Abstract

This thesis explores the construction of a ball balancing platform, a control system that stabilizes an inherently unstable metallic ball on a plate using robust control algorithms working in synchronization with real-time sensory feedback. The platform is mounted on top of a 3RRS (Revolute-Revolute-Spherical) parallel manipulator. The platform utilizes a touch screen (resistive touchpad) to detect the ball's position and employs three stepper motors to adjust the plate's tilt, aiming to minimize the error between the ball's actual and desired positions. The effectiveness of a PID (Proportional-Integral-Derivative) controller is evaluated within the linear domain of control. A Simulink model of the same is developed using Lagrangian Mechanics for a better theoretical understanding and in-depth analysis of the proposed control strategy. The thesis first establishes a theoretical model of the ball-on-platform scenario using MATLAB Simulink environment, then compares these theoretical results with those obtained from the physical construction.

Keywords: Mechatronics, Control Theory, PID, Microcontrollers, Balancing, Parallel Manipulator, Simulink.

Abstrakt

Niniejsza praca bada konstrukcję platformy do balansowania piłki, systemu sterowania, który stabilizuje naturalnie niestabilną metalową piłkę na płycie, wykorzystując solidne algorytmy sterowania działające w synchronizacji z informacjami zwrotnymi w czasie rzeczywistym. Platforma jest zamontowana na manipulatorze równoległym 3RRS (Revolute-Revolute-Spherical). Platforma wykorzystuje ekran dotykowy (rezystancyjny panel dotykowy) do wykrywania pozycji piłki i stosuje trzy silniki krokowe do regulacji nachylenia płyty, dążąc do minimalizacji błędu między rzeczywistą a pożądaną pozycją piłki. Skuteczność sterownika PID (Proporcjonalno-Integralno-Derivative) jest oceniana w ramach kontrolowania w liniowym obszarze sterowania. Został opracowany model w Simulinku oparty na mechanice Lagrange'a w celu lepszego zrozumienia teoretycznego i dogłębnej analizy proponowanej strategii sterowania. Praca najpierw opracowuje teoretyczny model scenariusza piłki na platformie przy użyciu środowiska MATLAB Simulink, a następnie porównuje wyniki teoretyczne z wynikami uzyskanymi z fizycznej konstrukcji.

Słowa kluczowe: Mechatronika, Teoria sterowania, PID, Mikrokontrolery, Balansowanie, Manipulator równoległy, Simulink.

Abbreviations

PID	Proportional-Integral-Derivative
DOF	Degrees of Freedom
RRS	Revolute-Revolute-Spherical
DC	Direct Current
PV	Process variable
SP	Setpoint

Table of Contents

1 Introduction

Control theory, a subfield of mathematics, plays a pivotal role in understanding and regulating dynamic systems. The goal of a controller is to stabilize an inherently unstable system or to modulate system variables to achieve a desired outcome. Emerging from foundational Mathematical principles, Control theory is deeply intertwined with modern technological advancements. This chapter explores the evolution of Control theory, use of PID controller and outlines the Application and Scope of the proposed work.

1.1 The Evolution of Control theory

Control theory is rooted in classical mechanical systems like water clocks[1] and windmills. In the 17th-century, Christiaan Huygens invented the pendulum clock[2]. While his invention of the pendulum clock in 1656 revolutionized timekeeping, his design was not to study feedback mechanisms. In the 18th century, Huygens' centrifugal governor[3] was adopted by James Watt in the steam engine, an early example of feedback control in a mechanical system. In the 19th century, control concepts were beginning to mathematically conceptualize. For instance, a well-known landmark paper is James Clerk Maxwell's 1868 paper "On Governors" [1], which developed the mathematical analysis of feedback control systems.

Only during the first half of the 20th century did modern control theory begin to take shape, impelled by contributions arising especially from electrical engineering, with now standard tools such as Bode plots and Root Locus diagrams. It received a new boost during World War II, spurred on by breakthroughs in missile guidance and radar systems. The rise of more practical control techniques was most associated with the invention of the PID controller which will also be the central topic of this thesis, and the foundational work by Rudolf E. Kálmán, who proposed the Kalman filter [2] and thus revolutionized estimation and control.

Then came state-space methods and optimal control in the second half of the 20th century-the robust control frameworks really signified the turning to complex and nonlinear systems. Digital computation has enabled substantial advances in the areas of adaptive control and nonlinear control, plus the more recent approaches of fuzzy logics and machine-learning-based control systems.

[1] https://en.wikipedia.org/wiki/Water_clock
[2] https://en.wikipedia.org/wiki/Pendulum_clock
[3] https://en.wikipedia.org/wiki/Centrifugal_governor

1.2 PID Controller

A PID controller is used to dynamically regulate the platform's orientation by combining three control terms: proportional, integral, and derivative. The proportional term addresses the current error, the integral term eliminates small steady-state errors, and the derivative term predicts and counters rapid changes in the error for improved stability.

A PID or three term controller is a feedback-based control loop mechanism commonly used to manage machines and processes that require continuous control and automatic adjustment. It is typically used in industrial control systems and various other applications where constant control through modulation is necessary without human intervention. The PID controller automatically compares the desired target value (SP) with the actual value of the system PV). The difference between these two values is called the error value [3].

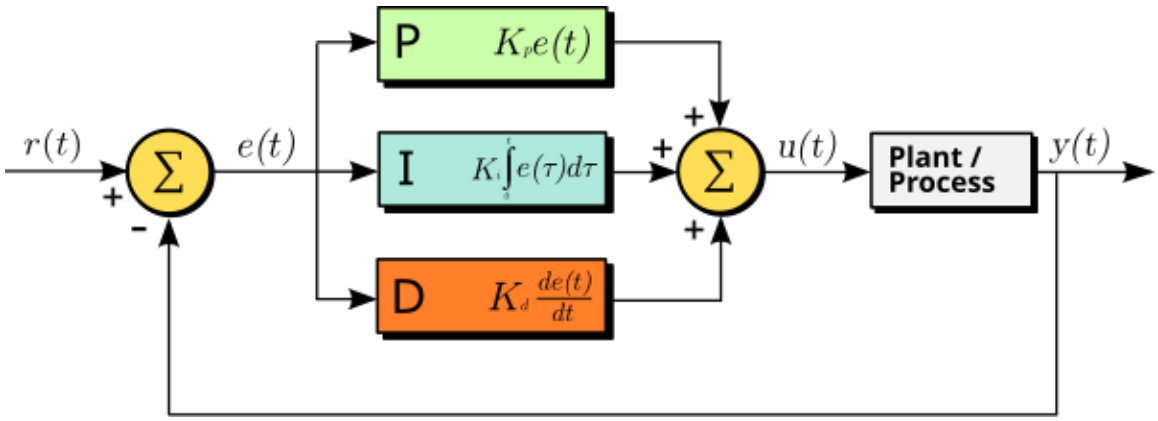

Figure 1.1 A block diagram of a PID controller in a feedback loop. r(t) is the desired process variable (PV) or setpoint (SP), and y(t) is the measured PV. [3]

The overall control function $u(t)$ is described by,

$$u(t) = K_p \times e(t) + K_i \int e(t)dt + K_p \frac{de}{dt} \quad (1)$$

In the case of a ball-balancing platform as described earlier, the proportional gain responds to the distance between the ball's current position and the set point. The derivative controller accounts for the speed at which the ball is moving away from the set point, while the integral term addresses any small steady-state error.

