THE PROJECT ENTITLED

BALL ON PLATE BALANCING SYSTEM

Submitted in partial fulfillment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRICAL ENGINEERING

SUBMITTED By

Raj Sutariya (U14EE030)

Sudhir Babariya (U14EE038)

Himanshu Singh (U14EE039)

Sajid Ali Kadiyar (U14EE037)

Chandan K. Nayak (U14EE015)

Guide**:**

Dr. S N SHARMA

2017-18

DEPARTMENT OF ELECTRICAL ENGINEERING

SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECHNOLOGY

SURAT – 395007, GUJARAT, INDIA

SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECHNOLOGY
SURAT-395 007, GUJRAT, INDIA
DEPARTMENT OF ELECTRICAL ENGINEERING

SVNIT

CERTIFICATE

This is to certify that the project report entitled "***BALL ON PLATE BALANCING SYSTEM***" submitted by Raj Sutariya (U14EE030), Sudhir Babariya (U14EE038), Himanshu Singh (U14EE039), Sajid Ali Kadiyar (U14EE037), Chandan K. Nayak (U14EE015) is a record of bonafide work carried out by them in partial fulfillment of the requirement for the award of the degree of "BACHELOR OF TECHNOLOGY IN ELECTRICAL ENGINEERING".

Date:

Place: EED SVNIT

(Dr. S.N. Sharma)

Examiners

Head of Department

ACKNOWLEDGMENT

We would like to express my deep sense of gratitude to our guide **Dr. Shambhu N. Sharma** (Associate Professor, EED, SVNIT, SURAT) for his valuable guidance, motivation and for his extreme cooperation to complete this report.

We would like to express our sincere respect and profound gratitude to **Prof. S.N. SHARMA**, Head of Electrical Engineering Department for supporting and providing the facilities for our seminar work.

We appreciate all our colleagues whose direct and indirect contribution helped us a lot to accomplish this report. We would also like to thank all the teaching and non-teaching staffs for cooperating with us and providing valuable advice which helped us in the completion of this work.

Finally, we would like to express our thanks to all the members of our lovely family, without whose support we would have never been able to aspire for this level of education.

- **RAJ SUTARIYA**
- **HIMANSHU SINGH**
- **SUDHIR BABARIYA**
- **CHANDAN K. NAYAK**
- **SAJID ALI KADIYAR**

ABSTRACT

The report presents a ball on plate balancing system useful for various educational purposes. A touch-screen placed on the plate is used for ball's position sensing and two servomotors are employed for balancing the plate in order to control ball's Cartesian coordinates. Due to its complexity, multiple steps were taken to solve the design challenges and develop the system. The major works were mathematical modelling, kinematic constraints and dimensional analysis, simulations, and construction of the system. The control system required an effective control strategy and a thorough analysis of system parameters. The system's feasibility and optimal operation were fully considered in the design phase. The design was then validated by simulations using Simulink/MATLAB™ and experimental testing.

A complete dynamic system investigation for the ball-on-plate system is presented in this paper. This includes hardware design, sensor and actuator selection, system modelling, parameter identification, controller design and experimental testing.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1 **Introduction**

"The ball on plate problem is a benchmark for testing control algorithms."

The ball and plate system is one of the most enduringly popular and important laboratory models for teaching control systems engineering. The control of unstable systems is critically important to many of the most difficult control problems. Since the system is Open Loop Unstable that is the system output (the ball position) increases without limit for a fixed input (plate angle) so the control of such a system is significantly important.

The initial objective of the project is to maintain a static ball position on the plate, rejecting position disturbances. To control the ball on plate system, a servo control system is used. Such a system is one of the most important and widely used forms of control systems. The job of servo motor here is to maintain a specific angle corresponding to the ball position until the ball reaches its desired position. The control loop of the servo mechanism is shown.

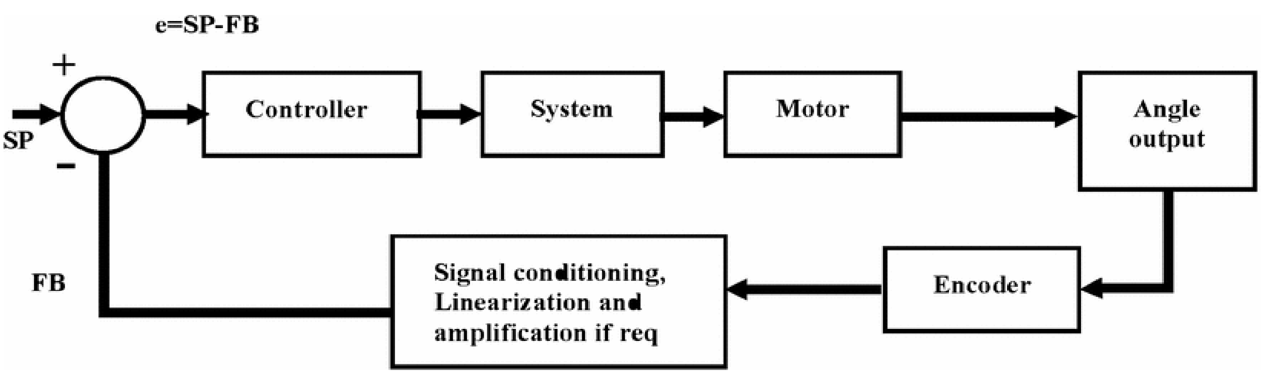


Figure 1 Control Mechanism

Considering all the above-mentioned factors, the design was developed to have following features:

- a. Quick sensing mechanism for the position of the ball.

- b. Efficient and light weight design.
- c. High performance actuators/motors.
- d. Control mechanism using PID algorithm.

The first requirement was to sense the position of the ball in real time. For that purpose, different techniques were considered. These are enlisted below with their limitations.

- a. Overhead camera and image processing. Cumbersome to develop a standalone system using this technique.
- b. Resistive plate, a kind of 2-dimensional potentiometer. Although it has complicated mechanism but due to low cost and easy design mechanism it has been used for project
- c. Grid of sensors for each axis, the mechanism required excessive wiring and provided limited resolution.

Design an actuation mechanism for the plate. The plate has to rotate about its two planer body axes, to be able to balance the ball. For this design, the following options were considered:

- Two linear actuators connected to two edges on the base of the plate that is supported by a U-joint in the center, thus providing the two necessary degrees of motion.
- Mount the plate on a gimbal ring. One motor turns the gimbal providing one degree of rotation; the other motor turns the plate relative to the ring thus providing a second degree of rotation.
- Use of cable and pulley arrangement to turn the plate using two motors (DC or Stepper).
- Use a spatial linkage mechanism to turn the plate using two motors (DC or Stepper)

Chapter 2 **Mechanical Design**

The physical system consists of an acrylic plate, an actuation mechanism for tilting the plate about two axes, a ball position sensor, instrumentation for signal processing, and real-time control software/hardware. The entire system is mounted on a wooden base plate and is supported by U-joint.

2.1 Servo motors

Servo Motors have been around for a long time and are utilized in many applications. They are small in size but pack a big punch and are very energy-efficient. These features allow them to be used to operate remote-controlled or radio-controlled toy cars, robots and airplanes. Servo motors are also used in industrial applications, robotics, in-line manufacturing, pharmaceutics and food services.

The servo circuitry is built right inside the motor unit and has a position able shaft, which usually is fitted with a gear (as shown below). The motor is controlled with an electric signal which determines the amount of movement of the shaft.

What's inside the servo?

Figure 2 SERVO MOTOR

Set-up:
a small DC motor, potentiometer, and a control circuit. The motor is attached by gears to the control wheel.

As the motor rotates, the potentiometer's resistance changes, so the control circuit can precisely regulate how much movement there is and in which direction.

When the shaft of the motor is at the desired position, power supplied to the motor is stopped. If not, the motor is turned in the appropriate direction. The desired position is sent via electrical pulses through the signal wire. The motor's speed is proportional to the difference between its actual position and desired position. So if the motor is near the desired position, it will turn slowly, otherwise it will turn fast. This is called proportional control. This means the motor will only run as hard as necessary to accomplish the task at hand, a very efficient little guy.

How is the servo controlled?

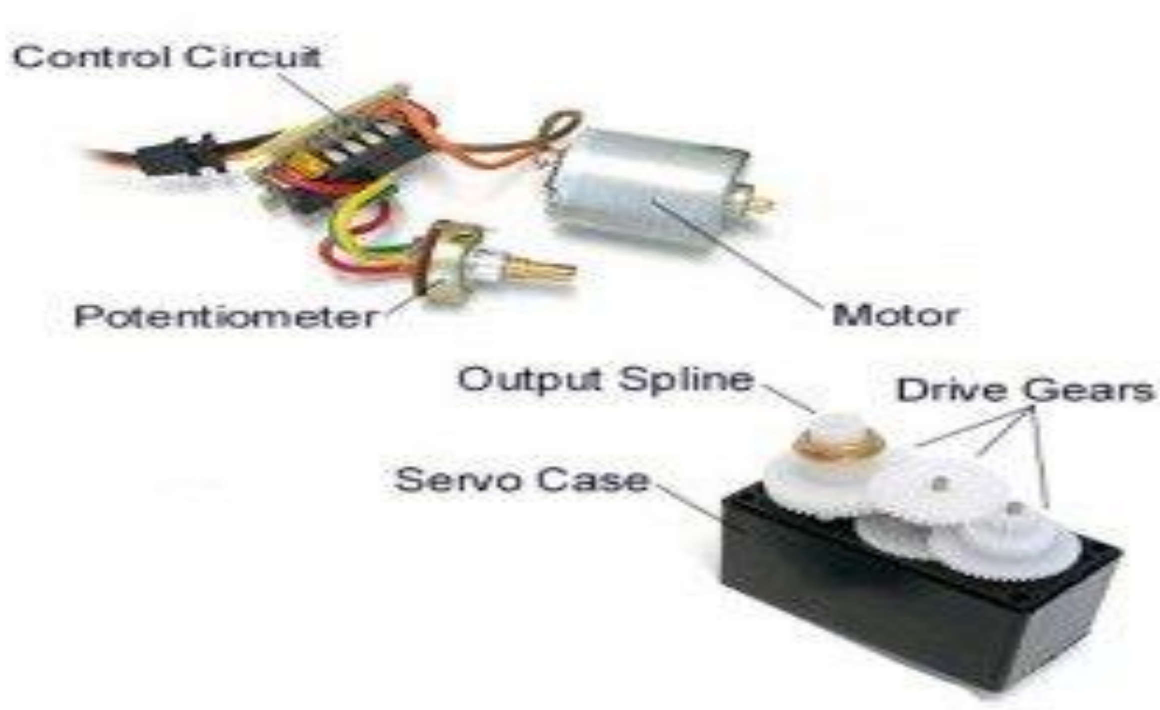


Figure 3 The guts of a servo motor (L) and an assembled servo (R)

Servos are controlled by sending an electrical pulse of variable width, or pulse width modulation (PWM), through the control wire. There is a minimum pulse, a maximum pulse, and a repetition rate. A servo motor can usually only turn 90° in either direction for a total of 180° movement. The motor's neutral position is defined as the position where the servo has the same amount of potential rotation in the both the clockwise or counter-clockwise direction. The PWM sent to the motor determines position of the shaft, and based on the duration of the pulse sent via the control wire; the rotor will turn to the desired position.

The servo motor expects to see a pulse every 20 milliseconds (ms) and the length of the pulse will determine how far the motor turns. For example, a 1.5ms pulse will make the motor turn to the 90° position. Shorter than 1.5ms moves it in the counter clockwise direction toward the 0° position, and any longer than 1.5ms will turn the servo in a clockwise direction toward the 180° position.

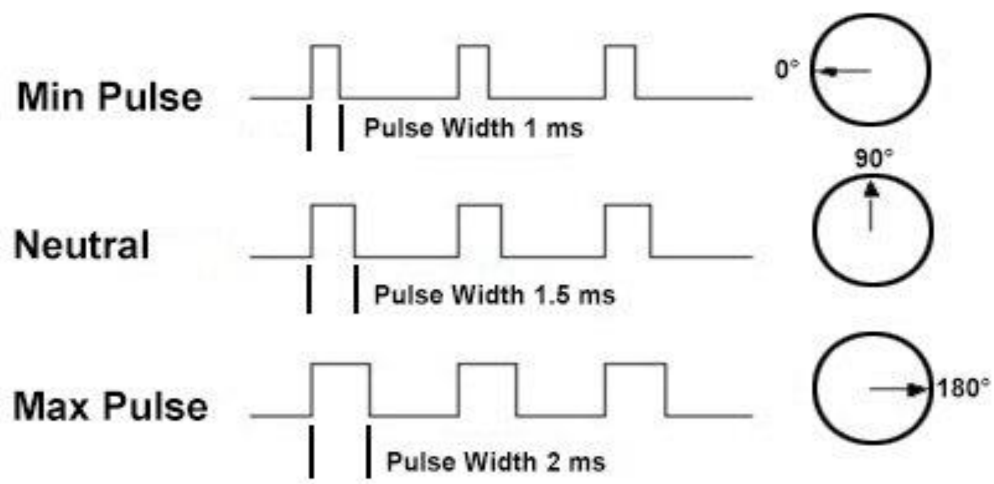


Figure 4 Variable Pulse Width Control Servo Position

When these servos are commanded to move, they will move to the position and hold that position. If an external force pushes against the servo while the servo is holding a position, the servo will resist from moving out of that position. The maximum amount of force the servo can exert is called the torque rating of the servo. Servos will not hold their position forever though; the position pulse must be repeated to instruct the servo to stay in position.

