Structural Kinematics of In-Parallel-Actuated Robot-Arms," *Journal of Mechanisms Transmissions and Automation in Design*, vol. 105, no. 4, p. 705, 1983.

[25] E. Fichter, "A Stewart Platform- Based Manipulator: General Theory and Practical Construction," *The International Journal of Robotics Research*, vol. 5, pp. 157–182, June 1986.

[26] J.-P. Merlet, *Parallel manipulators. Part I: Theory design, kinematics, dynamics and control.* PhD thesis, INRIA, 1987.

[27] B. Dasgupta and T. Mruthyunjaya, "A Newton-Euler formulation for the inverse dynamics of the Stewart platform manipulator," *Mechanism and Machine Theory*, vol. 33, pp. 1135–1152, Nov. 1998.

[28] J.-P. Merlet, "Parallel manipulators: state of the art and perspectives," *Advanced Robotics*, vol. 8, pp. 589–596, Jan. 1993.

[29] B. Dasgupta and T. Mruthyunjaya, "The Stewart platform manipulator: a review," *Mechanism and Machine Theory*, vol. 35, pp. 15–40, Jan. 2000.

[30] Z. Bingul and O. Karahan, "Dynamic Modeling and Simulation of Stewart Platform," *Control and Optimization*, p. 25, 2012.

[31] J.-P. Merlet, "Singular Configurations of Parallel Manipulators and Grassmann Geometry," *The International Journal of Robotics Research*, vol. 8, pp. 45–56, Oct. 1989.

[32] G. Lebret, K. Liu, and F. L. Lewis, "Dynamic analysis and control of a stewart platform manipulator," *Journal of Robotic Systems*, vol. 10, pp. 629–655, July 1993.

[33] K. J. Waldron and K. H. Hunt, "Series-Parallel Dualities in Actively Coordinated Mechanisms," *The International Journal of Robotics Research*, vol. 10, pp. 473–480, Oct. 1991.

[34] S.-H. Lee, J.-B. Song, W.-C. Choi, and D. Hong, "Position control of a Stewart platform using inverse dynamics control with approximate dynamics," *Mechatronics*, vol. 13, pp. 605–619, July 2003.

[35] S. Bhattacharya, H. Hatwal, and A. Ghosh, "Comparison of an exact and an approximate method of singularity avoidance in platform type parallel manipulators," *Mechanism and Machine Theory*, vol. 33, pp. 965–974, Oct. 1998.

[36] D. C. H. Yang and T. W. Lee, "Feasibility Study of a Platform Type of Robotic Manipulators from a Kinematic Viewpoint," *Journal of Mechanisms Transmissions and Automation in Design*, vol. 106, no. 2, p. 191, 1984.

[37] O. Masory and Jian Wang, "Workspace evaluation of Stewart platforms," *Advanced Robotics*, vol. 9, pp. 443–461, Jan. 1994.

[38] O. Bohigas, L. Ros, and M. Manubens, "A Unified Method for Computing Position and Orientation Workspaces of General Stewart Platforms," in *Volume 6: 35th Mechanisms and Robotics Conference, Parts A and B*, (Washington, DC, USA), pp. 959–968, ASME, 2011.

[39] A. Rastegarpanah, M. Saadat, and H. Rakhodaei, "Analysis and simulation of various Stewart Platform configurations for lower limb rehabilitation," p. 8, 2013.

[40] D. Royer, "Rotary Stewart Platform," Sept. 2013. Available at https://www.thingiverse.com/thing:152219.

[41] "Texas Instruments DRV8825 Stepper Motor Controller IC Datasheet," 2014. Available at http://www.ti.com/lit/ds/symlink/drv8825.pdf.

[42] "Texas Instruments LM7805 Linear Voltage Regulator Datasheet," 2003. Available at http://www.ti.com/lit/ds/symlink/lm340.pdf.

[43] J. E. Bresenham, "Algorithm for computer control of a digital plotter," *IBM Systems Journal*, vol. 4, no. 1, pp. 25–30, 1965.

Appendix A

Additional Mathematics and Algorithms

A.1 Mathematics of Bresenham's Algorithm

The maths behind the Bresenham line algorithm is as follows. First, it must be understood than any straight line (in which two points are contained) can be represented by Eq. A.1.

$$y = \frac{\Delta x}{\Delta y}x + c \tag{A.1}$$

By shifting all terms to one side of the equation, Eq. A.1 can be written as Eq. A.2.

$$(\Delta y)x - (\Delta x)y + (\Delta x)c = 0 \tag{A.2}$$

Let a function $f(x, y)$ be defined as per Eq. A.3.

$$f(x, y) = (\Delta y)x - (\Delta x)y + (\Delta x)c \tag{A.3}$$

From Eq. A.3, one should observe for points x and y that lie on the line, $f(x, y) = 0$. If the point (x, y) exists above the line, $f(x, y) < 0$ and if (x, y) exists below the line,

$f(x, y) > 0$.

Now, suppose a point (x_1, y_1) (where x_1 and y_1 are integers) is situated on a line and $0 \leq \frac{\Delta y}{\Delta x} \leq 1$. It is known that the next integer points on the line can only be $(x_1, y_1 + 1)$ or $(x_1 + 1, y_1 + 1)$.

If the value of $f(x_1 + \frac{1}{2}, y_1 + 1)$ is evaluated (which is the next integer point along the y-axis) and $f(x_1 + \frac{1}{2}, y_1 + 1) < 0$, the point which should be used to plot the line most closely will be $(x_1, y_1 + 1)$. If $f(x_1 + \frac{1}{2}, y_1 + 1) > 0$, the point $(x_1 + 1, y_1 + 1)$ should be chosen.

This set of equations is only valid for plotting lines where $0 \leq \frac{\Delta y}{\Delta x} \leq 1$. Bresenham [43] describes his algorithm with minor modifications for eight octants. He also describes how integer arithmetic can be used which is more efficient than using floating-point or fixed-point operations. The algorithm used in the microprocessor code uses a two octant, integer variant of the algorithm.

Appendix B

The Build Guide

B.1 Mechanical Build

In building the platform, the following parts in Fig. B.1 and Fig. B.2 will need to be manufactured through either 3D printing or laser cutting. All parts needing to be purchased can be found in the bill of materials at the end of this build guide. The SolidWorks part files can all be found at this address: `https://github.com/LiamClarkZA/Final-Year-Project/tree/master/Solidworks%20Files`.

Along with this, additional parts will need to be purchased such as the stepper motors, stepper motor drivers and bearings. Once again, refer to the bill of materials for a comprehensive list of the required items.