1.2.1 Proportional Control

The purpose of the proportional controller is to minimize the steady-state error by increasing the system's proportional gain. By introducing a constant K_P, known as the proportional gain, the proportional response is adjusted such that the steady-state error decreases inversely with the proportional gain. The proportional term, or output signal, is defined by equation (5.2).

$$u(p) = K_p e(t) \tag{2}$$

However, if the gain is set too high, it may lead to instability in the system by causing excessive output fluctuations. On the other hand, if the gain is too low, the system may struggle to effectively handle both external and internal disturbances.

1.2.2 Integral Control

The integral controller is proportional to both the duration and magnitude of the error over time. It corrects any accumulated offset in the system by summing the errors over a finite period, effectively eliminating steady-state error. However, the integral gain K_i can sometimes degrade the system's response, potentially causing transient or oscillatory behaviour. The output signal from the integral term is given by equation (5.3).

$$u_i(t) = K_i \int_0^t e(t)\ d(t) \tag{3}$$

1.2.3 Derivative Control

The derivative controller is determined through the calculation of the error's response slope over time. The slope of the error is then multiplied with K_D, the derivative gain. The derivative term is defined in equation (5.4).

$$u_D(t) = K_D \frac{d\ e(t)}{dt} \tag{4}$$

1.3 Simulink Modelling

MATLAB/Simulink modeling provides a comprehensive environment for designing, simulating, and analyzing dynamic control systems through block diagram representation.

1.4 Application and Scope

This study investigates the implementation and effectiveness of linear control methodologies in stabilizing a ball on a flat platform. The work encompasses both theoretical modelling of the proposed system in Simulink environment and building a working prototype. System responses to external disturbance like human interaction and environmental factors will be primarily studied.

Methodology:

The work will be carried out on dual tracks: one theoretical simulation and the other physical experimentation. Arduino UNO based Microcontroller, Maker UNO will be the brains of the system. Performance evaluation will be based on steady-state stability analysis, but with particular interest in response characteristics under controlled disturbance conditions. More about performance evalution is described in Section 2.2. Comparative analysis between simulated and experimental results is intended for identification and quantification of important performance factors and system limitations.

Research Questions:

1. Is the linear control methodology adequate for satisfactory performance in a ball-balancing platform system? This question looks into the adequacy of linear control techniques in managing an inherently nonlinear mechanical system.
2. Is it possible to create a digital twin of the prortotype? If not than what are the causes of these differences between theoretical simulations and actual experiments? The project aims at finding out the source of the difference between the theoretically predicted behavior and practically observed behavior and to quantify the causes-mechanically, electrically, and environmentally.
3. Is it possible to trace different shapes by changing the SP at a regular interval and moving the ball accordingly ?
4. Can a software be developed to define the path of the ball by manualling moving it and have the software repeat the traced path?

2 Objectives and Requirements

This chapter outlines the objectives of the project in a systematic order of implementation. There are some research-oriented objectives as well for which the results cannot be anticipated and hence are not considered for measuring performance metric. Requirements for both hardware and software development are listed as well.

2.1 Objectives

The objectives of the project are as follows:

1. **Construction:** Fabricate and assemble the platform successfully.
2. **Balancing:** Achieve precise balancing of a metallic ball on the platform with PID control.
3. **Simulation:** Develop a Simulink model to simulate the system.
4. **Drawing Curves:** Implement control to enable the ball to draw predefined figures on the platform.
5. **Research**: Conduct research of other control strategies that can be employed.

2.2 Performance Metrics

The outcome of the work and the system's success will be measured by the following metrics:

1. **Stability**: The time taken to stabilize the ball after an external disturbance. Duration less tha 3s is considered as exceptional.
2. **Precision**: The accuracy of the ball's position control within a predefined error range.
3. **Steady State Error**: Average offset between the set position and the actual position
4. **Speed**: The system's response time to input changes or disturbances.
5. **Drawing Accuracy**: The deviation between the predefined figure and the actual path traced by the ball.

2.3 Assumptions

This section outlines the assumptions made in work. As the Ball Balancing Platform is a demonstration built in a lab environment, the required assumptions are not very rigid. The following assumptions were made:

1. The platform's operation is intended to be demonstrated in a lab environment.
2. A PID controller will be used primarily. Similar control strategies may be explored.

3. No Friction is considered.
4. The ball that is to be balanced is perfectly spherical and has a smooth surface.
5. The ball does not slide on the platform or moves upwards.
6. The ball in static conditions is unstable even without external disturbances.
7. Vibrations and jerky motion are not considered for simulation.
8. The speed of actuation (the angular velocity of motors) is not taken into consideration for simulation.

2.4 Requirements for the projects

2.4.1 Electronics

1. Maker Uno Microcontroller
2. Nema 17 59 Ncm Stepper Motors (Bipolar)
3. TMC2209 V2.0 Stepper Motor Drivers
4. CNC shield
5. 30V Bench Power Supply (or any 24V power supply)
6. 5V Regulator

2.4.2 General Parts

1. 8.4" 4 Wire Resistive Touch Panel
2. 1" Steel Bearing Ball
3. 22mm long m3 tie rod
4. M3 x 6mm threaded inserts
5. M3 x 5mm Standoffs
6. M3 x 5mm Screws
7. M3 x 8mm Screws
8. M3 x 10mm Screws
9. M3 x 35mm Screws
10. M3 Nylon Locknuts
11. M4 x 20mm Screws
12. M4 x 25mm Screws
13. M4 Nylon Locknuts

The requirement list is supplemented with items like General mechanical tools, Soldering Iron, Electrical wire, and 3D printer.

3 Mechanical Configuration

This chapter details the mechanical design of the 3RRS parallel manipulator platform. The 3RRS platform has **3 DOF and three points of actuation**. Three stepper motors with TMC2209 drives are used for precise actuation.