2.2 Resistive Touch Panel

A touch screen is a 2- dimensional sensing device that is constructed of 2 sheets of material separated slightly by spacers. A common construction is a sheet of glass providing a stable bottom layer and a sheet of Polyethylene (PET) as a flexible top layer.
The 2 sheets are coated with a resistive substance, usually a metal compound called Indium Tin Oxide (ITO). The ITO is thinly and uniformly sputtered onto both the glass and the PET layer. Tiny bumps called spacer dots are then added to the glass side, on top of the resistive ITO coating, to keep the PET film from sagging, causing an accidental or false touch. When the PET film is pressed down, the two resistive surfaces meet. The position of this meeting (a touch) can be read by a touch screen controller circuit.

Notice that some pins switch functions depending on if the controller is looking for a X-touch or a Y-touch position. The controller reads the X and Y position many times per second so the user may move his stylus (or finger) rapidly across the touch screen and the data will be captured. This provides smooth operation and allows drag-and-drop or signature capture.

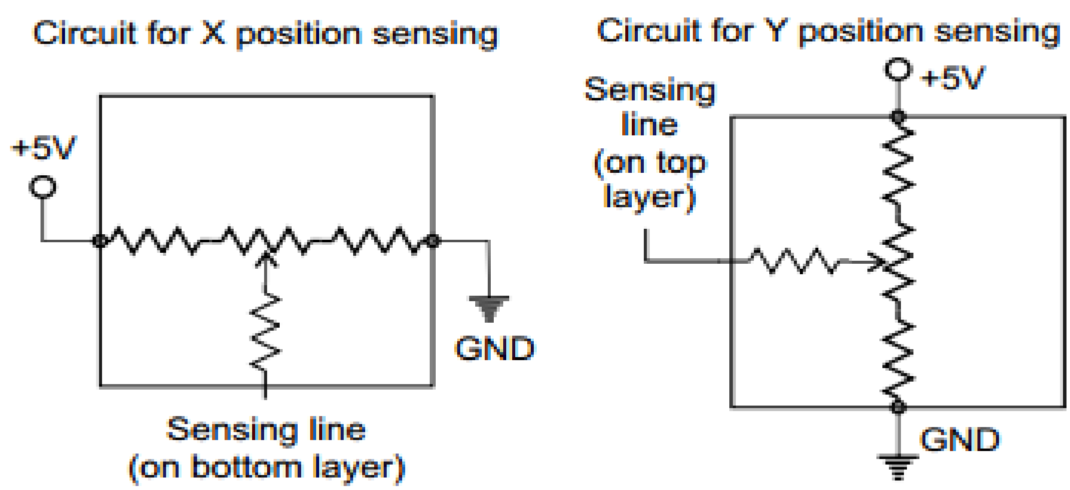


Figure 5 Sensing Mechanism

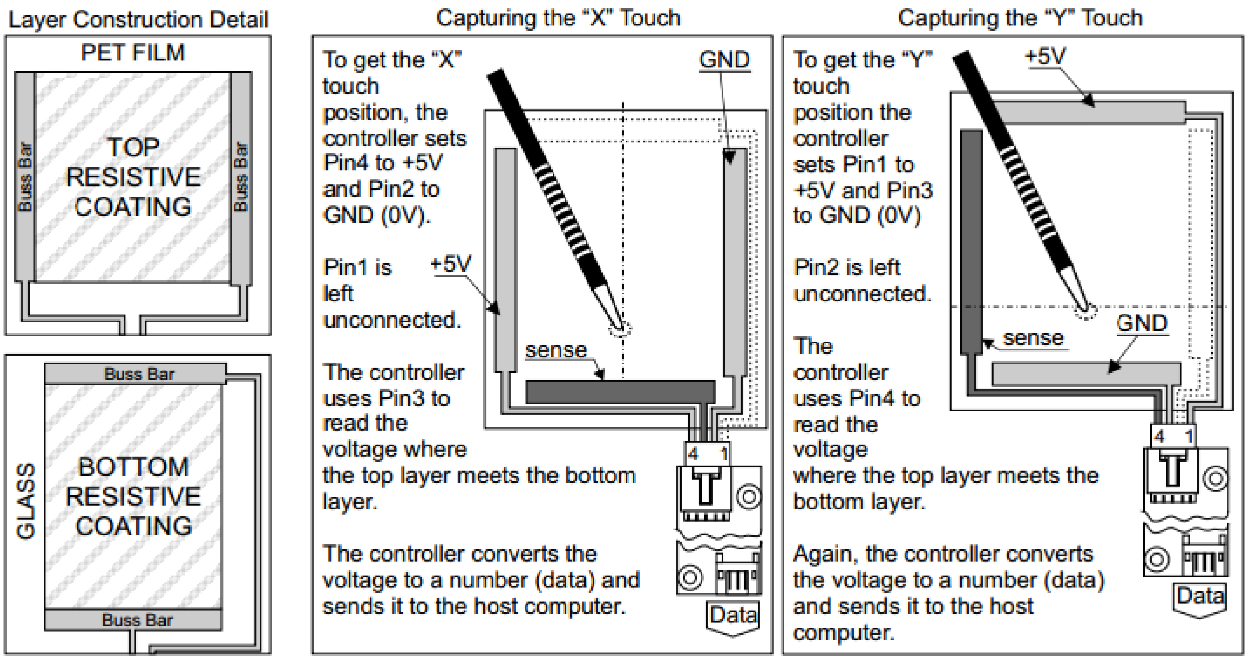


Figure 6 Mechanical Structure Components

2.3 Universal Joints

Universal joints (also called Hooke's joints) are used to allow torque (rotational power) to be transmitted through a varying angle. A typical use is in the prop shaft of a car or truck, where the torque from the engine is fed to the rear axle. The axle moves with the motion of the suspension, but the engine is fixed to the body, so a universal joint is used to accommodate the movement. UJs are also used for driveshaft's to non-steering wheels where the wheel motion is more limited, for example in an independent rear suspension design (IRS)[6].

Universal joints normally have to be used in pairs because one of their drawbacks is that they introduce a non-uniform change in rotational speed whenever the angle is not zero. A second UJ will cancel this effect.

Figure 7 Universal joint

2.4 Eyeball Bearing

It is a rod end bearing, also known as a heim joint (N. America) or rose joint (U.K. and elsewhere), is a mechanical articulating joint. Such joints are used on the ends of control rods, steering links, tie rods, or anywhere a precision articulating joint is required, and where a clevis end (which requires perfect 90 degree alignment between the attached shaft and the second component) is unsuitable. A ball swivel with an opening through which a bolt or other attaching hardware may pass is pressed into a circular casing with a threaded shaft attached. The threaded portion may be either male or female. The heim joint's advantage is that the ball insert permits the rod or bolt passing through it to be misaligned to a limited degree (an angle other than 90 degrees). A link terminated in two heim joints permits misalignment of their attached shafts (viz., other than 180 degrees) when used in tension. When used in compression, the through-rods are forced to the extreme ends of their ball's misalignment range, which cocks the link at an oblique angle.

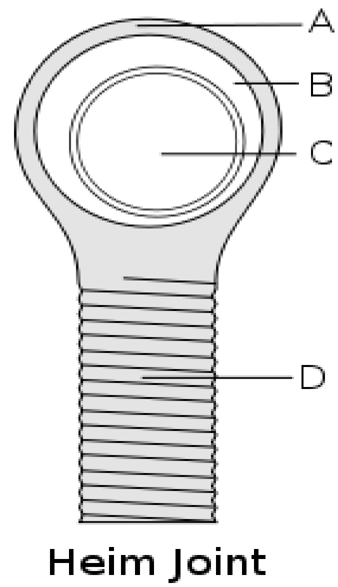


Figure 8 Eye Ball Bearing

2.5 Other components

- Centre rod
- L-clamp
- Servo arm
- Servo rod

2.6 Degrees of Freedom

In the proposed design the system requires two separate motors to be acting on the plate independently at the same time, hence making it a 2-DOF system. Avoiding conflict between two such motors was a major design problem. To overcome this problem an additional platform between the plate and the base of the system (ground) was introduced. Consider motor 'O1' as the base motor, fixed to the ground. Now the resistive touch panel is mounted on top of motor 'O2'and motor 'O1' at an angle of '90' degrees. Hence, motor 'O1' controls one axis while motor 'O2' controls second axis making 2-DOF system. This dependency is catered for by using separate closed loop feedback control systems for each axis/motor.

2.7 Actuation Mechanism

Each motor (O1 and O2) drives one axis of the plate-rotation angle and is connected to the plate by a spatial linkage mechanism. Referring to the schematic Each side of the spatial linkage mechanism (O1-P1-A-O and O2-P2-B-O) is a four-bar parallelogram linkage. This ensures that for small motions around the equilibrium, the plate angles (Q1 and Q2) are equal to the corresponding motor Angles (QM1 and QM2). The plate is connected to ground by means of a U-joint at O. Ball joints (at points P1, P2, A and B) connecting linkages and rods provide enough freedom of motion to ensure that the system does not bind.

The motors used for driving the linkage are simple brushed DC motors [3].

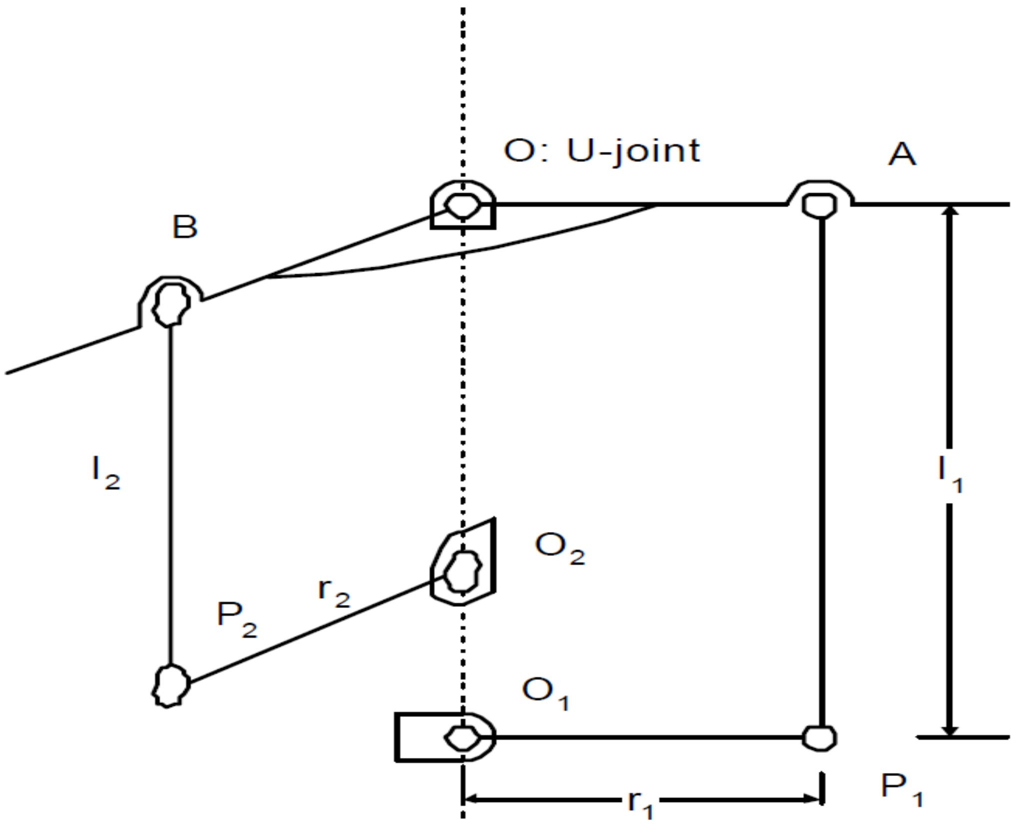


Figure 9 Plate Actuation Spatial Linkage Mechanism

2.8 Ball position sensor

A resistive touch sensitive glass screen that is actually meant to be a computer touchscreen was used for sensing the ball position. It provides an extremely reliable (less than 1% error), accurate (1024X1024 points across the screen), and economical solution to the ball position sensing problem. The screen consists of three layers: a glass sheet, a conductive coating on the glass sheet, and a hard-coated conductive toposheet. An external DC voltage is applied to the two edges of the glass layer. When the top layer is pressed by the weight the ball, the top sheet gets compressed into contact with the conductive coating on the glass layer. As a result, current is drawn from one side of the glass layer in proportion to the distance of the touch from the edge. This generates a voltage at the bottom layer of the glass-sheet. This voltage is filtered and subsequently used for computing the ball position coordinates (x_b and y_b) using simple linear relationships. The ball rolls on this touch-screen, which in turn is mounted on the acrylic plate.