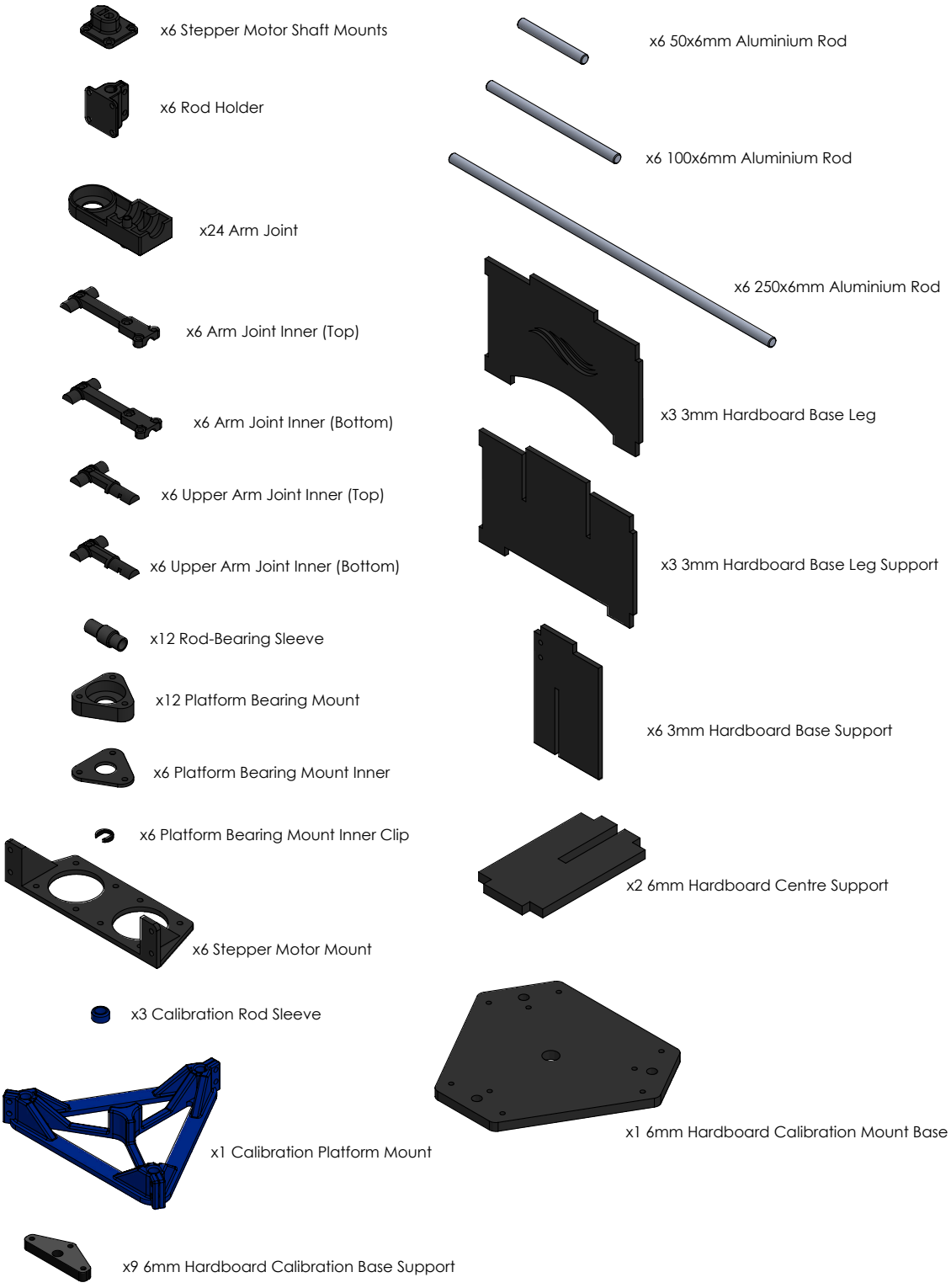

Figure B.1: Parts list 1

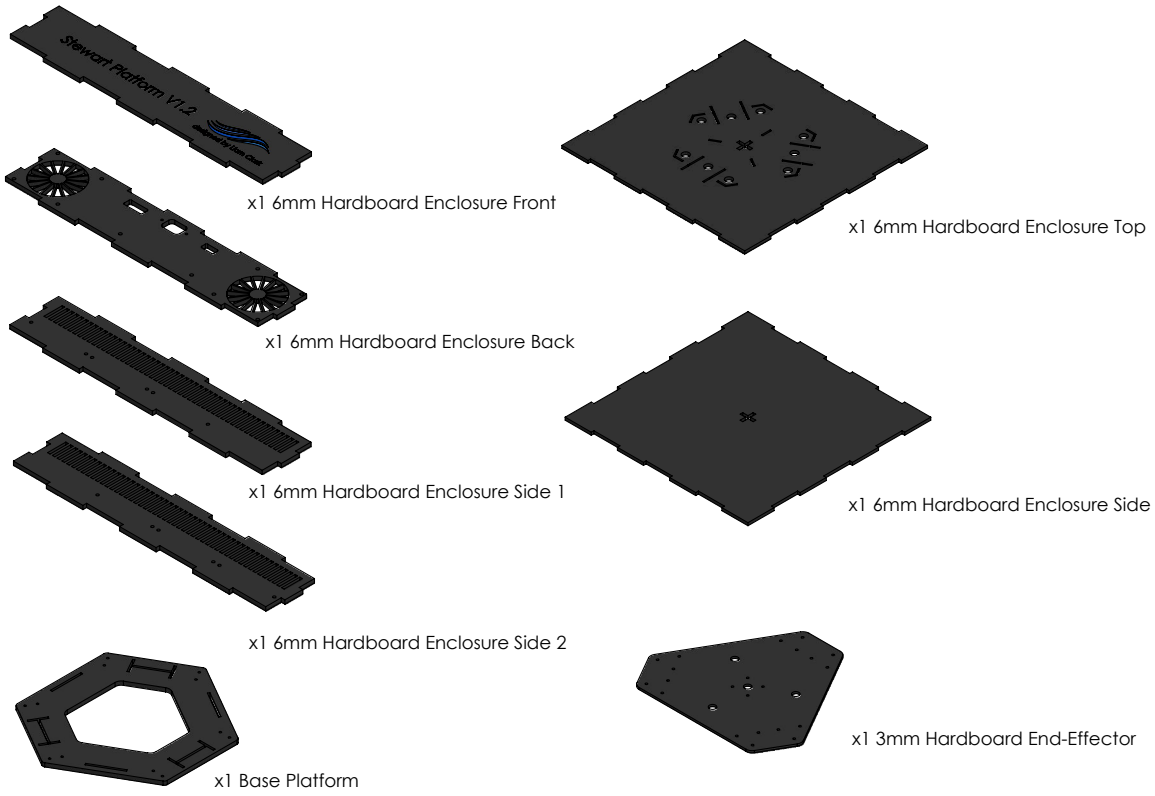

Figure B.2: Parts list 2

B.1.1 Legs

The first parts which should be assembled are the platform legs. To start, assemble six lower legs using the component diagram shown in B.3. Ensure that all the bearings are pushed into place firmly before screwing the joint together.