The design was taken from **Aaed Musa**'s Ball Balancing platform [4] with slight modifications to accommodate different housing sizes. But the overall working principle remains the same.

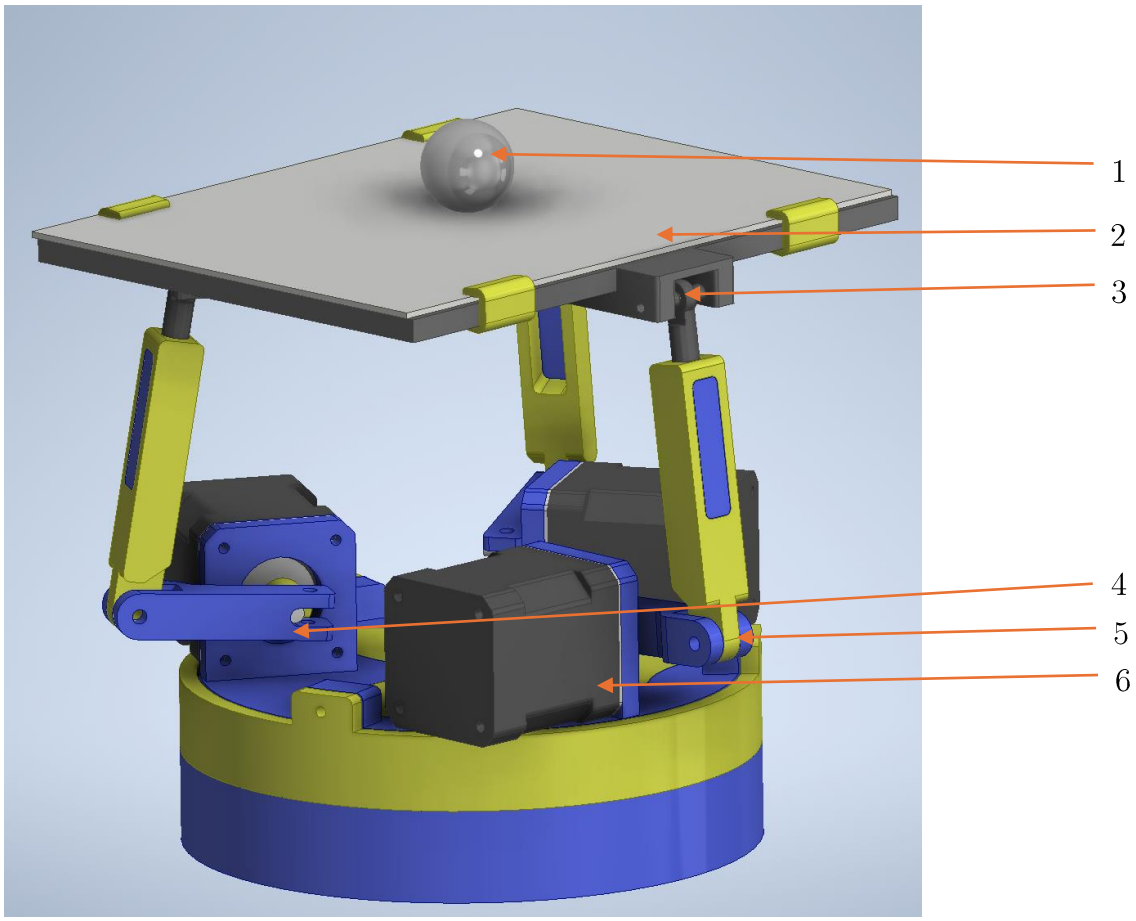

Figure 3.1 3RRS Ball Balancer mechanical design (description below)

A metallic ball **1** is placed on a resistive touchpad **2** which outputs the position of the ball by detecting the contact point and returning it as a cartesian point. The platform is connected to the actuator legs with a spherical joint **3.** The actuator leg has two links connected by a revolute joint **5.** The lower leg is connected to the rotor of the stepper motor **6** with a revolute joint **4.**

[4] https://www.youtube.com/watch?v=v4F-cGDGiEw

4 Electrical Configuration

This chapter outlines the electrical configuration of the project, including the Maker UNO, TMC2209 stepper driver, CNC shield, stepper motor, and resistive touchpad. A NEMA 17 stepper motor provides accurate movement, with manual homing at power-up. A resistive touchpad is employed to track the ball's position, offering a simple and reliable solution.

4.1 Arduino UNO

Maker UNO is a similar microcontroller to Arduino UNO. It has an identical pinout configuration as compared to Arduino UNO. The ease of use, versatile connectivity with ample IO ports and analogue ports, native programming IDE and real time control with low latency make it an ideal choice.

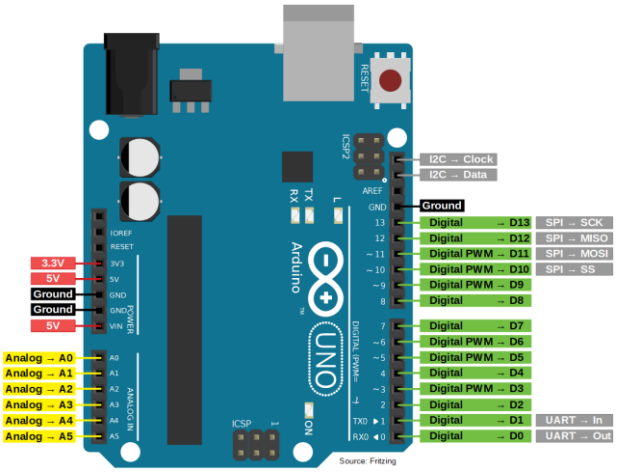

Figure 4.1 Arduino UNO pinout diagram. Maker UNO has an identical configuration.

Table 4.1 Maker UNO technical specification.

Category	Specification
Microcontroller	ATmega328P
Operating Voltage	5V
Digital I/O Pins	14
Analog Input Pins	6
Flash Memory	32 KB (ATmega328P)
Clock Speed	16 MHz

4.2 TMC2209 V2.0

The TMC2209 is known for its silent operation. Low noise levels, different micro stepping options, current control and low heat generation makes it an ideal choice to control the stepper motors. The V_{ref} is set using a potentiometer screw.

$$V_{ref} = I_{motor} \times 8 \times R_{sense} \tag{5}$$

From the stepper documentation,

$$I_{motor} = 1.5\,A$$
$$R_{sense} = 0.11\,\Omega$$
$$V_{ref} = 1.5A \times 0.11\,\Omega \times 8$$

$$\boldsymbol{V_{ref} = 1.32\,V}$$

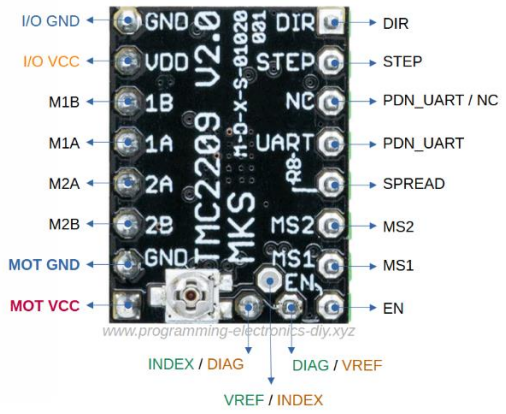

Figure 4.2 TMC2209 V2.0 stepper driver pinout diagram.