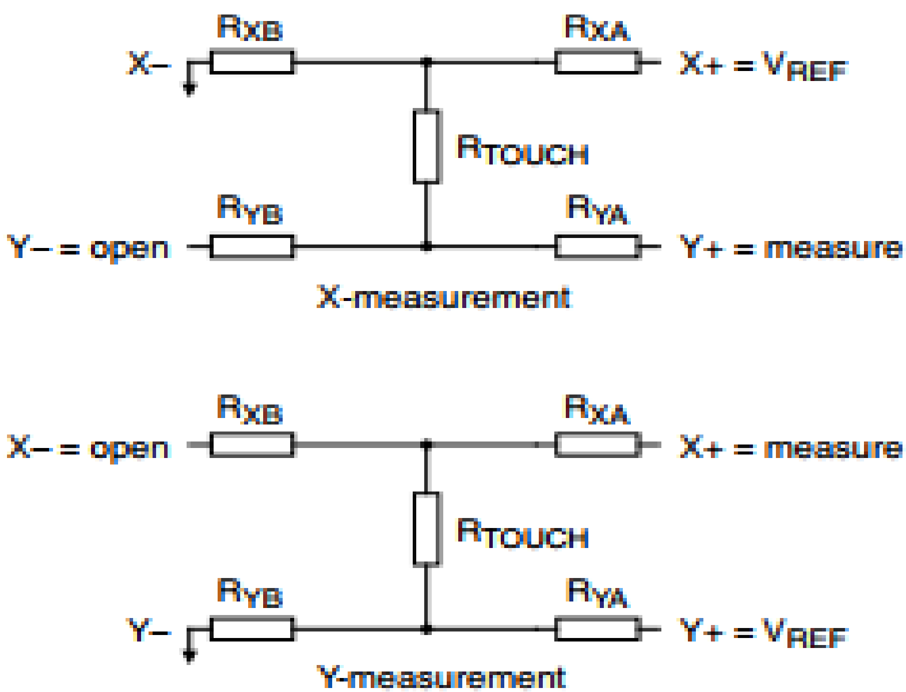


Figure 10 Touch Panel Voltage Calculation

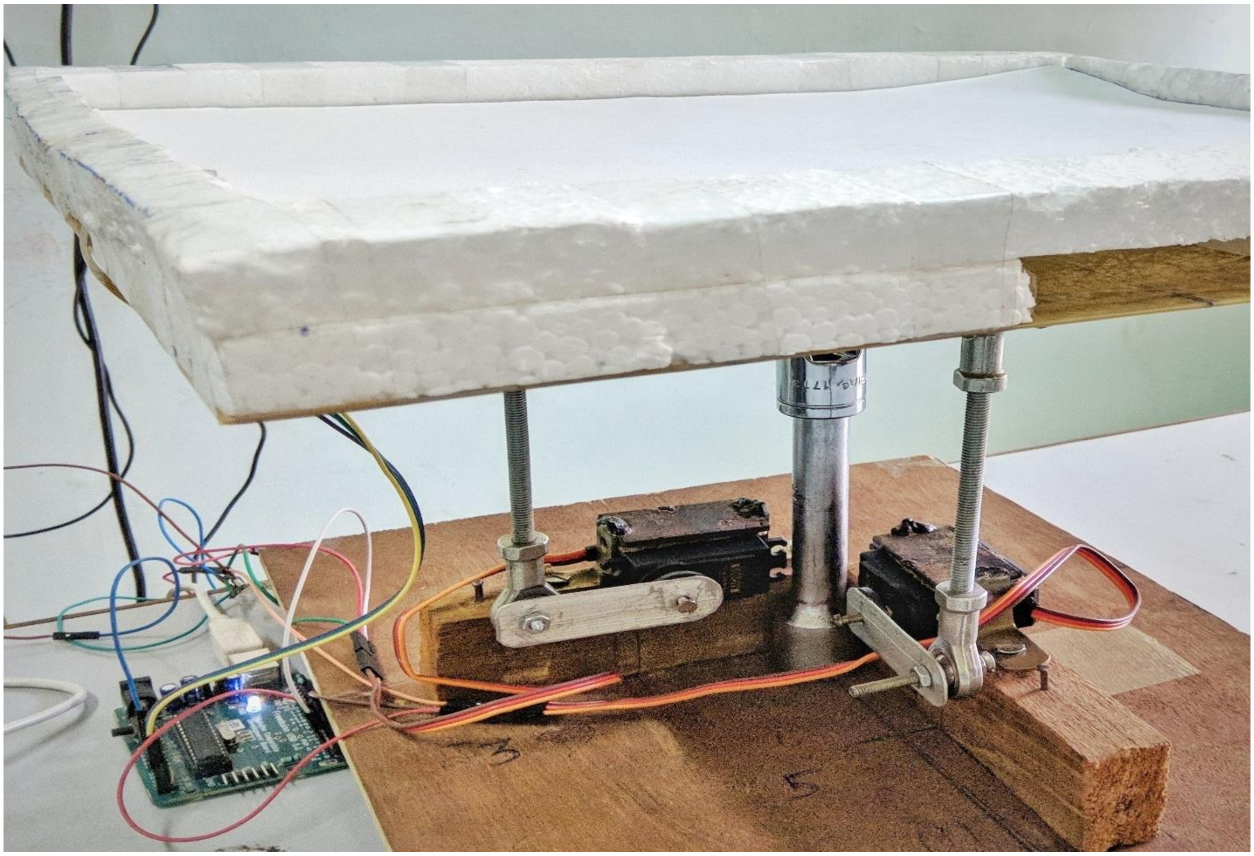


Figure 11 UPPER VIEW OF SYSTEM

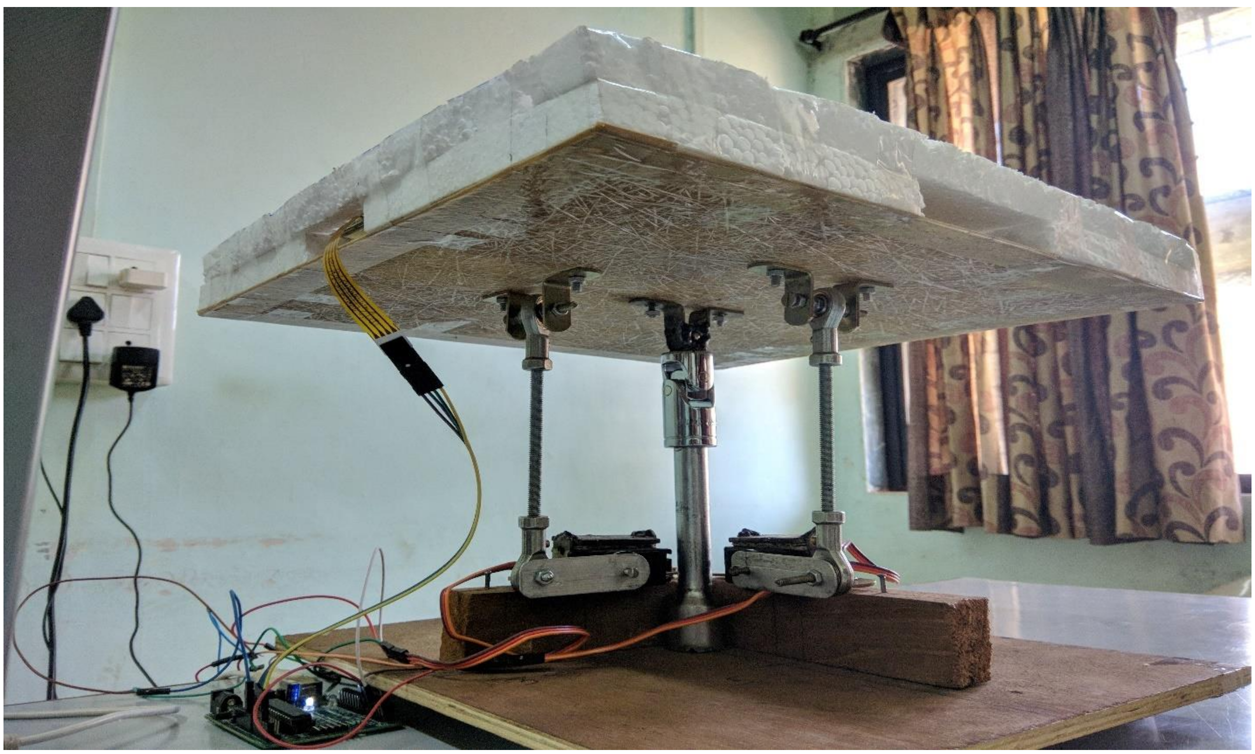


Figure 12 BOTTOM VIEW OF SYSTEM

2.9 Assumptions

The following assumptions are used in the modelling the above-described physical system:

1. It is assumed that the ball slides on plate with negligible friction.
2. The rotation of the ball about its vertical axis is assumed to be negligible.
3. Rolling friction between the ball and the plate is neglected.
4. It is assumed that there will be small motion of the plate about the equilibrium configuration. This ensures that the plate angles will be approximately equal to motor angles.
5. The plate is assumed to have mass-symmetry about its x-z and y-z planes. This ensures that there are no non-diagonal terms in the inertia matrix for the plate.

2.10 MATHEMATICAL THEORY

A physical model of the ball-on-plate system is provided in Figure, where x-y-z is the ground frame. The plate has two degrees of freedom and its orientation is defined by two angles (q1 and q2) that constitute a body (1-2) rotation. Frame x" -y"- z" is a plate fixed reference frame, while x'-y'-z' is an inter-mediate frame. All angles are defined to be positive in the CCW sense.

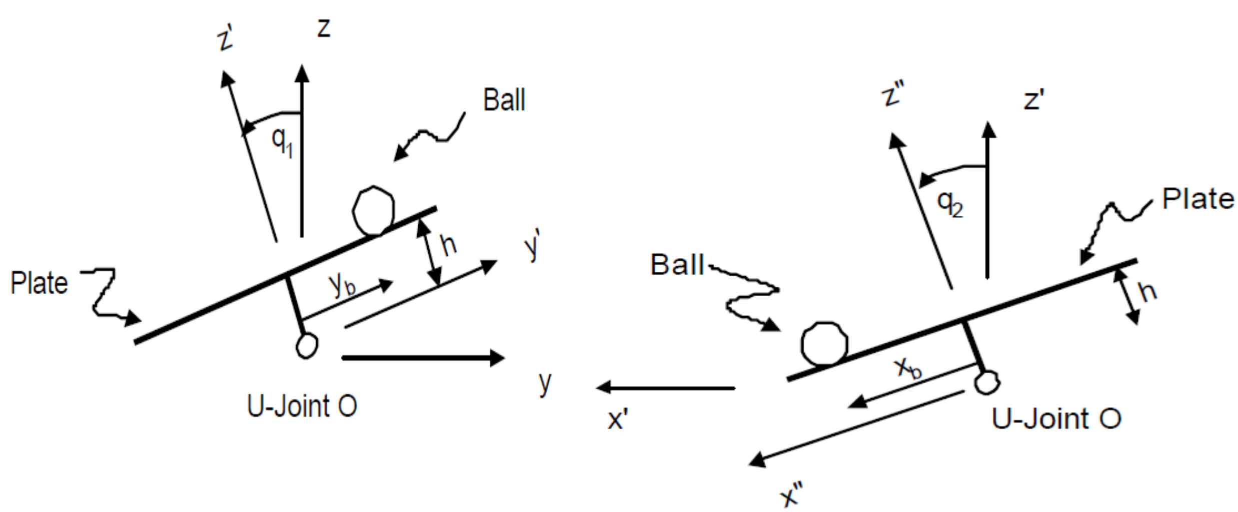


Figure 13 Physical Model of the Ball-on-Plate System

Chapter 3 **MATLAB SIMULATION**

3.1 Equational Approach

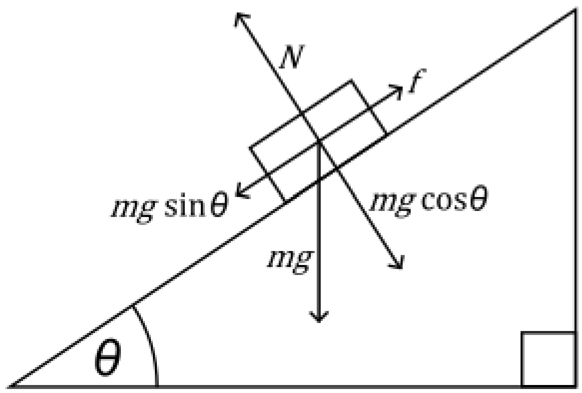


Figure 14 Subject on Inclined Plane

Angle of the inclined plane with the horizontal plane = θ

Acceleration down the inclined plane = g$\sin\theta$

Acceleration along the x direction = (Acceleration down the inclined plane) *$\cos\theta$

∴ Acceleration along the x direction a_x= g$\sin\theta$*$\cos\theta$

Position of the ball in x-direction = $\iint$ (Acceleration along the x direction)

∴ Position of the ball in x-direction = $\iint$ a_x

Similar equations of motion are applicable for motion along the z-direction.