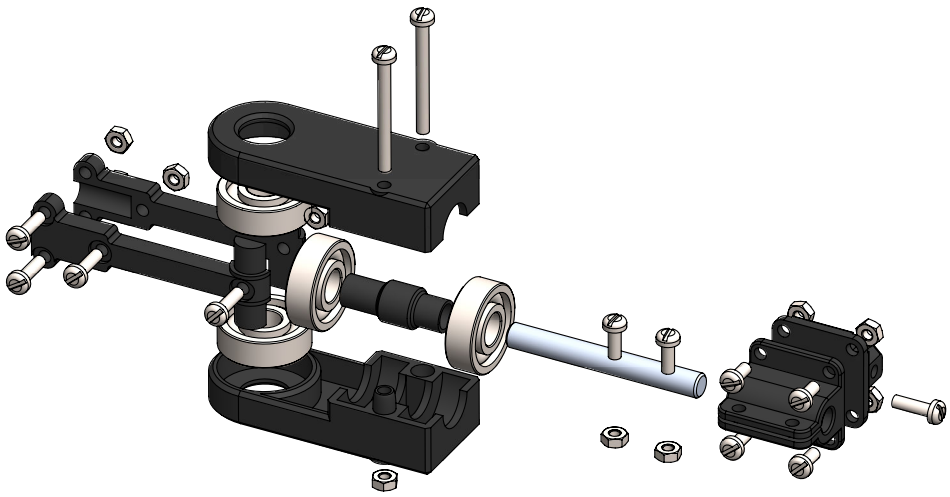

Figure B.3: Lower leg exploded view

The next parts to assemble are the upper legs as shown in Fig. B.4. Do not forget to add the clip between the bearings in the platform mount - this is used to separate the bearings.

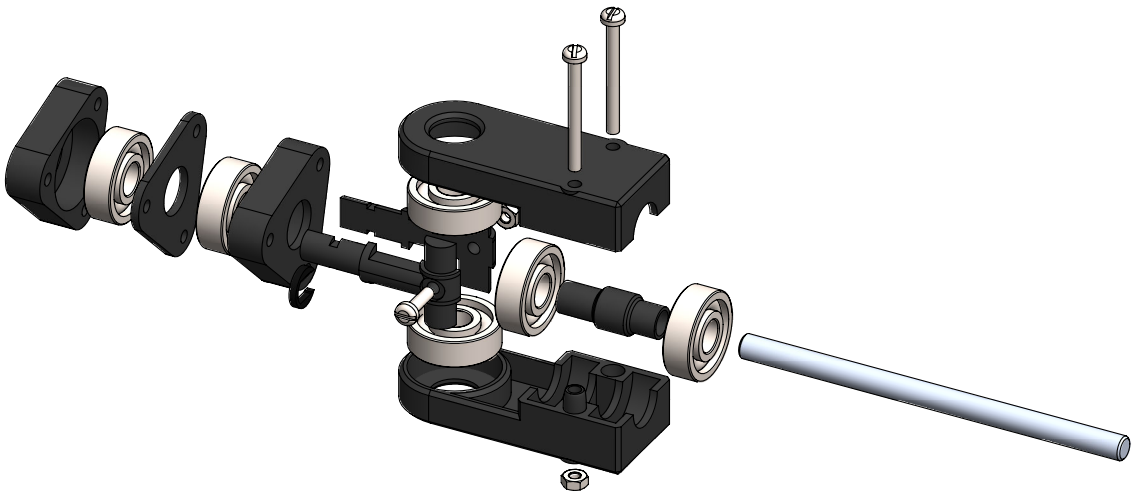

Figure B.4: Upper leg exploded view

When the upper and lower legs are assembled, join them together by sliding the aluminium rod from the upper leg assembly into the rod holder on the non-attached side of the joint. When complete, the leg should look like Fig. B.5. Make sure that all the bolts are tightly fastened as there should be no room for movement from any parts aside from the bearings.

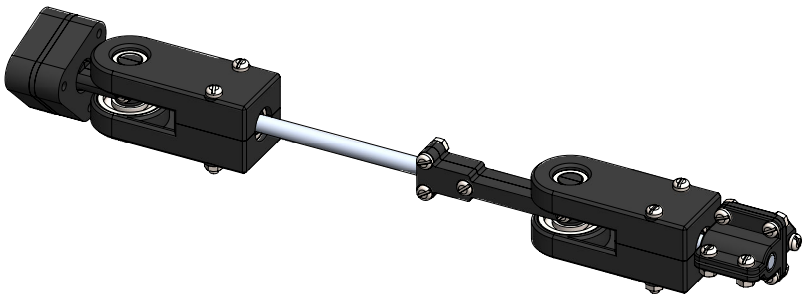

Figure B.5: Full leg assembly

Next, the motors should be mounted to the motor mount. This is shown in Fig. B.6.

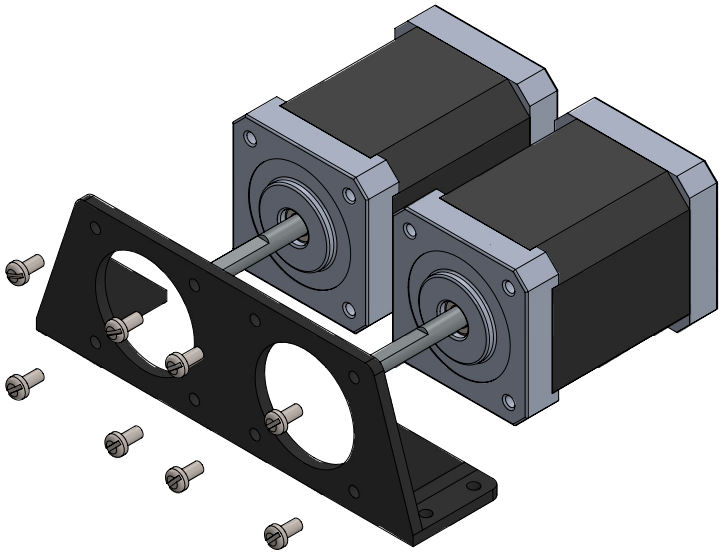

Figure B.6: Motor mount assembly exploded view

Once the motors are mounted, a leg can be attached to each motor. Once attached, the assembly should look like Fig. B.7.