Table 4.2 TMC2209 Micro stepping Configuration.

MS1	MS2	Micro stepping Mode
Low	Low	Full Step (1x)
High	Low	Half Step (2x)
Low	High	Quarter Step (4x)
High	High	Sixteenth Step (16x)

Table 4.3 TMC2209 technical specifications.

Specification	Value
Operating Voltage (VM)	4.75V to 29V DC
Output Current	2.0A RMS, 2.4A Peak (per phase)
Operating Temperature	-40°C to +150°C

4.3 CNC Shield

A CNC shield will be used instead of manually connecting the jumpers through a breadboard. The CNC shield has built in decoupling capacitors and integrates with Maker UNO seamlessly. Addition of CNC shield makes electrical management easy.

Micro stepping can be configured by using onboard micro stepping jumpers.

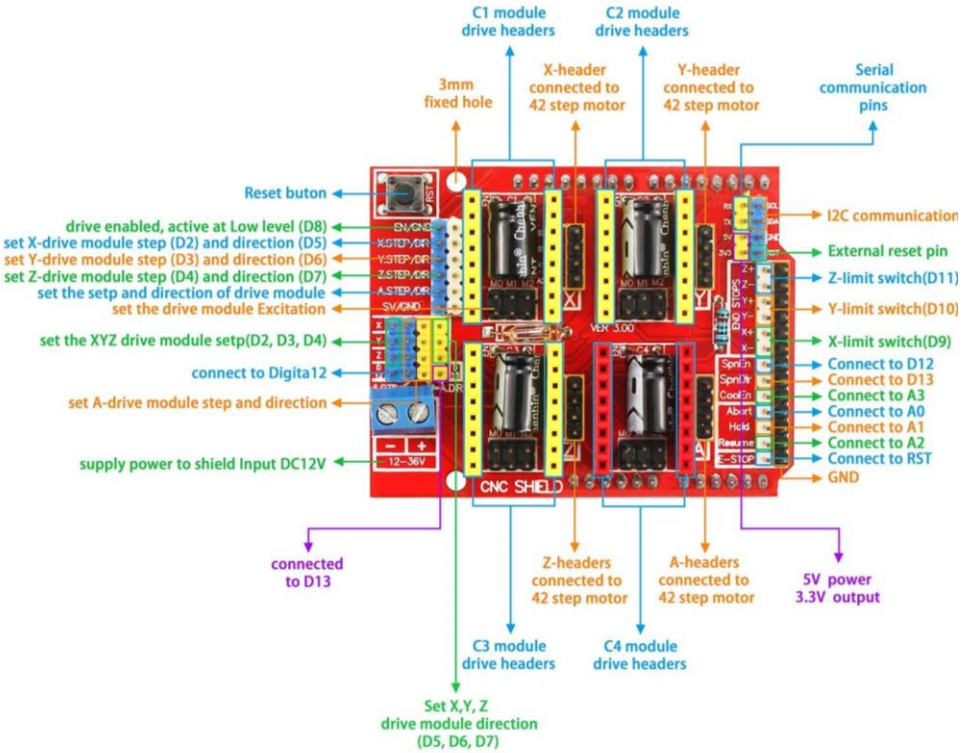

Figure 4.3 CNC shield pinout diagram

Table 4.4 Cross-referencing of Arduino pins

UNO Pin	CNC Shield Pin	Function
Pin 2	X Step	Step signal for X-axis motor
Pin 3	Y Step	Step signal for Y-axis motor
Pin 4	Z Step	Step signal for Z-axis motor
Pin 5	X Dir	Direction signal for X-axis motor
Pin 6	Y Dir	Direction signal for Y-axis motor
Pin 7	Z Dir	Direction signal for Z-axis motor
Pin 8	Enable	Enable signal for X-axis motor

In addition to the digital pins, analogue pins are also used. The resistive touchpad is connected to A0, A1, A2 and A3 pins shown in the schematics.

4.4 Stepper Motor

NEMA 17 bi-polar stepper motors from STEPPERONLINE were used for the project. Cost-efficient, precise control and low noise make it an ideal choice. Positional encoders are not used. Instead, homing is done by manually pushing the platform down. When the steppers are powered off the platform automatically falls to the home position.

Figure 4.4 Nema 17 Bi-polar stepper motor.

Table 4.5 Nema 17 technical specifications

Property	Value
Rated Current (A)	1.5
Step Angle (deg.)	1.8
Bipolar/Unipolar	Bipolar
Holding Torque (Ncm)	42

4.5 Resistive touchpad[5]

A four-wire resistive touchpad works by detecting touch through two flexible, transparent resistive layers that come into contact when pressure is applied. Each layer has electrodes along its edges: the top layer for the X-axis and the bottom layer for the Y-axis. To determine the touch point, a voltage is first applied across one layer (e.g., X-axis), and the resulting voltage at the contact point is measured to find the horizontal position. Then, a voltage is applied across the other layer (Y-axis) to measure the vertical position. By alternating these measurements, the touchpad calculates the X and Y coordinates of the touch point using the voltage divider principle.

[5] https://www.sparkfun.com/datasheets/LCD/HOW%20DOES%20IT%20WORK.pdf

5 System Modelling

This chapter explains how the system is implemented to achieve the desired outputs. To achieve the desired results, three aspects must be implemented:

1. Control Software
2. Inverse Kinematics
3. A model of the ball's motion, used solely for simulation purposes

5.1 Control Software

These aspects will be designed in detail in the upcoming sections.