Angle of the inclined plane with the horizontal plane = θ for x-direction

Angle of the inclined plane with the horizontal plane = $\emptyset$ for z-direction

Assuming the dimensions of the plate to be 20 units * 20 units and the centre of the plate as the origin,

$$\theta = \text{atan}(\frac{qy}{10})$$

Where qy = height of the point at the end of the plate along the positive x-direction. Similar equation can be obtained for ∅, using the height of the point at the end of the plate along the positive z-direction.

The y co-ordinate of the ball can be calculated using the equation of the plane and the x and z co-ordinates of the ball[4].

3.2 MATLAB MODEL:

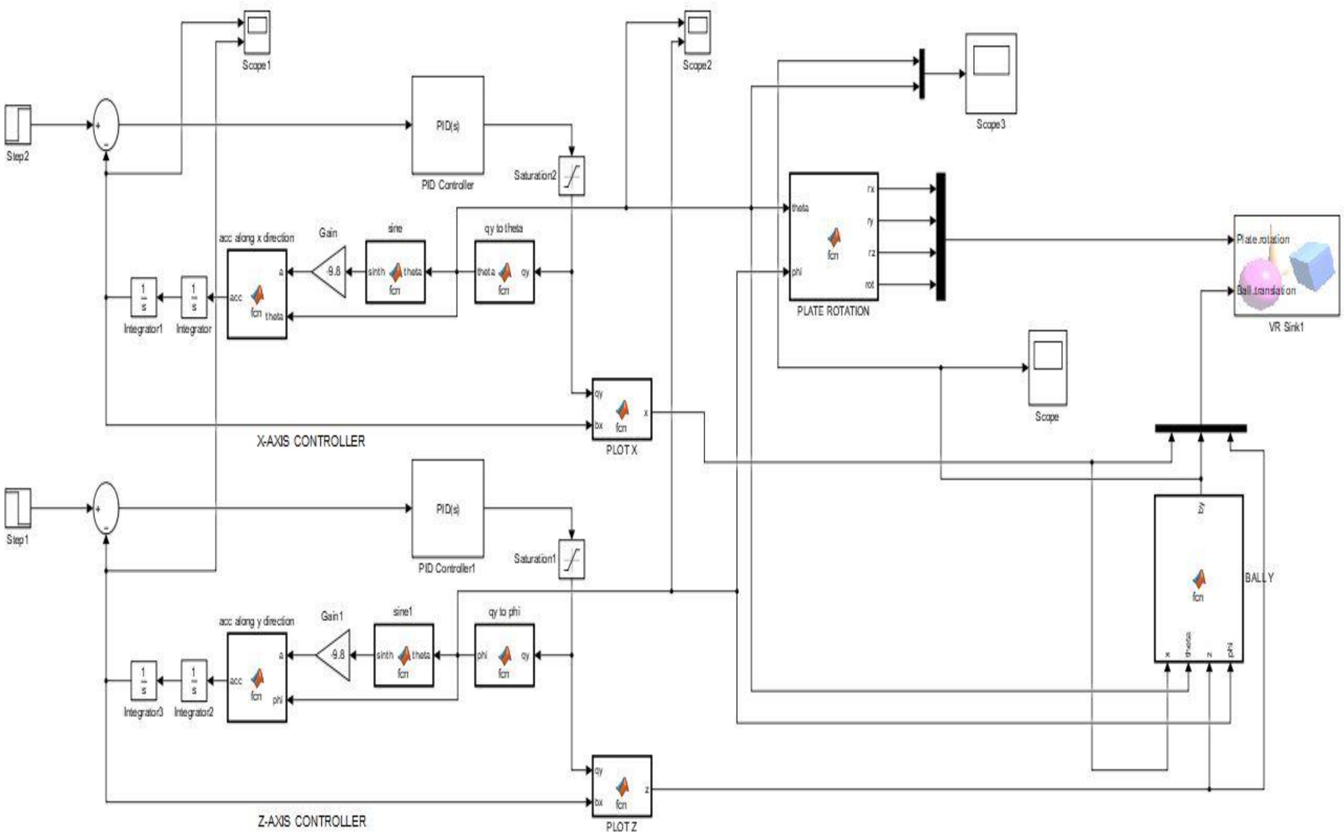


Figure 15 MATLAB SIMULATION MODEL

3.3 VRML SIMULATION

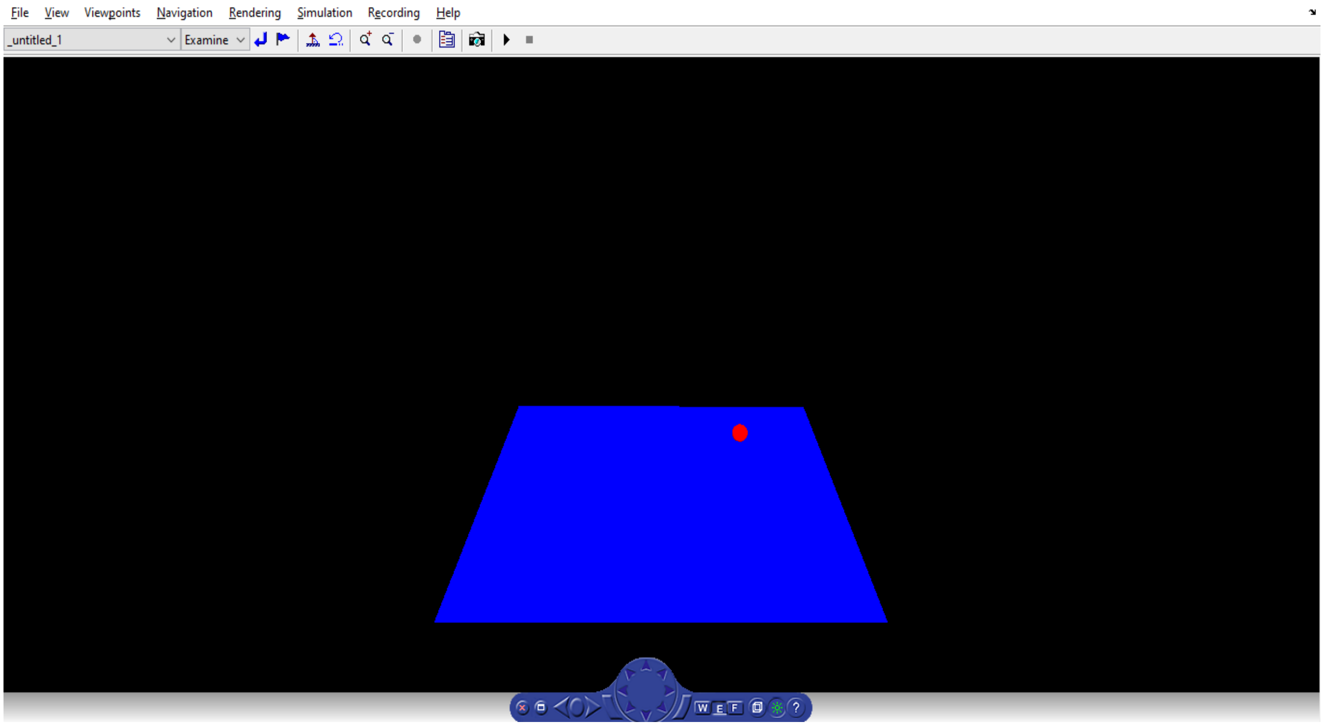


Figure 16 Ball at 5*5 Distant from Midpoint

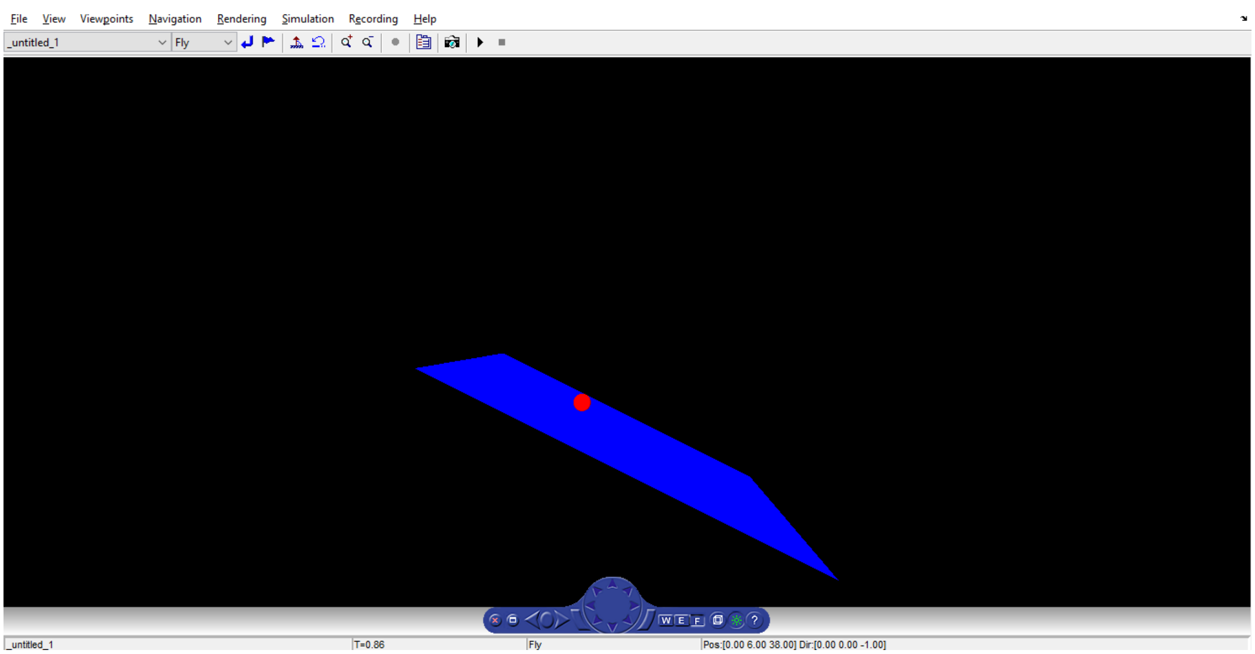


Figure 17 Motion of Ball

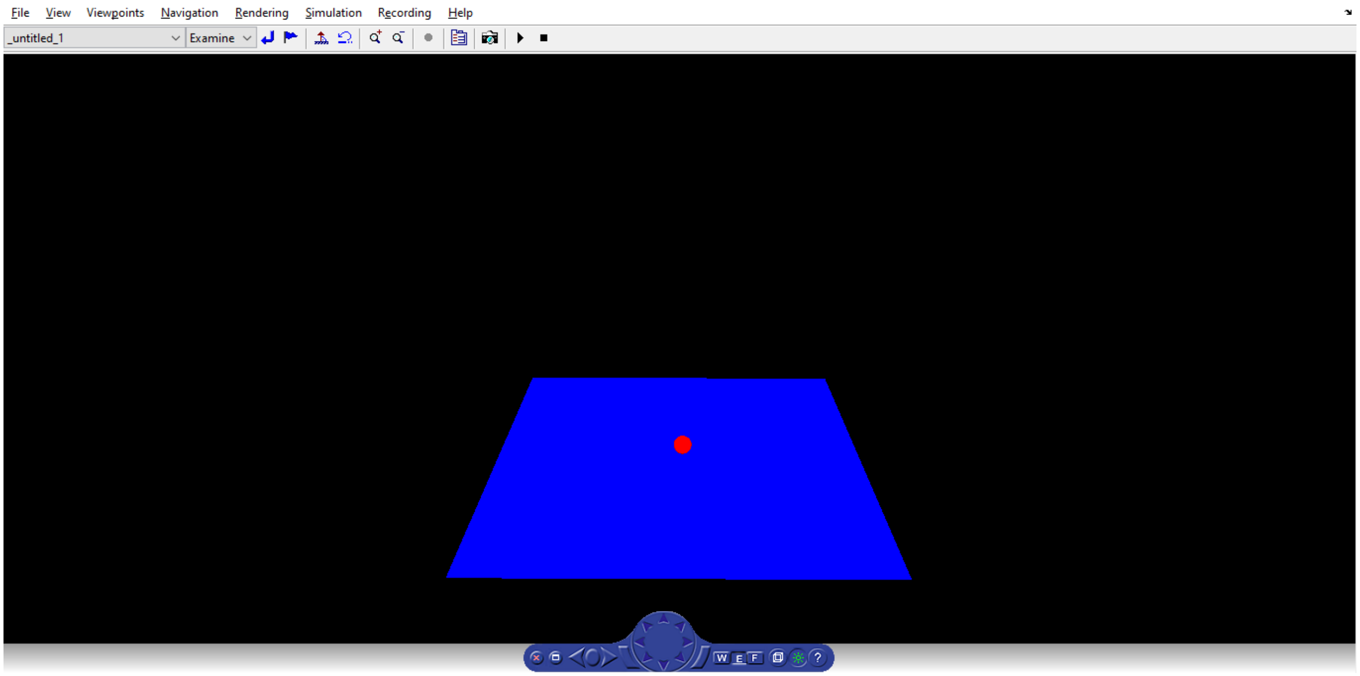


Figure 18 Stabilization of Ball is Complete

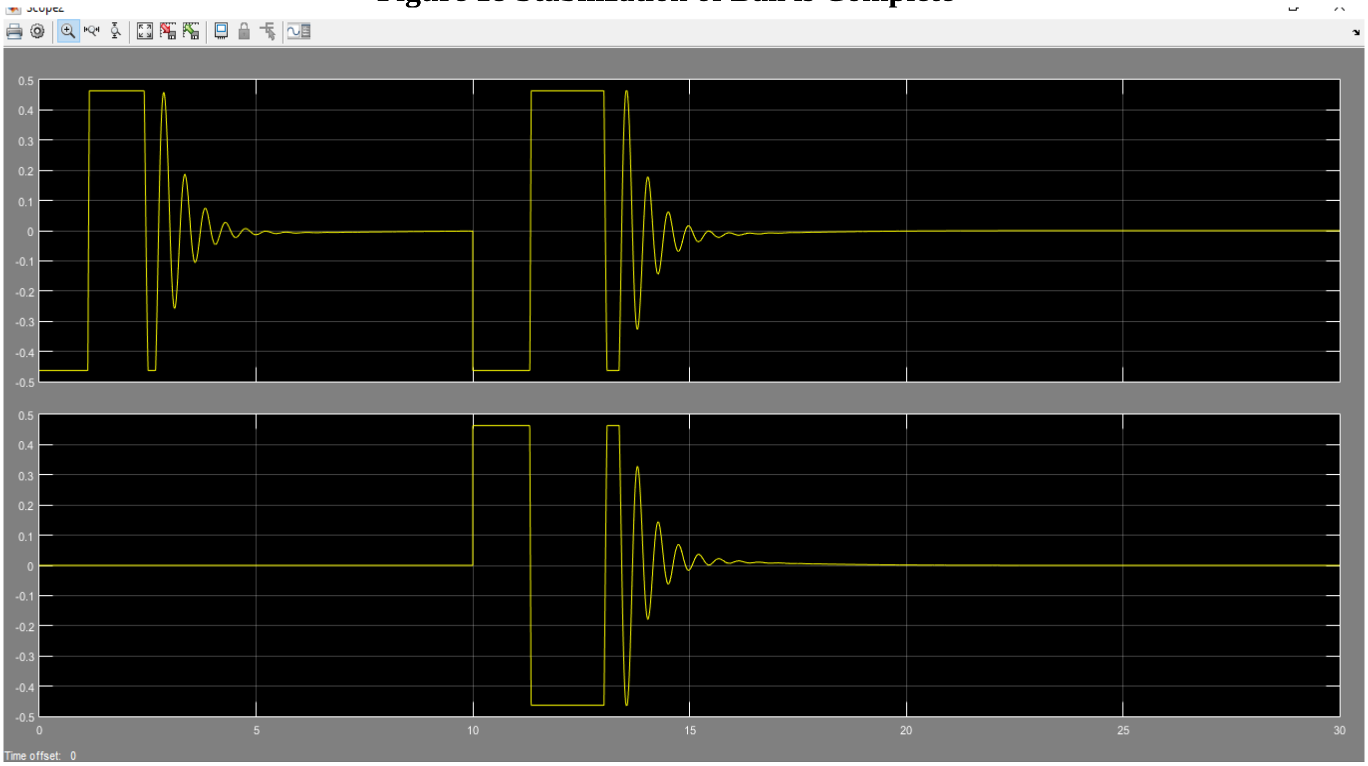


Figure 19 SIMULATION

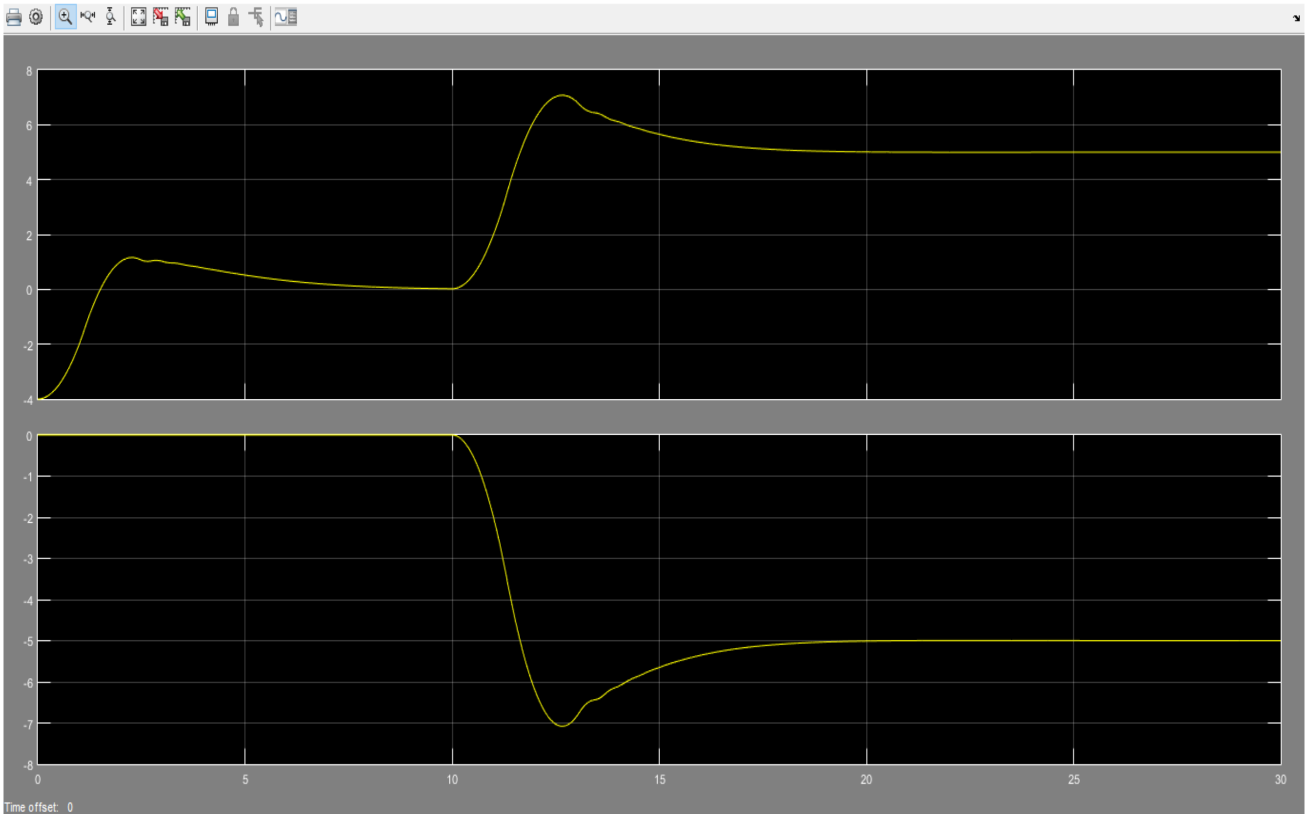


Figure 20 Scope Results

Figure 19 represents the Plane Rotation Angle of the Motors in Both X and Y Axis with reference to the Set Points.

Figure 20 Shows the Ball Positions in X and Y Axis while it is Stabilizing itself with reference to the Set Points.

Chapter 4 Control Algorithm

The Arduino board receives the coordinates through the touch panel and calculates both servo motor arm angles considering a PID control mechanism. Next it drives the servos with the required control PWM (Pulse Width Modulated) signal.

A. System flowchart and PUs relationship