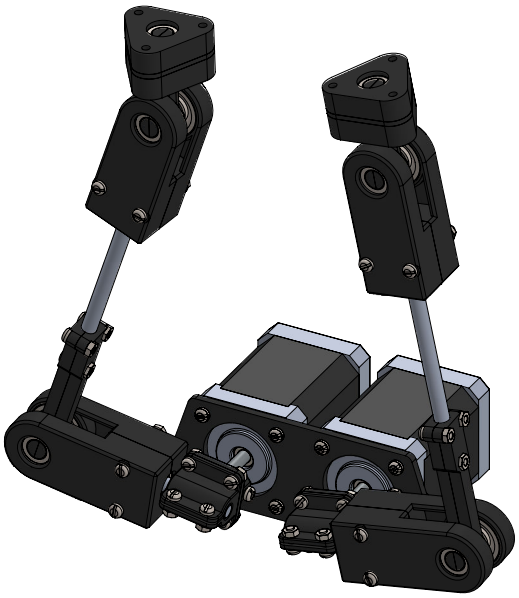

Figure B.7: Full motor and legs assembly

B.1.2 Platform and Base

After the legs have been assembled, the base platform and base enclosure need to be assembled. This will require super glue or wood glue to fix the sides of the enclosure to the base of the enclosure. The assembly is straightforward with all the parts fitting together like jigsaw pieces. The exploded view of the base enclosure and base platform assembly is shown in Fig. B.8.

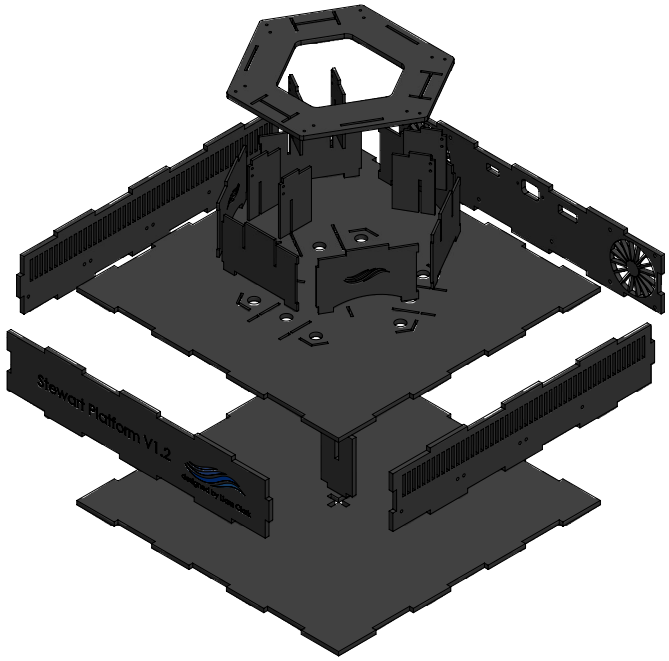

Figure B.8: Base platform and base enclosure exploded assembly

Do not glue the top of the enclosure onto the sides as this is intended to be removable. Once assembled, the base enclosure and platform should look like Fig. B.9.

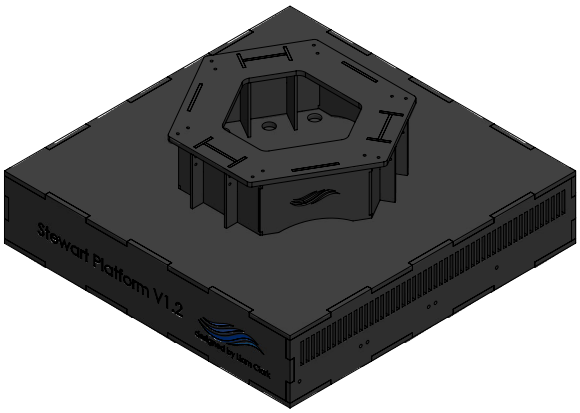

Figure B.9: Base platform and base enclosure assembly

B.1.3 Calibration Rig

In order to calibrate the platform, the platform calibration mount needs to be assembled as well as the base calibration block. The exploded view of the platform calibration mount is shown in Fig. B.10 and the exploded view of the base calibration block is shown in Fig. B.11.

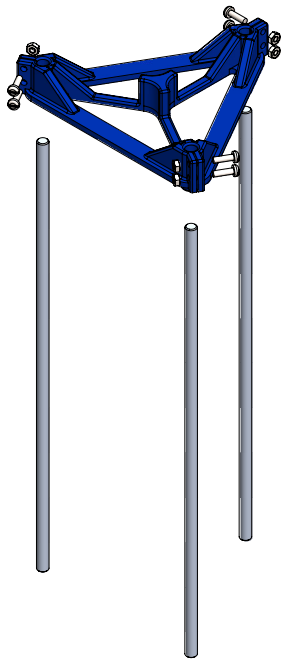

Figure B.10: Platform calibration mount exploded view

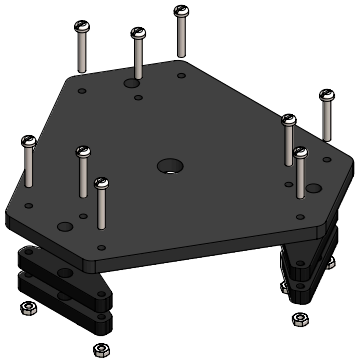

Figure B.11: Base calibration block exploded view

The calibration block assembly can then be slid into the centre of the base of the platform.

B.1.4 Full Assembly

Once the legs, platform base, base enclosure and calibration rig are assembled, they can all be put together. The end-effector will need to be attached to the legs and this is easiest once the motors are mounted to the base platform. The exploded view of the full assembly is shown below in Fig. B.12.