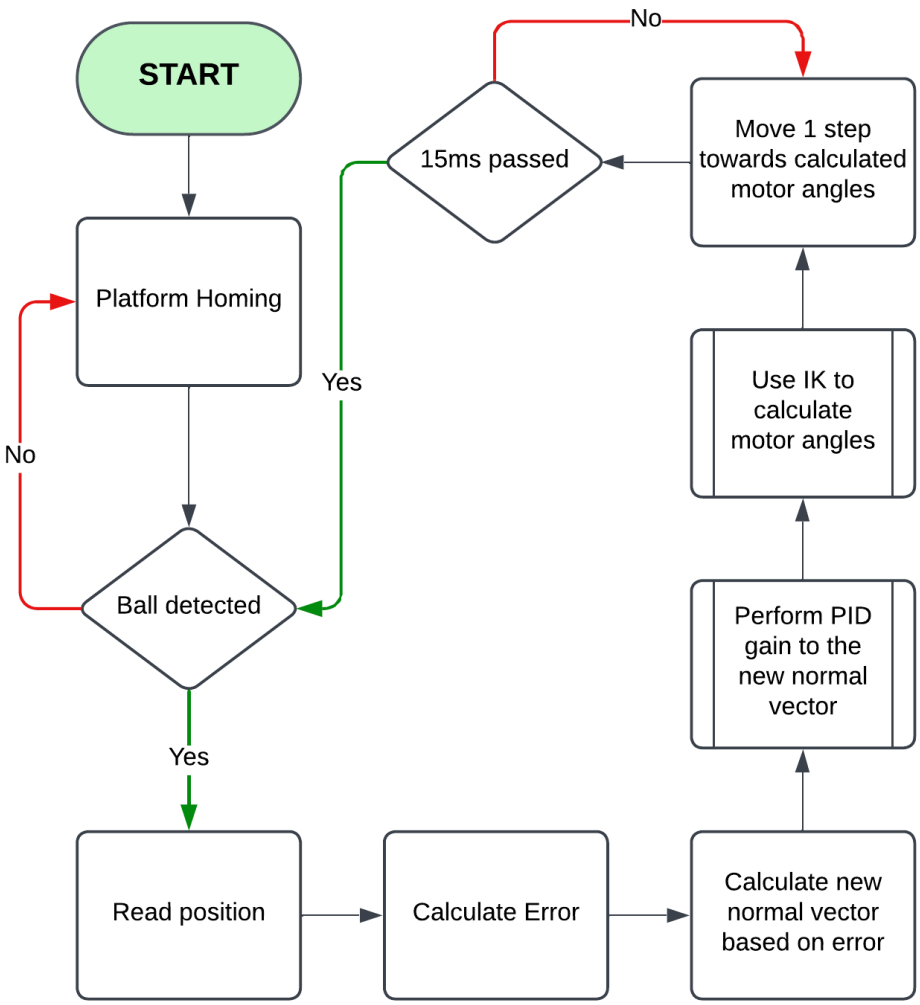

Figure 5.1 Control Algorithm describing the process.

The following is a detailed description of the flowchart shown in Figure 5.1:

1. **Start**: The process begins.
2. **Platform Homing**: The platform undergoes a homing sequence, moving to a predefined position and orientation described by the normal vector [0,0,4.25] to ensure a consistent starting point. When powering on the system for first time it is essential for the motor to be pushed down.
3. **Ball detection:** The system checks whether a ball is present on the platform. Depending on the outcome the following happens:
    - If no ball is detected, the system returns to the 2nd step.
    - If detected, the process continues to the next step.
4. **Read Position**: The current position of the ball is identified by reading the touchpad measurements.
5. **Calculate Error**: The error is calculated, which is the difference between the ball's current position and the Set point of the platform. This error indicates how far the ball is from the set position in [x, y] vector format.
6. **Calculate New Normal Vector**: Based on the calculated error, a new normal vector is computed by subtracting the Error from the normal vector describing the platform's current position. This new vector represents the required orientation to move the ball toward the centre.
7. **Perform PID Gain to the New Normal Vector**: A PID controller is implemented. The output of PID controller is constrained between the range [-0.25,0.25] to prevent the platform from over-tilting.
8. **Use IK to Calculate Motor Angles**: Pre-defined Inverse Kinematics (IK) equations is utilized to translate the desired platform tilt, represented by the normal vector, into specific angles for each motor that controls the platform.
9. **Move 1 Step Towards Calculated Motor Angles**: The motors are incrementally adjusted with a non-blocking function, moving towards the calculated angles. This gradual movement ensures smooth control and prevents abrupt shifts that might destabilize the ball.
10. **15ms wait:** The system introduces a wait time of 15 milliseconds for the steppers to reach the desired angle. The code moves to next step even if the required motor angles are not reached after 15ms.

11. **Loop**: If the ball is detected by the touchpad the loop continues, if ball is not detected the platform returns to home.

In essence, the system starts by homing the platform, then continuously detects the ball's position and calculates the necessary platform tilt using a PID controller and inverse kinematics. Finally, it moves the platform's motors incrementally to adjust its position, maintaining the ball's balance, looping back until no longer detected.

5.2 Inverse Kinematics of the system

In this sub-chapter, the inverse kinematics equations required are described to orient the platform in such a way that it sends the ball towards setpoint. The platform is supported by three legs, each modelled as a two-link manipulator connecting the base to the platform. The configuration of each leg consists of two revolute joints and two rigid links, forming a planar structure.

Given a unit normal vector Z' of the platform plane, the IK equations then compute the required angles of the revolute joint B_1 , B_2 *and* B_3 [4].

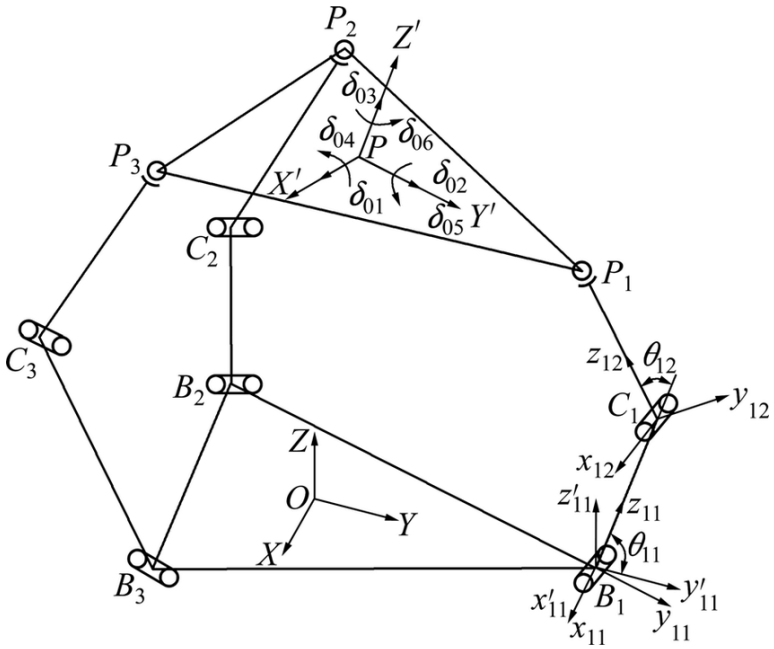

Figure 5.2 Kinematic sketch of 3RRP parallel manipulator - for representation only. [4]