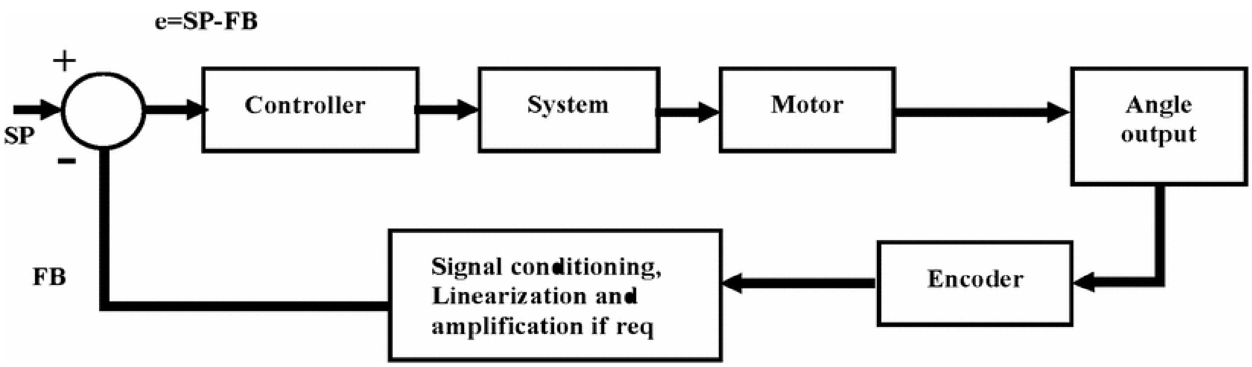


Figure 21 System control

The system utilizes two RC servos to tilt and control the plate. In order to drive the servos, the standard PWM pulses are used. A pulse of 1.5ms puts the servo arm at neutral position. The control over the servo arms are accomplished by increasing or decreasing of the PWM pulse length. The exact length of pulses are calculated by calling a function that receives the ball's coordinates as its arguments, and runs the PID control and outputs the calculated pulse length as its return values. Finally, the PWM pulses will be applied to drive the RC servo motors and tilt the plate. The steps to drive the servos are shown in the figure.

B. PID Control

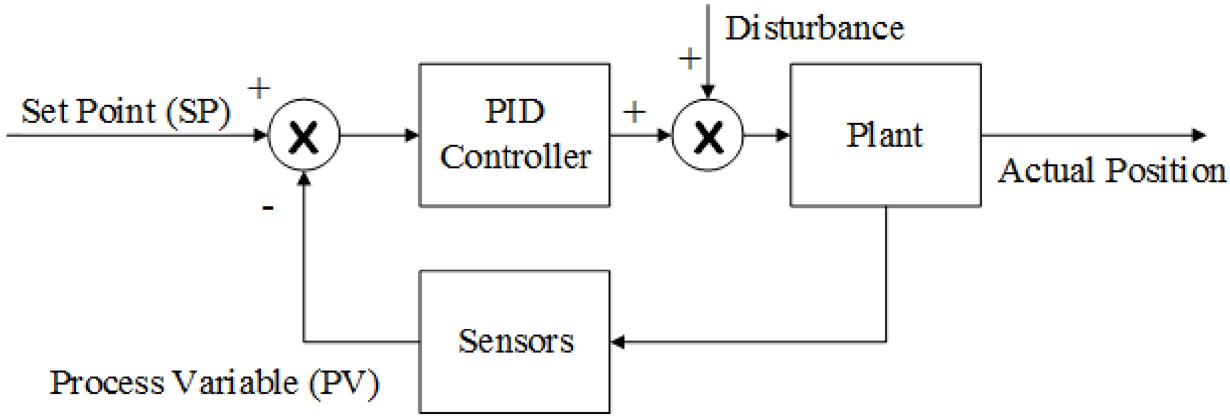


Figure 22 General PID Control Block Diagram

The general PID control flowchart is shown in the figure. We define the SP (Set Point) at coordinate (500, 500) which is the center of the plate. The disturbance is introduced by exerting a force on the ball alongside of an arbitrary direction. To implement the above controller in C programming language the diagram needs to be converted into its mathematical formula and then an algorithm needs to be written that can be fully implemented into machine code. Equation (1) shows the general PID control formula.

$$u(t) = Kp.e(t) + Ki\int_0^t e(\tau)d\tau + Kd\frac{d}{dt}e(t)$$

The above equation consists of the summation of three terms. Each term has a coefficient with the following definition:

Kp : Proportional coefficient that is proportional to the error.
Ki : Integral coefficient that accounts for past errors.
Kd : Derivative coefficient that predicts future errors.