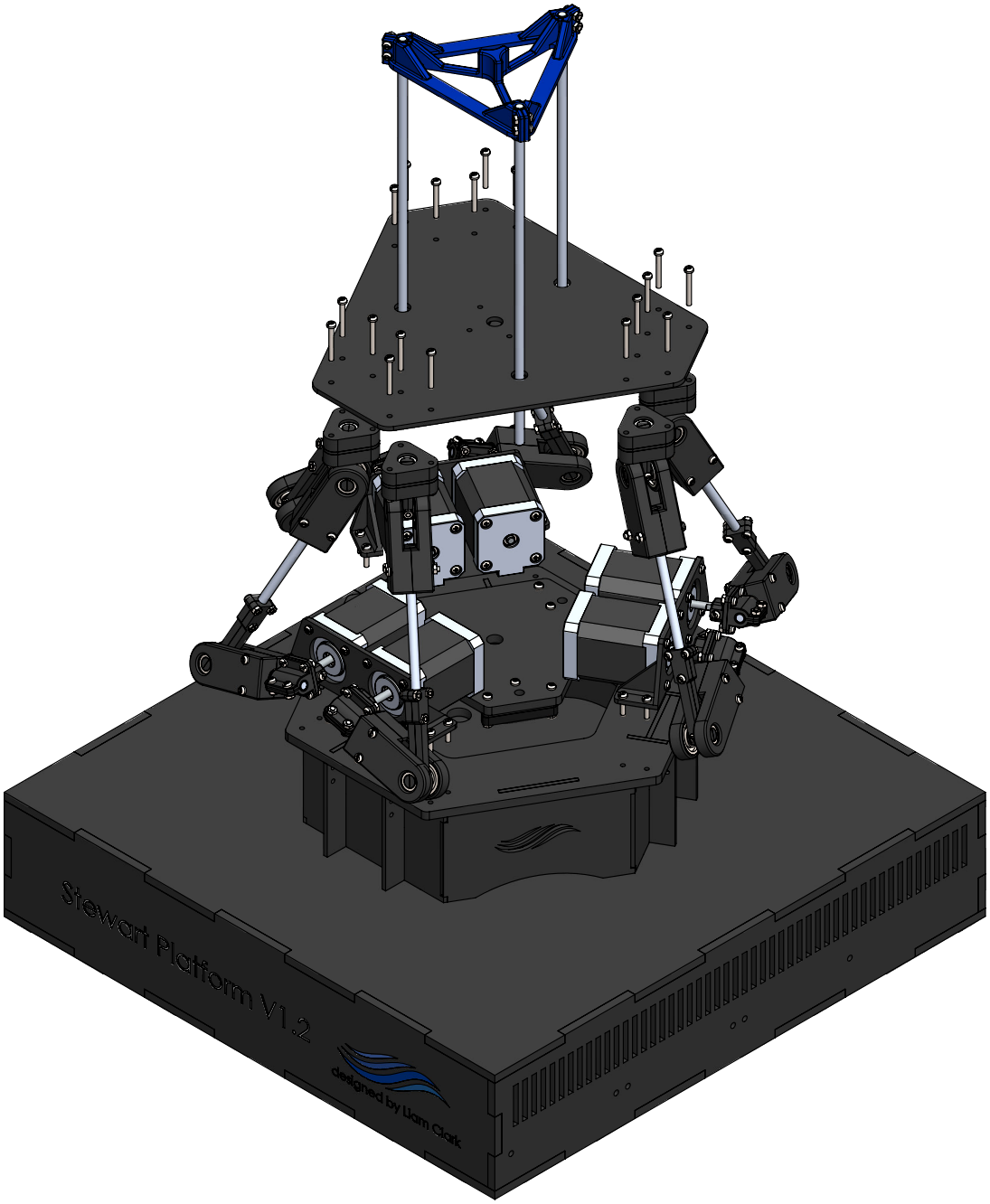

Figure B.12: Full assembly exploded view

Finally, when everything is in place, the full assembly should look like Fig. B.13.