Table 5.1 Description of symbols in Figure 4.2

SYMBOL	DESCRIPTION
O	Origin of the global co-ordinate system

$B_1, B_2, B_3, C_1, C_2, C_3$	Revolute Joints
P_1, P_2, P_3	Spherical Joints
P	Platform Plane origin
Z'	Unit normal vector of the platform plane

The system is divided into three separate kinematic chains. The origin for the global frame (O) is marked at the mid-point of the base plane. The mid-point of the platform plane is represented by O_1.The required rotor angles is calculated as a function of tilt of the platform plane represented by $\vec{N}$ and the height between the platforms represented by h.

$$[\,\theta_1, \theta_2, \theta_3] = f(\vec{N}\,, h\,) \tag{6}$$

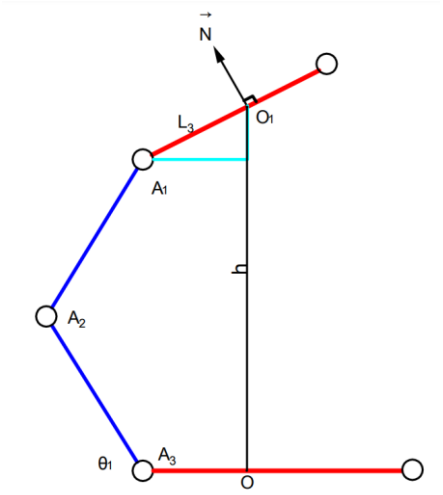

Figure 5.3 Kinematic diagram for a single chain

Figure 5.4 represents one kinematic chain out of three.

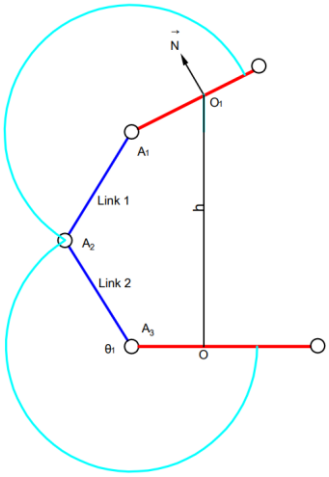

Figure 5.4 Kinematics of Link 1 and Link 2

Unit normal Vector:

$$n_{mag} = \sqrt{n_x^2 + n_y^2 + 1} \tag{7}$$

$$n_x = \frac{n_x}{n_{mag}}, \quad n_y = \frac{n_y}{n_{mag}}, \quad n_z = \frac{1}{n_{mag}}$$

Equations for $\boldsymbol{\theta_1}$:

$$\boldsymbol{\theta_1} = \arccos\left(\frac{y}{mag}\right) + \arccos\left(\frac{mag^2 + f^2 - g^2}{2 \cdot mag \cdot f}\right) \tag{8}$$

Where:

$$y = d + \frac{e}{2}\left(1 - \frac{n_x^2 + 3n_z^2 + 3n_z}{n_z + 1 - n_x^2 + \frac{n_x^4 - 3n_x^2 n_y^2}{(n_z + 1)^2}}\right)$$

$$z = h_z + en_y$$

$$mag = \sqrt{y^2 + z^2}$$

Equations for $\boldsymbol{\theta_2}$:

$$\boldsymbol{\theta_2} = \arccos\left(\frac{\sqrt{3}x + y}{-2 \cdot mag}\right) + \arccos\left(\frac{mag^2 + f^2 - g^2}{2 \cdot mag \cdot f}\right) \tag{9}$$

Where:

$$x = \frac{\sqrt{3}}{2}\left(e\left(1 - \frac{n_x^2 + \sqrt{3}n_x n_y}{n_z + 1}\right) - d\right)$$

$$y = \frac{x}{\sqrt{3}}$$

$$z = h_z - \frac{e}{2}\left(\sqrt{3}n_x + n_y\right)$$

$$mag = \sqrt{x^2 + y^2 + z^2}$$

Equations for $\boldsymbol{\theta_3}$:

$$\boldsymbol{\theta_3} = \arccos\left(\frac{\sqrt{3}x - y}{2 \cdot mag}\right) + \arccos\left(\frac{mag^2 + f^2 - g^2}{2 \cdot mag \cdot f}\right) \tag{10}$$

Where:

$$x = \frac{\sqrt{3}}{2}\left(d - e\left(1 - \frac{n_x^2 - \sqrt{3}n_x n_y}{n_z + 1}\right)\right)$$

$$y = \frac{-x}{\sqrt{3}}$$

$$z = h_z + \frac{e}{2}\left(\sqrt{3}n_x - n_y\right)$$

$$mag = \sqrt{x^2 + y^2 + z^2}$$

5.3 Equations describing motion of the ball

To simulate the movement of the ball, a model describing the equations of motion of the ball is necessary. The three-dimensional ball-on-platform system will be split into two separate two-dimensional ball-on-beam systems, as shown in Fig. 5.5. These systems will share the same equations but with different variables. Index 1 will refer to the system observed in the x-z plane of the room, while index 2 will correspond to the system viewed in the y-z plane. The equations will be derived for the first system and later translated into their equivalent form for the second system.

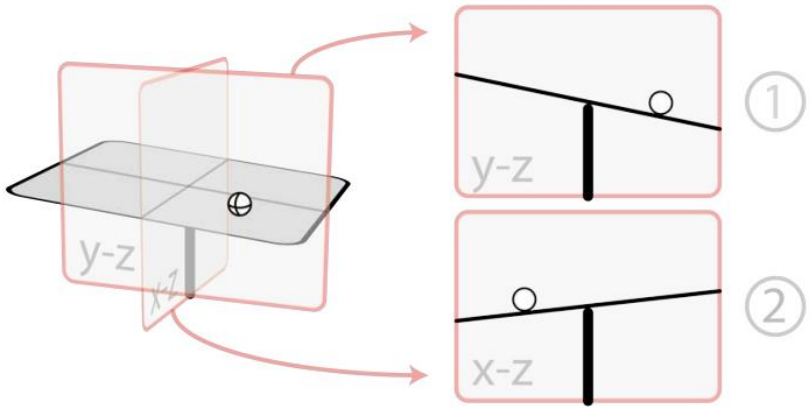

Figure 5.5 Three-dimensional system represented as a pair of two-dimensional system [5]