The proportional term is just a simple multiplication that can be easily implemented into machine code using the C programming language.

The Integration term can be calculated using Euler integration technique by defining a small fixed time interval which is named dt.
The accumulation of this value in each run of the loop gives the desired integration result with good enough precision. For the derivative term only to keep track of the previous error is sufficient. The newly calculated error is subtracted from the previous error, then the result will be divided by dt to get the rate of change of the error in each iteration of PID control loop.

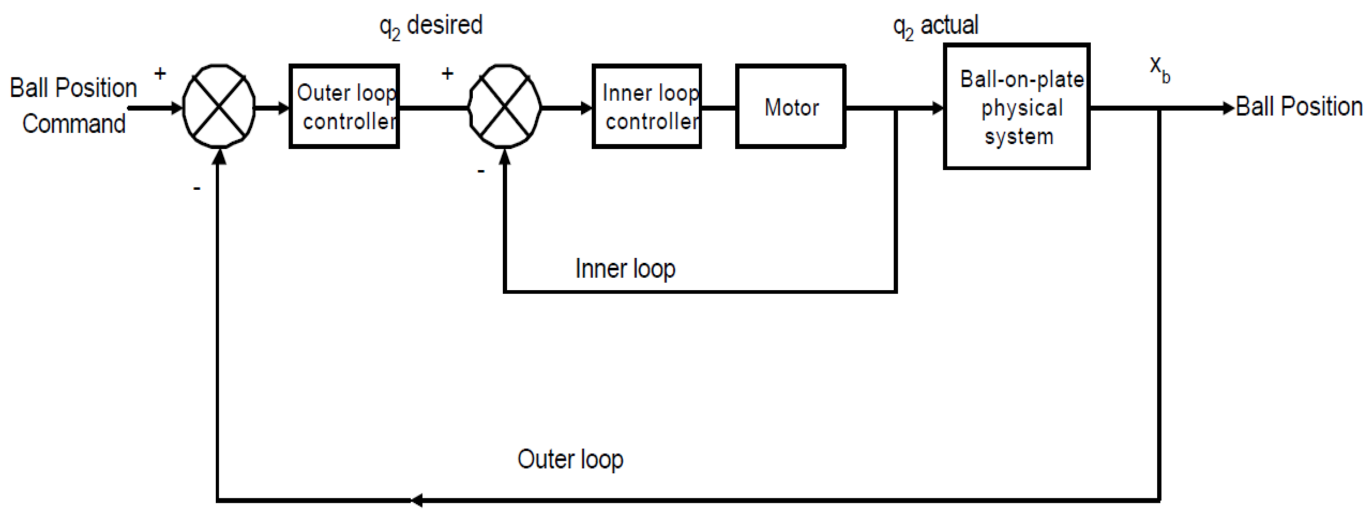


Figure 23 Control Scheme

The inner loop is then placed in an outer loop that controls the ball position. The next step in the control system design is to obtain a controller for the outer loop, based on the transfer function between ball position and the corresponding plate angle. The overall control scheme can be explained as follows. While the controller in the outer loop computes the angle by which the plate should move to balance the ball, the inner loop controller actually, moves the plate by that angle. Ideally, the inner loop should do this instantaneously, which is not possible in reality. Nevertheless, it is desirable to keep the speed of the inner loop much higher than that of the outer loop. This simple scheme is extremely effective in achieving the desired objective of balancing the ball.

Chapter 5 **Moving Average Filter**

The moving average filter is a simple Low Pass FIR (Finite Impulse Response) filter commonly used for smoothing an array of sampled data/signal. It takes M samples of input at a time and take the average of those M-samples and produces a single output point. It is a very simple LPF (Low Pass Filter) structure that comes handy for scientists and engineers to filter unwanted noisy component from the intended data.

As the filter length increases (the parameter M) the smoothness of the output increases, whereas the sharp transitions in the data are made increasingly blunt. This implies that this filter has excellent time domain response but a poor frequency response[5].

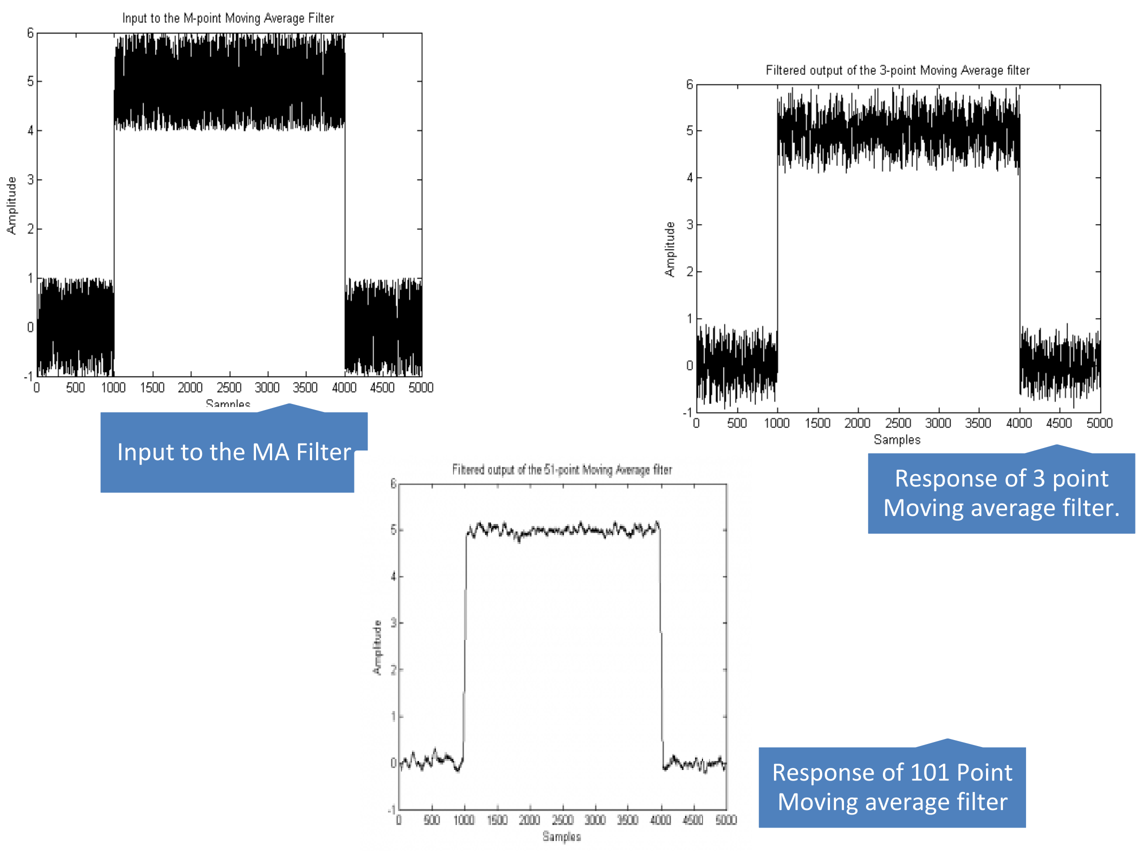


Figure 24 Moving Average

The MA filter perform three important functions:

1. It takes M input points, computes the average of those M-points and produces a single output point.
2. Due to the computation/calculations involved, the filter introduces a definite amount of delay.
3. The filter acts as a Low Pass Filter (with poor frequency domain response and a good time domain response).

The moving average of length N can be defined as

$$y[n] \equiv \frac{1}{N} \sum_{i=-N+1}^{0} x[n+i],$$

written as it is typically implemented, with the current output sample as the average of the *previous* N samples.

Chapter 6 Experiment and Results

The PID coefficients were manually determined through trial and error. First the K_i and K_d are set to zero and K_p is increased to reach the desired proportional gain. Then K_i is increased to eliminate the steady state error. Finally, K_d is gradually increased to reduce overshoot.

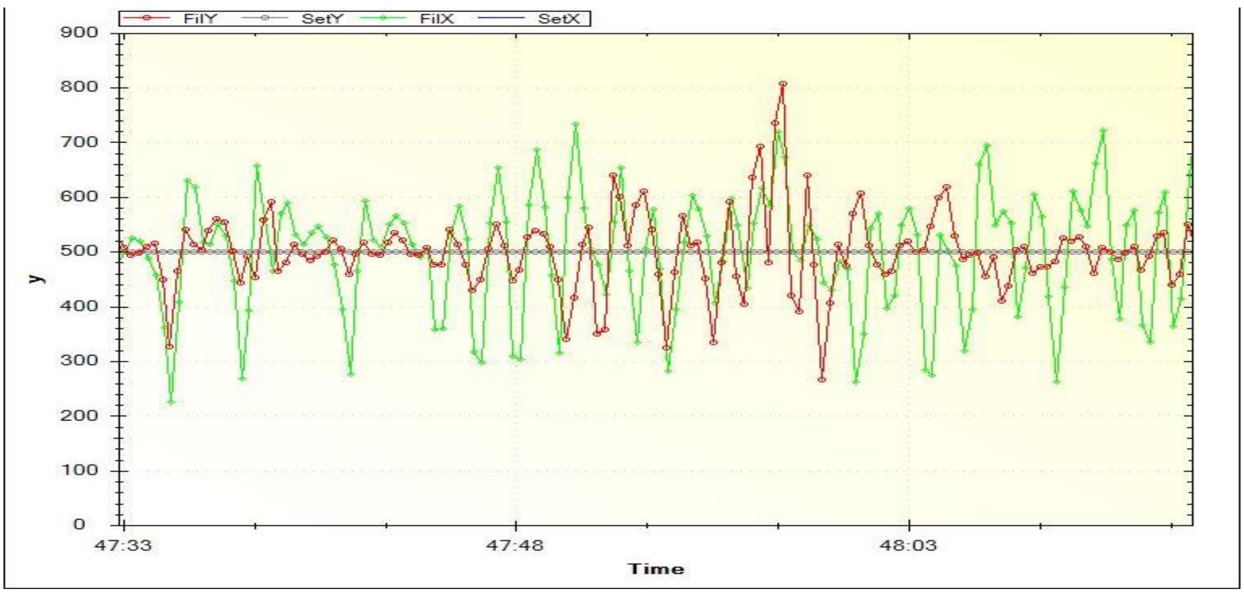


Figure 25 Ball Position without Tuned PID Control

The Graph Plot represents

- Filtered Value of X (Green) with Reference to Set Value of X (Blue).
- Filtered Value of Y (Red) with reference to Set Value of Y (Silver).

PID Tuned Ball Positions:

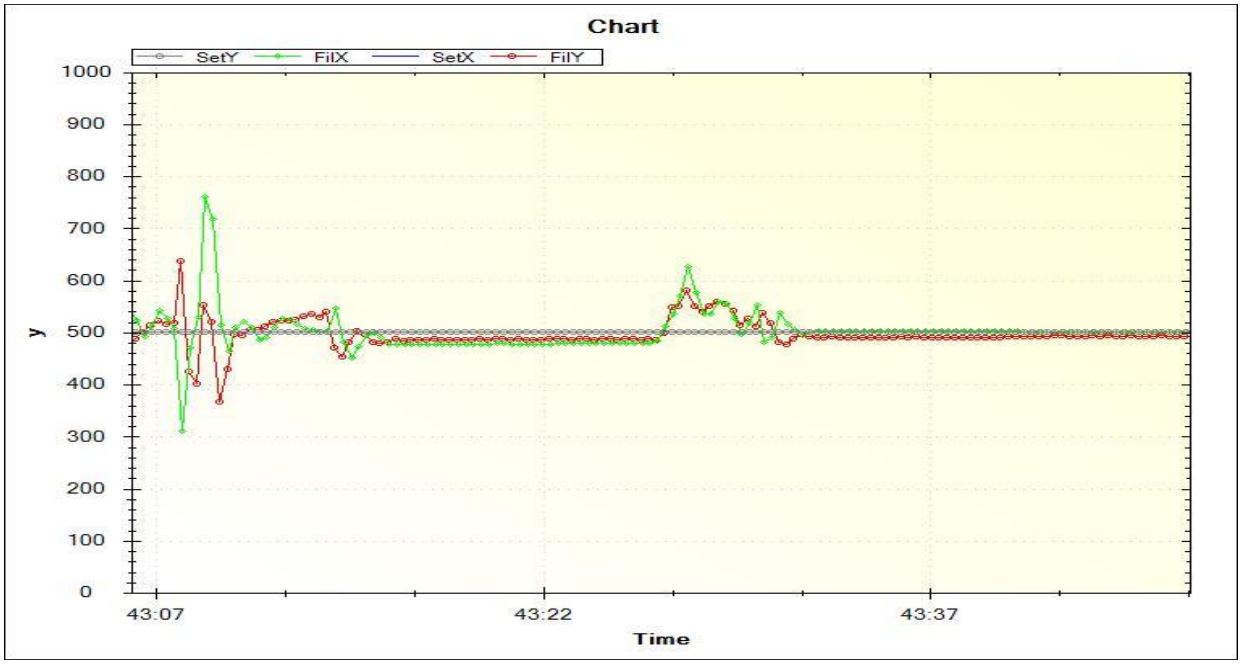


Figure 26 Ball Position with PID Control

PID Constants value

Kpx = 0.04, Kix = 0.018, Kdx = 0.018
Kpy =0.03, Kiy=0.022 , Kdy =0.018

The Graph Plot represents

- Filtered Value of X (Green) with Reference to Set Value of X (Blue).
- Filtered Value of Y (Red) with reference to Set Value of Y (Silver).