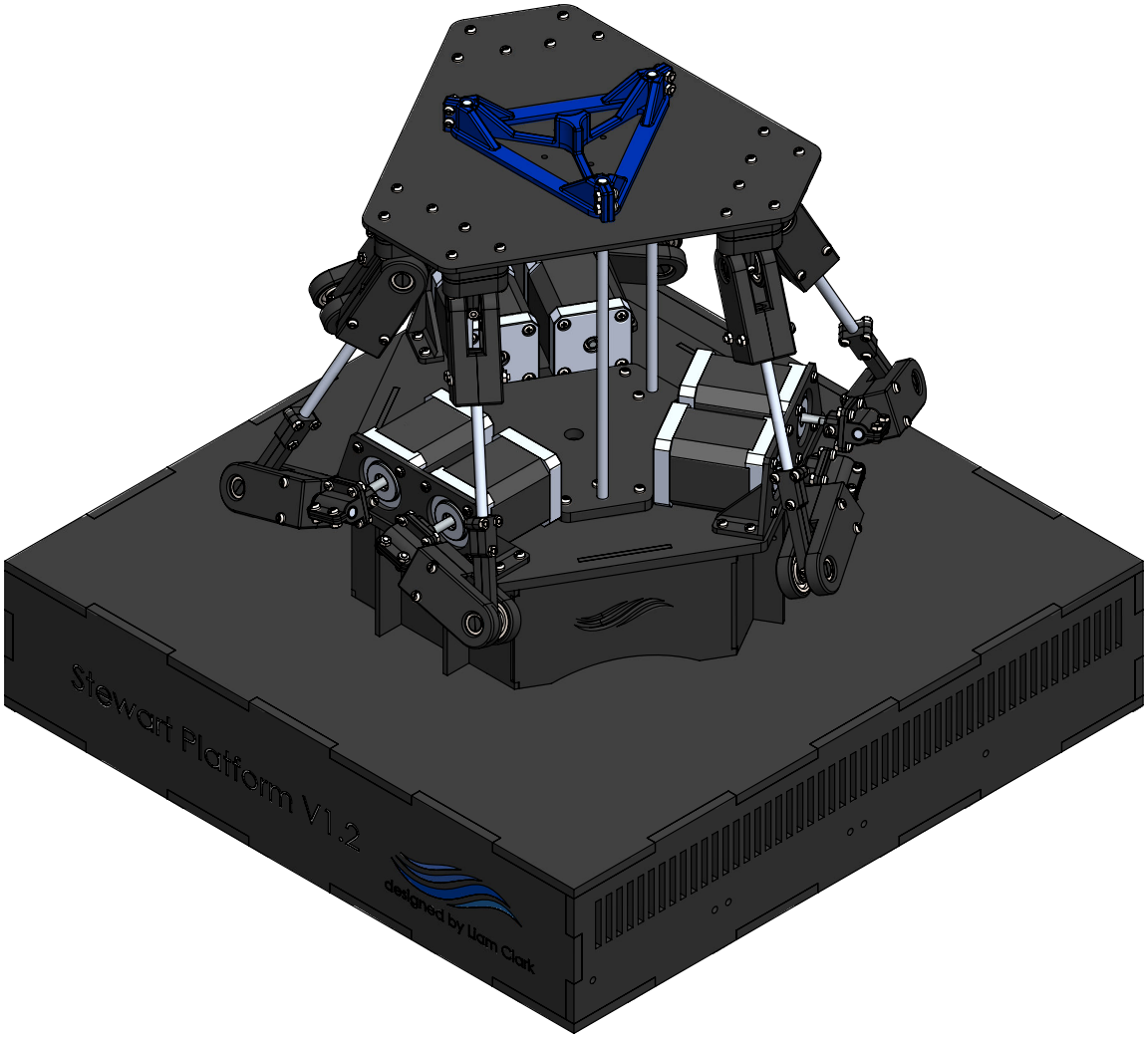

Figure B.13: Full Stewart platform assembly

B.2 Electronics and Software Setup

B.2.1 Wiring

The first step in setting up the electronics is to wire up each motor driver. While this can be achieved by using a breadboard, it is recommended that veroboard modules for each stepper motor driver are created using female pin headers so that the stepper motor driver is removable. The veroboard should be wired as per Fig. B.14. A four pin molex connector should be used for the stepper motor phases. Two pin molex connectors should be used for the 12V power supply connection and as connectors for the step and direction pins which are connected to the microcontroller. It is also worth checking the wiring on the veroboard modules, especially the output of the voltage regulator, to avoid causing any damage to the stepper motor driver when it is plugged in.

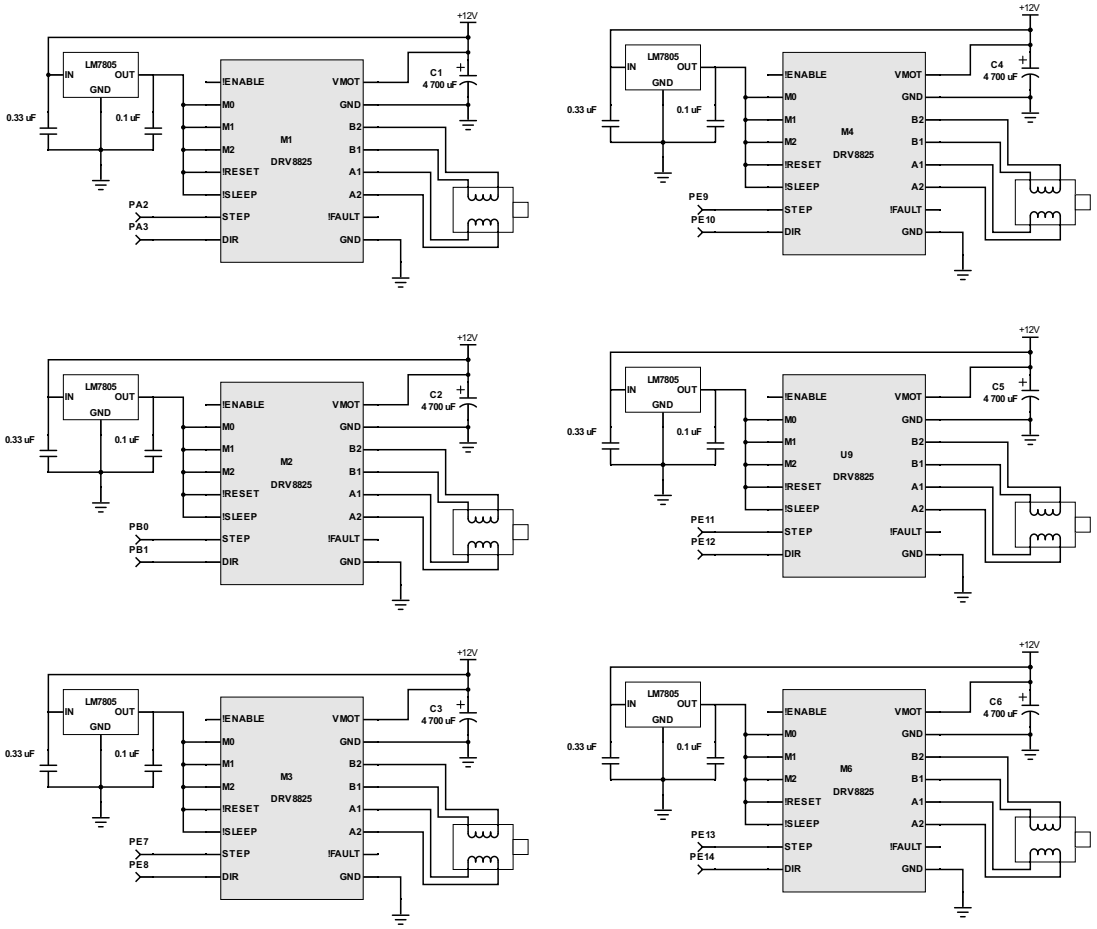

Figure B.14: A schematic diagram showing the connections of the 6 DRV8825 ICs

Before continuing, it is worth installing the power supply and fans into the Stewart plaform's base enclosure. This will ensure that connecting wires can easily be cut to the correct lengths. The stepper motor drivers should be connected to the microcontroller and to the 12V power supply. The microcontroller should be grounded by connecting the GND pin to the 0V output on the power supply.

Following this, the fans can be connected to the 12V power supply. A LM7805 voltage regulator can be soldered to the fan's 12V, 5V and 0V wires without the need for a veroboard; however, heat-shrink should be used to ensure no wires are exposed. Bypass capacitors should also be soldered to the inputs and outputs of the voltage regulators as per the LM7805 datasheet [42].

The power supply can now be wired up to the on/off switch and kettle plug socket. Before turning on the power supply, ensure all the wires are connected correctly and there are no bare wires which could cause a short during operation. The platform can now be powered on.

B.2.2 Microcontroller Setup

To set up the microcontroller, a micro USB cable and mini USB cable are required in order to connect the STM32F4 Discovery board to the computer. In addition to this, the Atollic TrueStudio IDE will need to be downloaded as well as the following project files from GitHub: `https://github.com/LiamClarkZA/Final-Year-Project/tree/master/Microcontroller%20Code`.

Once Atollic TrueStudio is installed, open the Atollic project file that was downloaded from GitHub. The microcontroller should be connected to the computer via the two USB cables now. In Atollic TrueStudio, the project can be compiled by clicking the build icon and then uploaded to the microcontroller by clicking the run icon. The microcontroller code will have now been flashed to the microcontroller. Atollic TrueStudio may now be closed. Once closed, reset the microcontroller to ensure it is no longer being blocked by any breakpoints from the debugger. The microcontroller should now appear as a serial communication device on the computer.