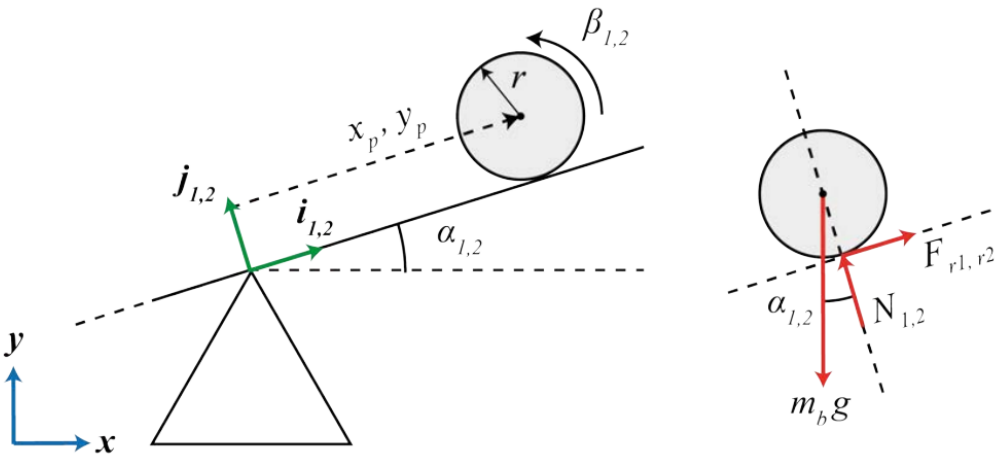

Figure 5.6 Free body diagram of the ball in two dimension [5]

The goal is to model the motion of the ball in XY plane given the tilt angles of the YZ and XZ planes.

Lagrangian Method: Equations of Motion

The Lagrangian Method derives the equation of motion through the relation of kinetic and potential energy of the system. This method is particularly useful for complex systems with multiple degrees of freedom. Below are the detailed derivations and descriptions. A similar approach as M. T. Alexander Hasp Frank is used to derive the equations [5].

Lagrangian Equation:

$$\frac{\partial}{\partial t}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} = 0$$

L (Lagrangian) is the difference between the kinetic and potential energy of the system:

$$L = E_{kin} - E_{pot}$$

Kinetic Energy:

$$E_{kin} = E_{kin,T} + E_{kin,R} = \frac{1}{2} m_b v_b^2 + \frac{1}{2} J_b \omega_b^2$$

Where:

$$E_{kin,T} = \frac{1}{2} m_b v_b^2 = \frac{1}{2} m_b \left(\dot{x_b^2} + \dot{y_b^2}\right)$$

$$E_{kin,R} = \frac{1}{2} J_b \omega_b^2 = \frac{1}{2} J_b \frac{v_b^2}{r_b^2} = \frac{1}{2} J_b \frac{\left(\dot{x_b^2} + \dot{y_b^2}\right)}{r_b^2}$$

Thus, the total kinetic energy becomes:

$$E_{kin} = \frac{1}{2}\left(m_b + \frac{J_b}{r_b^2}\right)\left(\dot{x_b^2} + \dot{y_b^2}\right)$$

Potential Energy:

$$E_{pot} = m_b g h_b = -m_b g x_b \sin(\alpha) - m_b g y_b \sin(\beta)$$

Lagrangian:

$$L = E_{kin} - E_{pot} = \frac{1}{2}\left(m_b + \frac{J_b}{r_b^2}\right)\left(\dot{x_b^2} + \dot{y_b^2}\right) + m_b g x_b \sin(\alpha) + m_b g y_b \sin(\beta)$$

Equation of Motion (x-direction):

$$\frac{\partial}{\partial t}\left(\frac{\partial L}{\partial \dot{x_b}}\right) = \frac{\partial}{\partial t}\left(\left(m_b + \frac{J_b}{r_b^2}\right)\dot{x_b}\right) = \left(m_b + \frac{J_b}{r_b^2}\right)\ddot{x_b}$$

$$\frac{\partial L}{\partial x_b} = m_b g \sin(\alpha)$$

Substituting these into Equation 4.1, the differential equation becomes:

$$\left(m_b + \frac{J_b}{r_b^2}\right)\ddot{x_b} - m_b g \sin(\alpha) = 0$$

Simplifying for $\ddot{x_b}$, the equation of motion is derived as:

$$\boldsymbol{\ddot{x_b} = \frac{m_b g r_b^2}{m_b r_b^2 + J_b}\sin(\alpha)} \tag{5.9}$$

Equation of Motion (y-direction):

Applying the same method to the y-direction, the equation of motion is:

$$\boldsymbol{\ddot{y_b} = \frac{m_b g r_b^2}{m_b r_b^2 + J_b}\sin(\beta)} \tag{5.10}$$

6 Construction and Assembly

The construction of the ball balancer was the most exciting part of the work. The process began with purchasing of the listed components in the Section 2.4. Followed by 3D printing of the CAD models. PETG filament was used for printing. It took about 6 days of continuous printing to make all the parts.

6.1 3D printing

3D printing is one of the most versatile ways to prototype and build nonstandard parts. All the parts with some exceptions like the touchpad and the joints were fabricated using FDM printing. Standard slicer settings with raft and higher infill density were used. It is important to take into consideration the shrinkage of parts and adjust the tolerances accordingly beforehand.

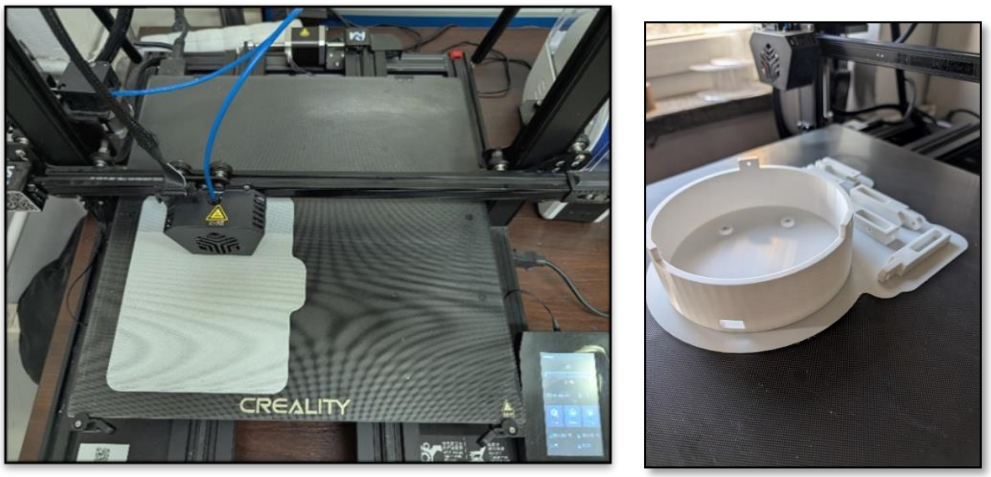

Figure 6.1 Printing the top platform (L) Printed parts (R)

6.2 Electronics

MakerUNO is the main control unit. The CNC shield is compatible with an Arduino UNO and is stacked on top of it. This eliminates the need to use protoboard and jumper wires which makes the electronics very organized and easy to handle.