The figure above refers to x and y coordinates of ball position. At first the ball is unstable with respect to the Reference to the Set value.
As time progresses the system stabilizes the ball around the plate's center.

Chapter 7 **Applications**

7.1 Speed Cruiser Simulator

The Stewart platform design is extensively used in flight simulation, particularly in the so-called full flight simulator for which all 6 degrees of freedom are required. A flight simulator is a device that artificially re-creates aircraft flight and the environment in which it flies, for pilot training, design, or other purposes. It includes replicating the equations that govern how aircraft fly, how they react to applications of flight controls, the effects of other aircraft systems, and how the aircraft reacts to external factors such as air density, turbulence, wind shear, cloud, precipitation, etc. [2].

Figure 27 SC07 Speed Cruiser Simulator

7.2 Robocrane

The Robocrane is a kind of manipulator resembling a Stewart platform but using an octahedral assembly of cables instead of struts. Like the Stewart platform, the Robocrane has six degrees of freedom (x, y, z, pitch, roll, & yaw).

It was developed by Dr. James S. Albus of the US National Institute of Standards and Technology (NIST), using the Real-Time Control System which is a hierarchical control system. Given its unusual ability to "fly" tools around a work site, it has many possible applications, including stone carving, ship building, bridge construction, inspection, pipe or beam fitting and welding. Albus invented and developed a new generation of robot cranes based on six cables and six winches configured as a Stewart platform. The NIST RoboCrane™ has the capacity to lift and precisely manipulate heavy loads over large volumes with fine control in all six degrees of freedom.

Figure 28 Robocrane

7.3 NASA Docking System

The NASA Docking System (NDS) is a spacecraft docking and berthing mechanism being developed for future US human spaceflight vehicles, such as the Orion Multi-Purpose Crew Vehicle and the Commercial Crew vehicles. The NDS docking mechanism is androgynous, the first system to use low impact technology and the first system to allow both docking and berthing. It supports both autonomous and piloted dockings and features pyrotechnics for contingency undocking. Once mated the NDS interface can transfer power, data, commands, air, communication and in future implementations will be able to transfer water, fuel, oxidizer and pressurant as well. The passage for crew and cargo transfer has a diameter of 800 millimeters [1].

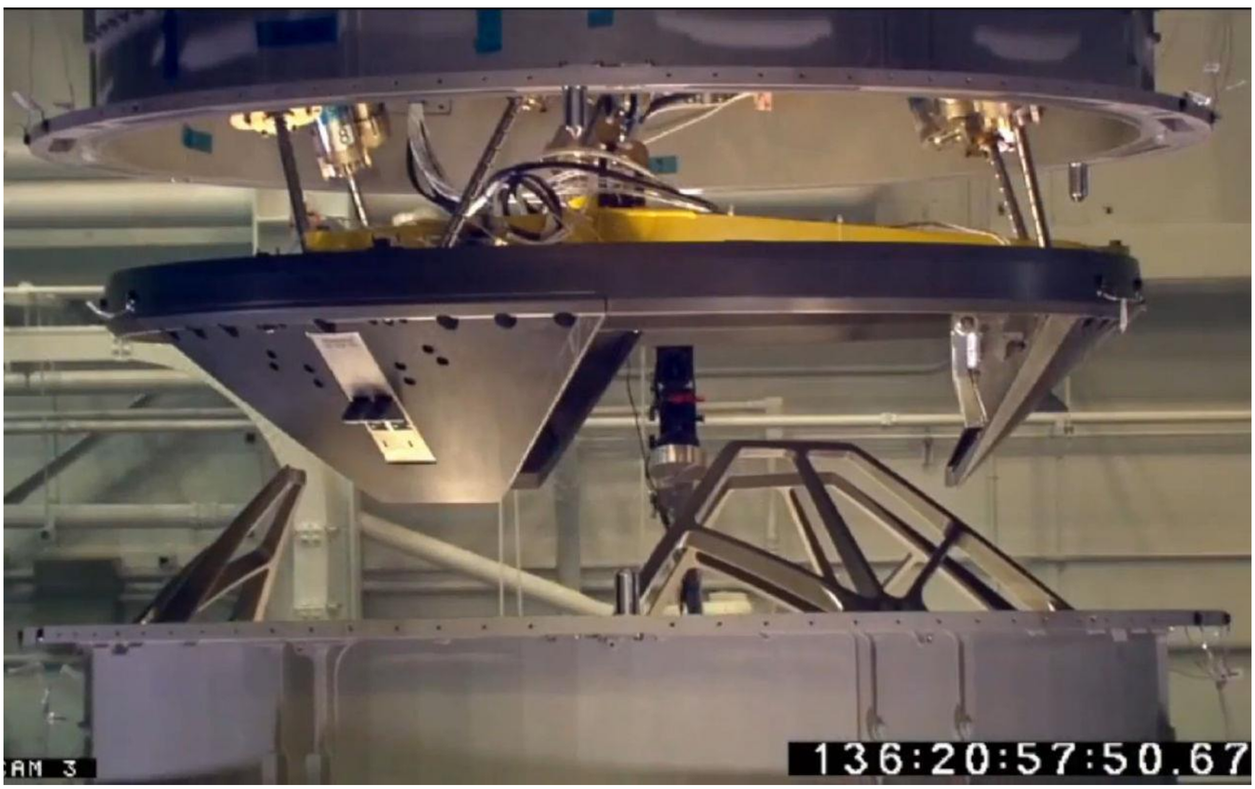


Figure 29 NASA Docking System

7.4 AMiBA

Array for Microwave Background Anisotropy (AMiBA), is a radio telescope designed to observe the cosmic microwave background and the Sunyaev-Zel'dovich effect in clusters of galaxies. AMiBA was initially configured as a 7-element interferometer, using 0.576 m Cassegrain dishes mounted on a 6 m carbon fiber hexapod mount. It is located on Mauna Loa, Hawaii, and observes at 3 mm (86–102 GHz) to minimize foreground emission from other, non-thermal sources. The telescope has a retractable shelter, made from seven steel trusses and PVC fabric.

Figure 30 AMiBA

Chapter 8 Conclusion

The control mechanism used here is a free-model approach based on PID control. The PID tuning is done manually by applying Trial and Error method. The experiment was conducted by dropping a ball on the plate and waiting for the ball equilibrium around the center of the plate. A disturbance is introduced after having the ball stabilized and the motion of servo arms and ball coordinates were recorded and plotted.

Despite being based on a linearized model, the controllers performed extremely well with the nonlinear system. When the system is in operation, the ball can be commanded to stay balanced at any point on the plate. It can also be directed to move from one point to another and stay there. The system once built and tested can be further used as an excellent test-bed for testing various other control schemes. An optimal controller using full-state feedback can be designed to achieve yet better performance. Although controllers based on the linear model perform extremely well, it will be interesting to apply the principles of non-linear controls and seek any further improvements in the system performance.

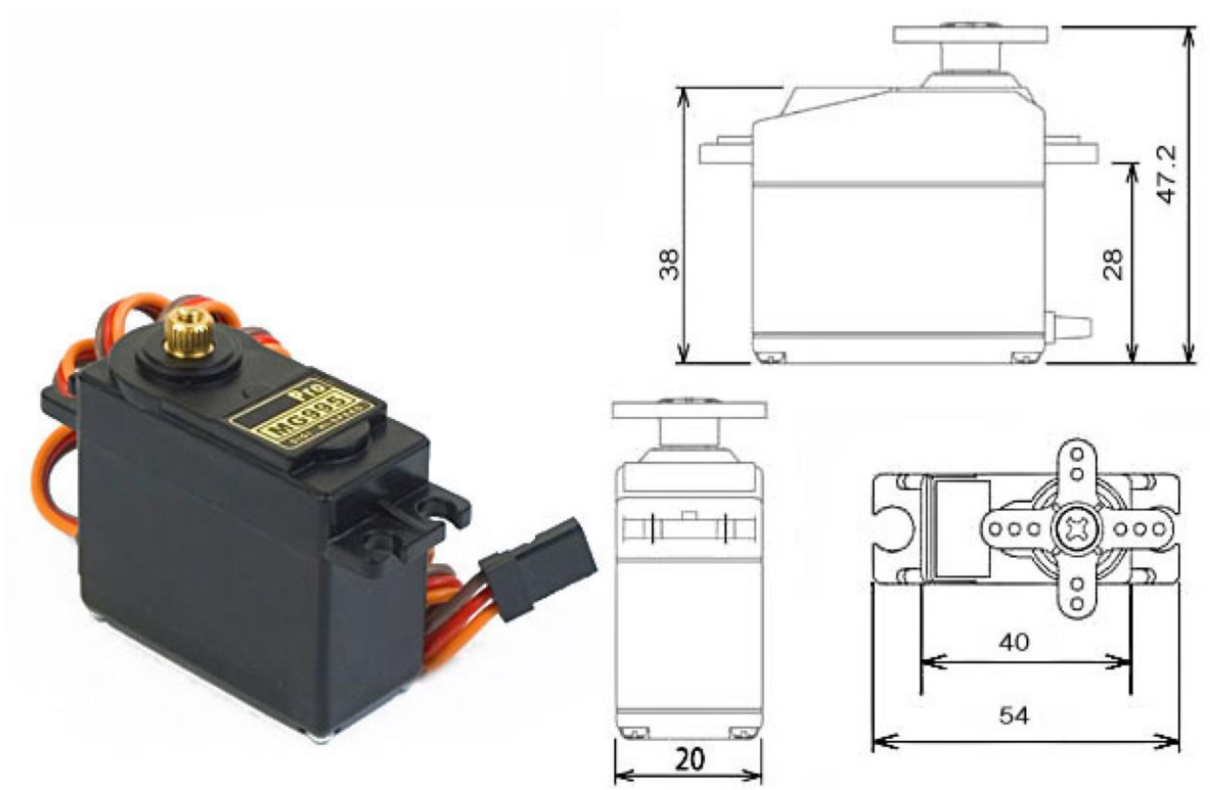
An extension of this objective is movement to specified positions on the system: given a position, a trajectory is plotted and the ball is moved to the new location. If this is extended further, trajectories such as a circle or a figure-eight can be followed given judicious choices for ball position requests.

REFERENCES (as per IEEE format)

1. *Parma, George (2011-05-20).* "Overview of the NASA Docking System and the International Docking System Standard" *(PDF). NASA. Archived from* the original *(PDF) on 15 October 2011*. Retrieved 11 April 2012.
2. http://www.flightglobal.com/pdfarchive/view/1962/1962%20-%201616.html
3. Stewart, D. (1965–1966). "*A Platform with Six Degrees of Freedom*". *Proc. Institution of Mechanical Engineers (UK)*. 180 (Pt 1, No 15).
4. Keeling, Christopher J. "*Modeling a Ball and Beam System Driven* by an Electric Motor" 23 Nov. 2010.
5. Graham Goodwin, Stefan Graebe, and Mario Salgado. *Control system design - ball-on-plate tutorial*. Available: http://csd.newcastle.edu.au/control/simulations/ball sim.html, 2001
6. S. Awtar, C. Bernard, N. Boklund, A. Master, D. Ueda, and K. Craig. "*Mechatronic design of a ball-onplate balancing system"*. Technical report, Rensselaer Polytechnic Institute, 2002.