B.2.3 MATLAB Setup

There are only a few steps required for setting up the platform in MATLAB. Firstly, the `StewartPlatform.m` file will need to be downloaded from GitHub: `https://github.com/LiamClarkZA/Final-Year-Project/tree/master/Matlab%20Code`. At the same time, it is recommended that the other MATLAB files are downloaded as examples or tools to help with any debugging. The `StewartPlatform.m` file will either need to be placed in the current working directory or can alternatively be added to MATLAB's file search path.

After this, the serial port which the Stewart platform is connected to needs to be determined. This is necessary for connecting to the platform as it is used as an argument to the `connect(<port>)` function in the `StewartPlatform` class. See the `Tester.m` file for examples.

Lastly, when instantiating a `StewartPlatform` object, ensure that the correct platform configuration parameters are entered for the Stewart platform.

The platform will now be ready to be used for simulations. Be sure to take a look at the `ReportTests.m` file for a more advanced example of how to create simulations. This can also just be used as a template requiring only the `update(<time>)` function to be modified. This allows anyone to simply define the platform's roll, pitch, yaw, x, y and z translations as functions of time while everything else is taken care of in the background.

B.3 Bill of Materials

Item	Cost	Quantity	Total Cost	Supplier
Stepper Motors (NEMA-17, 60mm length)	R298.20	6	R1789.20	UCT
Power Supply (350W, 12V)	R799.48	1	R799.48	ACDC Dynamics
Microcontroller (STM32F4 Discovery)	R379.29	1	R379.29	UCT
Fans (12V, 80mm x 80mm, 2500RPM)	R32.00	2	R64.00	UCT
Stepper Motor Drivers (Polulu DRV-8825)	R38.16	6	R76.32	Banggood
Bearings (608zz)	R10.87	60	R652.20	Silkuni
3D Printed Parts	R710.00	$710g$	R710.00	UCT
Hardboard (3mm x 1220mm x 620mm)	R96.00	1	R96.00	Mica
Hardboard (6mm x 1220mm x 620mm)	R156.00	2	R312.00	Mica
Matte Black Spray Paint	R128.00	1	R128.00	Mica
Voltage Regulators (LM7805)	R3.64	7	R25.48	UCT
Capacitors (4700uF, electrolytic)	R2.31	6	R13.86	UCT
Capacitors (0.33uF, ceramic)	R0.12	7	R0.84	UCT
Capacitors (0.1uF, ceramic)	R0.19	7	R1.33	UCT
Molex Connector (2 pin, male)	R0.56	18	R10.08	UCT
Molex Connector (2 pin, female)	R0.56	12	R6.72	UCT
Molex Connector (4 pin, male)	R0.56	6	R3.36	UCT
Molex Connector (4 pin, female)	R0.56	6	R3.36	UCT
Pin Header (1 x 8 pins, female)	R1.17	12	R14.04	UCT
Pin Header (2 x 25 pins, female)	R2.07	2	R4.14	UCT
Veroboard (100mm x 300mm sheet)	R60.00	2	R120.00	UCT
Kettle Plug Port	R14.50	1	R14.50	Mica
On/Off Switch (230V, 10A)	R12.50	1	R12.50	Mica
Crimp Spade Connector	R1.50	5	R7.50	Mica
Terminal Spade Connector	R1.50	6	R9.00	Mica
M3 Bolts (32mm)	R0.40	24	R9.60	Bolt It
M3 Bolts (25mm)	R0.40	18	R7.20	Bolt It
M3 Bolts (16mm)	R0.40	12	R4.80	Bolt It
M3 Bolts (10mm)	R0.40	36	R14.40	Bolt It
M3 Bolts (8mm)	R0.40	36	R14.40	Bolt It
M3 Bolts (6mm)	R0.40	24	R9.60	Bolt It
M4 Bolts (8mm)	R0.50	2	R1.00	Bolt It
M4.5 Screw (16mm)	R0.50	8	R4.00	Bolt It
M3 Hex Nuts	R0.20	126	R25.2	Bolt It
Total Cost			R4964.11	

Appendix C

Result Tables

C.1 Accuracy

C.1.1 Roll

Roll Input (degrees)	Average Error (degrees)
0	0.00
10	0.00
-10	0.33
20	0.67
-20	0.67
30	2.33
-30	3.00

Table C.1: A table showing the unloaded roll test results for accuracy

Roll Input (degrees)	Average Error (degrees)
0	0.00
10	0.00
-10	0.00
20	0.00
-20	1.00

Table C.2: A table showing the loaded roll test results for accuracy

C.1.2 Pitch

Pitch Input (degrees)	Average Error (degrees)
0	0.00
10	1.00
-10	0.67
20	3.00
-20	1.00

Table C.3: A table showing the unloaded pitch test results for accuracy

Pitch Input (degrees)	Average Error (degrees)
0	0.00
10	0.33
-10	0.67
20	1.00
-20	0.67

Table C.4: A table showing the loaded pitch test results for accuracy

C.1.3 Yaw

Yaw Input (degrees)	Average Error (degrees)
0	0.00
10	0.67
-10	0.33
20	1.67
-20	0.33
30	2.33
-30	1.00

Table C.5: A table showing the unloaded yaw test results for accuracy

Yaw Input (degrees)	Average Error (degrees)
0	0.00
10	0.67
-10	0.33
20	1.67
-20	0.33
30	2.33
-30	1.00

Table C.6: A table showing the loaded yaw test results for accuracy

C.1.4 X-Y Translation

X Input (mm)	Y Input (mm)	Average X Error (mm)	Average Y Error (mm)
0	0	0.00	0.00
0	10	0.00	0.33
0	-10	0.00	0.33
0	20	0.00	0.33
0	-20	0.00	0.67
0	30	0.00	1.00
0	-30	0.33	1.33
10	0	1.00	0.33
10	10	1.00	0.00
10	-10	0.67	0.67
10	20	1.00	0.33
10	-20	0.67	1.00
10	30	0.67	1.67
10	-30	0.67	2.00
-10	0	0.67	0.33
-10	10	0.33	0.67
-10	-10	0.67	0.33
-10	20	0.00	1.00
-10	-20	0.33	0.33
-10	30	0.67	1.33
-10	-30	1.00	0.67
20	0	1.67	1.33
20	10	1.67	0.67
20	-10	1.67	1.67
20	20	1.67	0.33
20	-20	1.67	2.00
20	30	2.00	1.00
20	-30	1.67	2.67
-20	0	1.00	0.33
-20	10	1.00	0.67
-20	-10	1.33	0.33
-20	20	1.00	1.00
-20	-20	1.33	0.67
-20	30	1.00	1.67
-20	-30	1.00	1.00