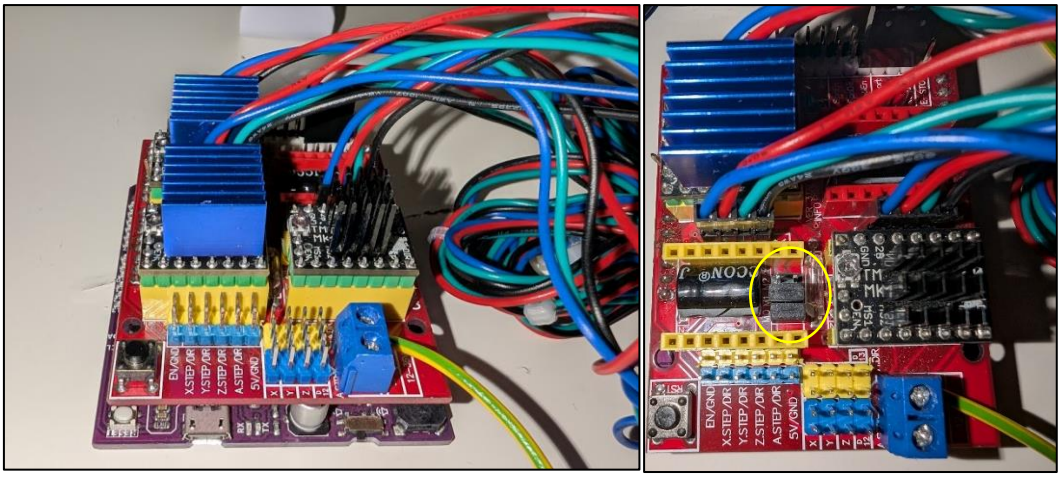

Figure 6.2 CNC shield with drivers connected to UNO (L) Micro stepping jumper pins ($1/16^{th}$ configuration) (R)

The CNC shield is powered with a 24V bench power supply. It is important to set the right micro stepping configuration ($1/16^{th}$) for a smooth motion. Heat sinks are also necessary as the drivers tend to overheat quickly.

6.3 Assembly

The assembly of the platform was done in the following sequence.

1. Spacers were added to each stepper motor rotor shaft to ensure proper alignment of legs during assembly
2. Each stepper motor was secured onto the base plate using M3 x 10mm screws (x12).
3. The Maker UNO with CNC shield was screwed to stand using M3 x 5mm screws (x4).
4. Each link1 was attached to its corresponding stepper motor using an M4 x 20mm screw and an M4 locknut.
5. A tie rod was attached to one end of each link2 using an M3 x 8mm screw.
6. Each link2 was then connected to the platform frame at the tie rod end using an M3 x 35mm screw. M3 x 5mm standoffs (x2) were used on either side of each tie rod to fill the gap.
7. The other end of each link2 was fastened to the corresponding link1 using an M4 x 25mm screw and an M4 locknut. During this step, it was ensured that Stepper A was connected to the designated side of the platform frame.
8. The base plate was mounted onto the base stand using M3 x 8mm screws (x3).
9. Finally, the resistive touchpad was clipped onto the platform frame using four retainer clips.

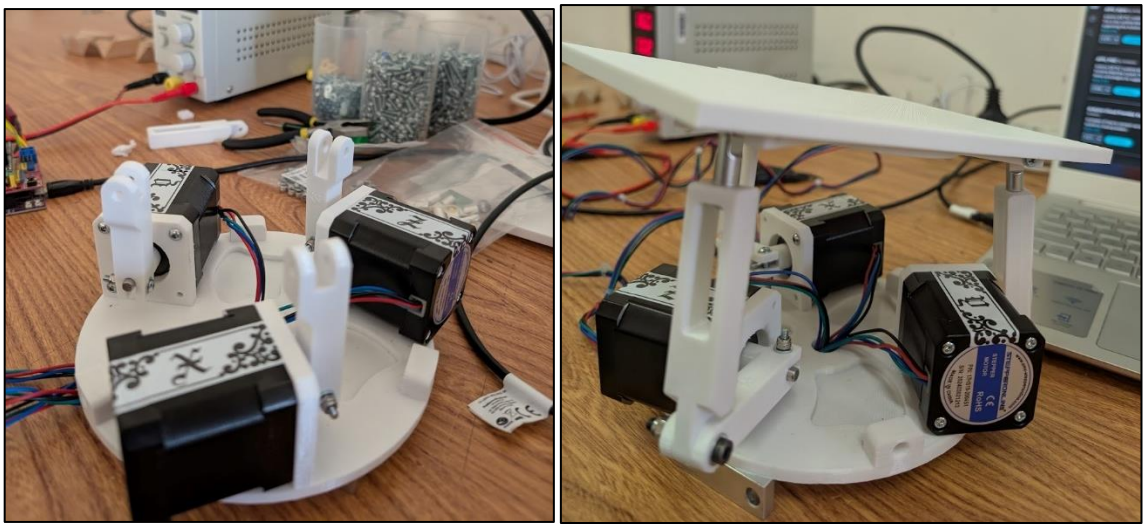

Figure 6.3 Partially assembled platforms

It is important to ensure proper connection of + and – power terminals. Small movements were performed first to ensure that everything was in order.

7 Simulink Modelling

8 PID Tuning

9 Results and Analysis

10 Further Research

Table of Figures

References

[1] J. Maxwell, "ON GOVERNORS," 1868.

[2] R. Kalman, "A New Approach to Linear Filtering," *Journal of Basic Engineering,* 1960.

[3] Wikipedia, "PID Controller," [Online]. Available: https://en.wikipedia.org/wiki/Proportional–integral–derivative_controller.

[4] B. Zi, C. Liu and Y. Yueqing, "Dynamics of 3-DOF spatial parallel manipulator with flexible links," 2010.

[5] M. T. ALEXANDER HASP FRANK, "Construction and theoretical study of ball balancing platform," Stockholm, Sweden, 2019.

[6] N. Apazidis, Mekanik II - Partikelsystem, stel kropp och analytisk mekanik, Kartonnage, Svenska, 2019.

[7] Y. Chen, System simulation techniques with MATLAB and Simulink, John Wiley & Sons, 2013.

[8] S. W. T. P. E. D. I. S. Zhen Gao, "Design and Implementation of a Ball Balancing System for Control Theory Course," *IJMEC,* 2015.