APPENDIX A

A.1 MG995 High Speed Metal Gear Dual Ball Bearing Servo

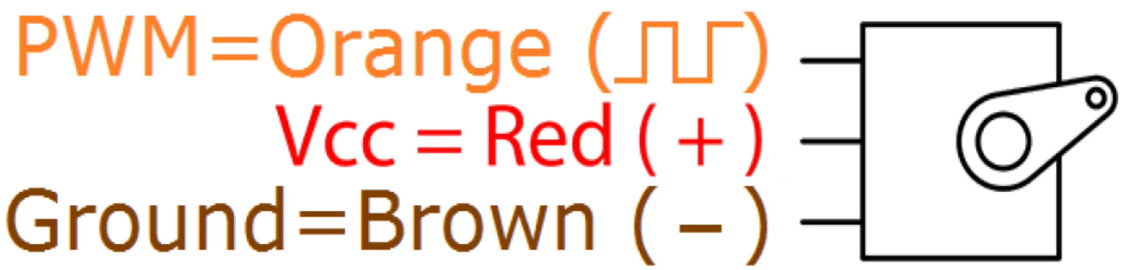


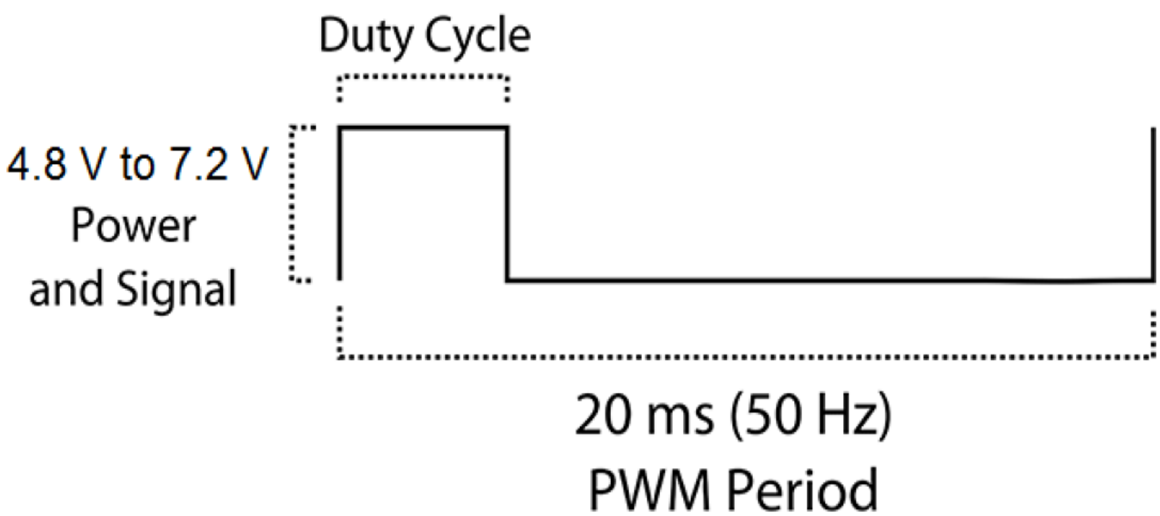
The unit comes complete with 30cm wire and 3 pin 'S' type female header connector that fits most receivers, including Futaba, JR, GWS, Cirrus, Blue Bird, Blue Arrow, Corona, Berg, Spektrum and Hitec. This high-speed standard servo can rotate approximately 120 degrees (60 in each direction). You can use any servo code, hardware or library to control these servos, so it's great for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. The MG995 Metal Gear Servo also comes with a selection of arms and hardware to get you set up nice and fast!

Specifications

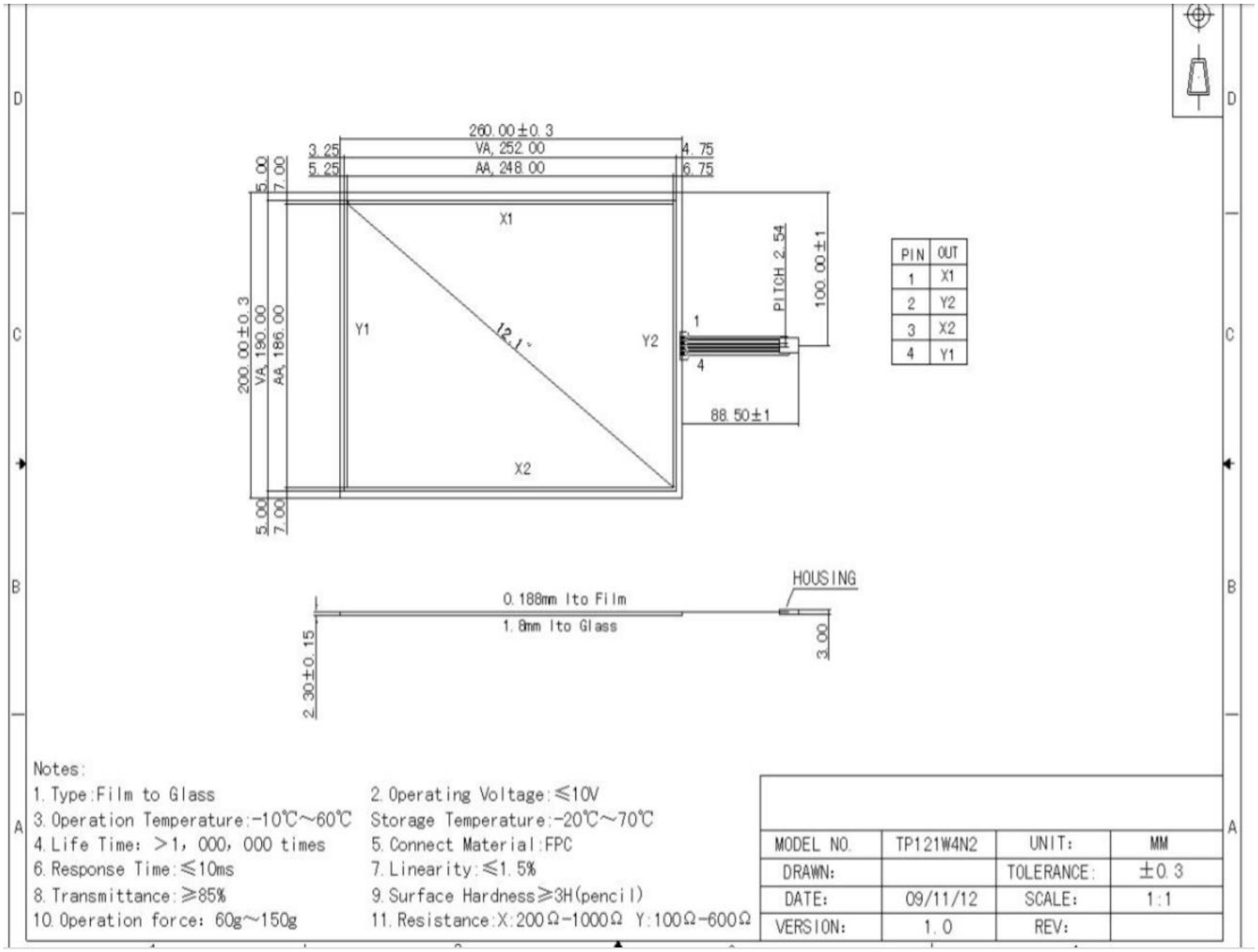
- Weight: 55 g
- Dimension: 40.7 x 19.7 x 42.9 mm approx.

□ Stall torque: 8.5 kgf·cm (4.8 V), 10 kgf·cm (6 V)
□ Operating speed: 0.2 s/60º (4.8 V), 0.16 s/60º (6 V)
□ Operating voltage: 4.8 V a 7.2 V
□ Dead band width: 5 µs
□ Stable and shock proof double ball bearing design
□ Temperature range: 0 ºC – 55 ºC





A.2 TOUCH PANEL



Operational Temperature -10ºC~60ºC

Response Time <=10ms

Transmittance: >=86%

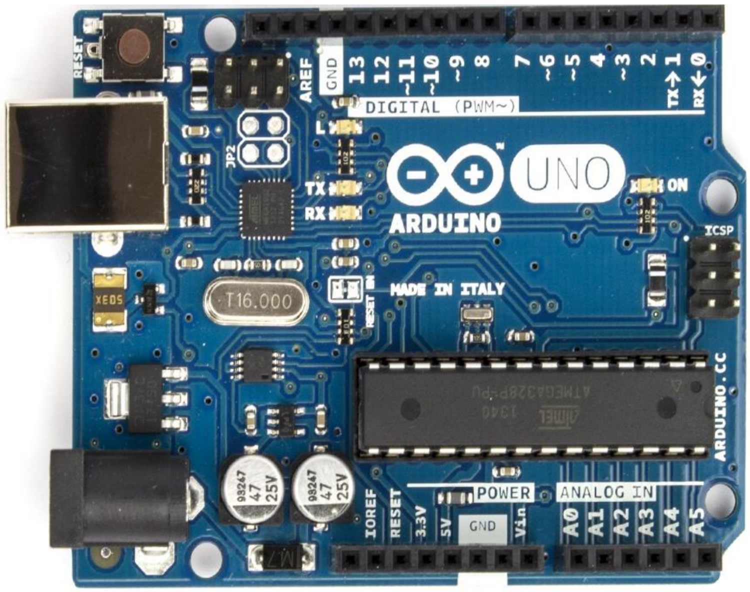
Operational Force: 60 grams~150 grams

Operational Voltage: <=10V

Resistance X: 200ohms~1000ohms

Y: 100ohms ~600ohms

A.3. Arduino Uno



The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 Analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

- VIN. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- 5V. This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator and can damage your board. We don't advise it.
- 3V3. A 3.3-volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- GND. Ground pins.

Specifications:

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328)
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

APPENDIX B

CODE USED:

```
#include <Servo.h>
#include <PID_v1.h>
#include "MegunoLink.h"
#include "RunningAverage.h"

//Touch panel filter (Running Average or Moving Average filter)
RunningAverage myRAX(6);
RunningAverage myRAY(6);

int plotCounter = 0;

int Xnew, Ynew;
int xServoPin = 3;
int yServoPin = 5;

//Reference positions or desired positions
double xRef = 500;
double yRef = 500;

//Parameters of PID controller
```

```
double kpx = 0.04;
double kix = 0.018;
double kdx = 0.018;

double kpy = 0.03;
double kiy = 0.022;
double kdy = 0.018;

double xInput, yInput;
double xOutput, yOutput;

//Sampling time*******
int Ts = 50;

XYPlot MyPlot;
TimePlot MyYTimePlot;
TimePlot MyXTimePlot;

PID xPID(&xInput, &xOutput, &xRef, kpx, kix, kdx, DIRECT);
PID yPID(&yInput, &yOutput, &yRef, kpy, kiy, kdy, DIRECT);

Servo xServo, yServo;
```

```
void setup() {

  xServo.attach(xServoPin);
  yServo.attach(yServoPin);

  xOutput = 90;
  yOutput = 90;

  Serial.begin(9600);

  xPID.SetMode(AUTOMATIC);
  xPID.SetOutputLimits(60, 120);
  xPID.SetSampleTime(Ts);

  yPID.SetMode(AUTOMATIC);
  yPID.SetOutputLimits(60, 120);
  yPID.SetSampleTime(Ts);

  xServo.write(xOutput);
  yServo.write(yOutput);
  delay(5000);
}
```

```
void loop() {

  Xnew = getX();
  myRAX.addValue(Xnew);
  xInput = myRAX.getAverage();
  xPID.Compute();
  xServo.write(180 - xOutput);

  Ynew = getY();
  myRAY.addValue(Ynew);
  yInput = myRAY.getAverage();
  yPID.Compute();
  yServo.write(180 - yOutput);

  //Plot values
  if(plotCounter == 100){
   plotCounter = 0;
   //MyPlot.SendData("SetXY", xRef, yRef);
   MyPlot.SendData("Filter", xInput, yInput);
   MyPlot.SendData("Original", Xnew, Ynew);
   //MyTimePlot.SendData(F("Ori"), Ynew);
   //MyYTimePlot.SendData(F("FilY"), yInput);
   //MyYTimePlot.SendData(F("SetY"), yRef);
```

```
    //MyXTimePlot.SendData(F("FilX"), xInput);
    //MyXTimePlot.SendData(F("SetX"), xRef);
   }
   plotCounter++;
}

int getX(){

  int X;

  pinMode(A0, OUTPUT);
  pinMode(A2, OUTPUT);
  pinMode(A3, INPUT);
  pinMode(A1, INPUT);

  digitalWrite(A0, HIGH);
  digitalWrite(A2, LOW);
  delay(1);

  X = analogRead(A3);
  delay(1);

  return X;
```

```
}

int getY(){

  int Y;

  pinMode(A0, INPUT);
  pinMode(A2, INPUT);
  pinMode(A3, OUTPUT);
  pinMode(A1, OUTPUT);

  digitalWrite(A3, HIGH);
  digitalWrite(A1, LOW);
  delay(1);

  Y = analogRead(A0);
  delay(1);

  return Y;
}
```