X Input (mm)	Y Input (mm)	Average X Error (mm)	Average Y Error (mm)
30	0	2.67	2.33
30	10	2.67	2.00
30	-10	2.33	2.67
30	20	2.67	1.67
30	-20	2.67	3.00
30	30	3.00	1.33
30	-30	2.67	4.00
-30	0	1.33	0.67
-30	10	2.33	1.67
-30	-10	2.00	1.00
-30	20	2.33	1.33
-30	-20	1.67	1.67
-30	30	1.67	1.67
-30	-30	1.33	1.33

Table C.7: A table showing the unloaded x-y translation test results for accuracy

X Input (mm)	Y Input (mm)	Average X Error (mm)	Average Y Error (mm)
0	0	0.00	0.00
0	10	0.00	0.33
0	-10	0.33	0.33
0	20	0.33	0.33
0	-20	0.33	1.00
10	0	1.33	0.00
10	10	1.33	0.00
10	-10	0.67	0.33
10	20	2.00	0.67
10	-20	0.33	0.67
-10	0	1.00	0.00
-10	10	0.33	0.33
-10	-10	1.33	0.00
-10	20	0.33	1.00
-10	-20	1.33	0.67
20	0	3.00	0.67
20	10	3.33	0.33
20	-10	2.67	1.33
20	20	3.00	0.33
20	-20	2.67	1.67
-20	0	1.00	0.33
-20	10	1.00	1.00
-20	-10	1.33	0.33
-20	20	0.67	1.67

Table C.8: A table showing the loaded x-y translation test results for accuracy

C.1.5 Z Translation

Z Input (mm)	Average Error (mm)
120	0.00
130	0.00
140	0.33
150	0.33
160	0.67
170	1.00
180	1.00
190	1.00
200	1.33

Table C.9: A table showing the unloaded z translation test results for accuracy

Z Input (mm)	Average Error (mm)
120	0.00
130	0.00
140	1.00
150	1.00
160	1.67
170	2.33
180	2.33
190	2.67
200	2.67

Table C.10: A table showing the loaded z translation test results for accuracy

C.2 Precision

C.2.1 Roll

Roll Input (degrees)	Standard Deviation (degrees)
0	0.00
10	0.00
-10	0.47
20	0.47
-20	0.47
30	0.94
-30	0.82

Table C.11: A table showing the unloaded roll test results for precision

Roll Input (degrees)	Standard Deviation (degrees)
0	0.00
10	0.00
-10	0.00
20	0.00
-20	0.00

Table C.12: A table showing the loaded roll test results for precision

C.2.2 Pitch

Pitch Input (degrees)	Standard Deviation (degrees)
0	0.00
10	0.00
-10	0.94
20	0.00
-20	0.94

Table C.13: A table showing the unloaded pitch test results for precision

Pitch Input (degrees)	Standard Deviation (degrees)
0	0.00
10	0.47
-10	0.47
20	1.41
-20	0.47

Table C.14: A table showing the loaded pitch test results for precision

C.2.3 Yaw

Yaw Input (degrees)	Standard Deviation (degrees)
0	0.00
10	0.94
-10	0.47
20	0.94
-20	0.47
30	0.47
-30	0.00

Table C.15: A table showing the unloaded yaw test results for precision

Yaw Input (degrees)	Standard Deviation (degrees)
0	0.00
10	0.47
-10	0.00
20	0.47
-20	0.47

Table C.16: A table showing the loaded yaw test results for precision

C.2.4 X-Y Translation

X Input (mm)	Y Input (mm)	X Standard Deviation(mm)	Y Standard Deviation (mm)
0	0	0.00	0.00
0	10	0.00	0.47
0	-10	0.00	0.47
0	20	0.00	0.47
0	-20	0.00	0.94
0	30	0.00	0.82
0	-30	0.47	1.25
10	0	0.00	0.47
10	10	0.00	0.00
10	-10	0.47	0.47
10	20	0.00	0.47
10	-20	0.47	0.82
10	30	0.47	0.94
10	-30	0.47	0.82
-10	0	0.47	0.47
-10	10	0.47	0.47
-10	-10	0.47	0.47
-10	20	0.00	0.82
-10	-20	0.47	0.47
-10	30	0.47	1.25
-10	-30	0.82	0.94
20	0	0.47	1.63
20	10	0.47	0.47
20	-10	0.47	1.25
20	20	0.47	0.47
20	-20	0.47	0.82
20	30	0.82	0.82
20	-30	0.47	0.47
-20	0	0.00	0.47
-20	10	0.00	0.47
-20	-10	0.47	0.47
-20	20	0.00	0.82
-20	-20	0.47	0.82
-20	30	0.00	1.25
-20	-30	0.82	1.25

X Input (mm)	Y Input (mm)	X Standard Deviation(mm)	Y Standard Deviation (mm)
30	0	0.47	2.87
30	10	0.47	1.41
30	-10	0.47	2.05
30	20	0.47	1.63
30	-20	0.47	1.63
30	30	0.82	1.25
30	-30	0.47	1.63
-30	0	0.47	0.47
-30	10	0.47	1.63
-30	-10	0.82	1.41
-30	20	0.47	1.41
-30	-20	0.47	1.70
-30	30	0.47	1.63
-30	-30	1.25	0.94

Table C.17: A table showing the unloaded x-y translation test results for precision

X Input (mm)	Y Input (mm)	X Standard Deviation(mm)	Y Standard Deviation (mm)
0	0	0.00	0.00
0	10	0.00	0.47
0	-10	0.47	0.47
0	20	0.47	0.47
0	-20	0.47	0.82
10	0	0.47	0.00
10	10	0.94	0.00
10	-10	0.47	0.47
10	20	0.82	0.47
10	-20	0.47	0.94
-10	0	0.00	0.00
-10	10	0.47	0.47
-10	-10	0.47	0.00
-10	20	0.47	1.25
-10	-20	0.47	0.82
20	0	0.82	0.94
20	10	0.94	0.47
20	-10	0.94	1.25
20	20	0.82	0.47
20	-20	0.94	0.94
-20	0	0.82	0.47
-20	10	0.82	0.94
-20	-10	0.94	0.47
-20	20	0.47	1.70

Table C.18: A table showing the loaded x-y translation test results for precision

C.2.5 Z Translation

Z Input (mm)	Standard Deviation (mm)
120	0.00
130	0.00
140	0.47
150	0.47
160	0.94
170	1.25
180	1.25
190	1.25
200	1.41

Table C.19: A table showing the unloaded z translation test results for precision

Z Input (mm)	Standard Deviation (mm)
120	0.00
130	0.00
140	0.82
150	0.82
160	1.25
170	1.25
180	1.25
190	1.70
200	1.70

Table C.20: A table showing the loaded z translation test results for precision