Submitted To

Dr. M. Usman

Control System II: Inverted Pendulum Control

Submitted By

Zainab Tahir (2019 -MC -252)

Qandeel Fatma (2019 -MC -255)

Sabir Husnain (2019-MC-263)

Hassam Ali Raza (2019-MC-291)

University of Engineering & Technology Lahore, Faisalabad Campus

Table of Contents

Control of Inverted Pendulum

1. Abstract

The purpose of this project of to design the inverted pendulum system and control the inverted pendulum model using DC motor and encoder. Here in this project, we have used the LQR and PID control approach to control the model. Here we have sued encoder motors and MPU6050 Inertial Measurement Unit (IMU) for the motion data of the pendulum and to know about its states. For the LQR control we computed the LQR gains and controller the position of cart and position of pendulum of model, for PID we used transfer function of theta to voltage input which can be used to control the output that which is the angular position of the pendulum.

2. Objectives

- To plan, design and build an Inverted Pendulum (IP).
- To use the controller design method to achieve stable control.
- To provide an instructional guide for anyone looking to build an inverted pendulum using PID control.
- To provide model, assembly, and tuning instructions and a cost estimate for the projects.

3. Components Required

- Wooden or Plastic Frame
- Encoder Motors × 2
- MPU-5060 Board
- Timing Belt
- Motor Driver × 2
- ESP32 Development Board
- Steel Rod (Pendulum)
- Steel Rails
- Shaft Bearings × 2
- Linear Bearings x 2

4. Introduction

An inverted pendulum is a pendulum that has its center of mass above its pivot point. It is usually mounted with the pivot point on a moving cart, as shown below. The inverted pendulum is unstable and will fall over unless a force (F) is applied. We will be using the inverted pendulum to learn to tune using servo control. Although the masses and forces may be different for the inverted pendulum system than for your balancing robot, the method for tuning the system is the same.

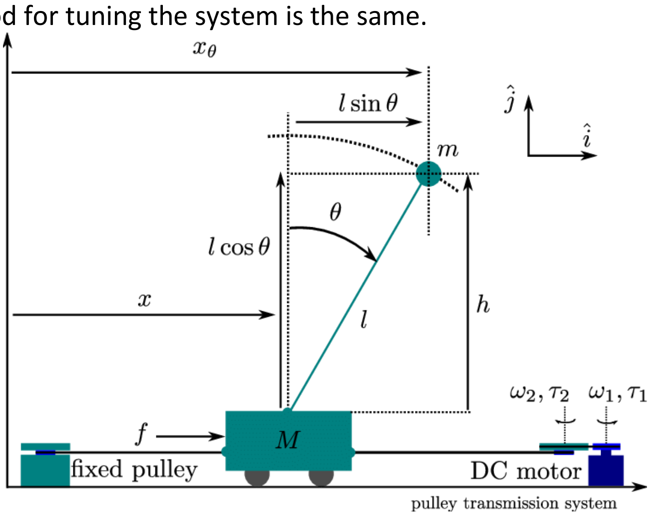

Figure 1: Inverted Pendulum Model

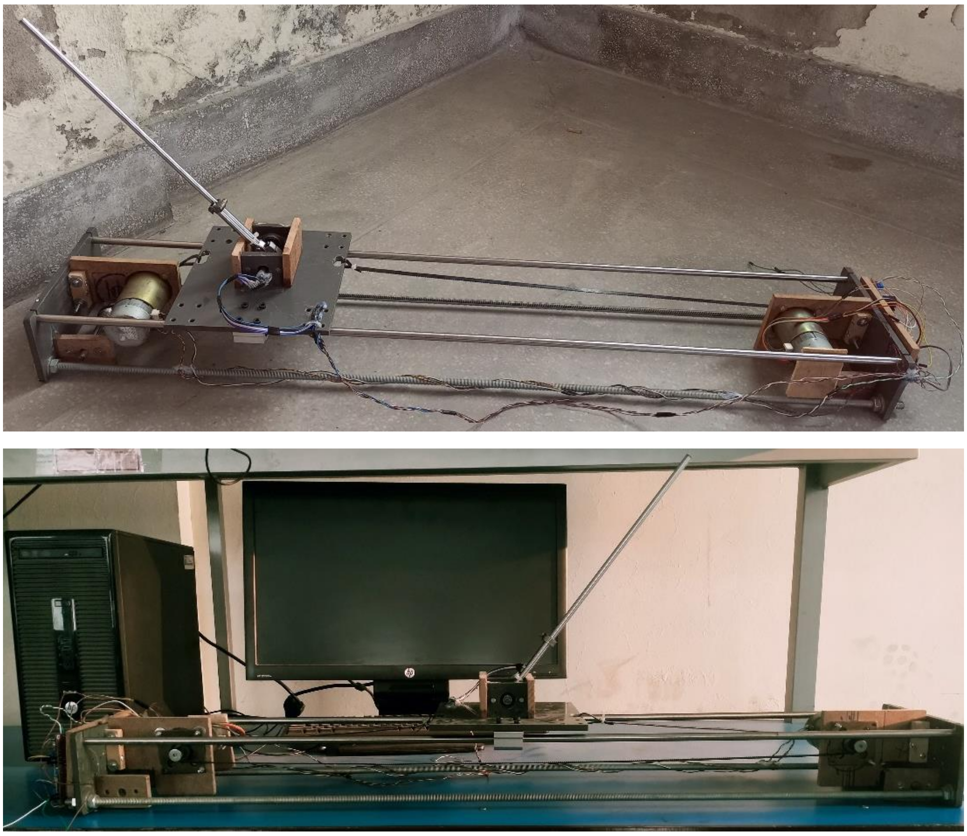

Figure 2: A View of Inverted Pendulum and Cart Hardware

4.1. Components Description

4.1.1. ESP 32

ESP32 is a series of low-cost, low-power systems on a chip microcontroller with integrated Wi-Fi and dual-mode Bluetooth. The ESP32 is a very versatile System on a Chip (SoC). It can be used as a general-purpose microcontroller with quite an extensive set of peripherals, including Wi-Fi and Bluetooth wireless capabilities. Works on low voltage levels (3.3V), ESP32 has 18 ADCs channels while Arduino Uno has only six. ESP32 comes with 48 GPIO pins, while Uno has only14 digital input/output pins and 6 analog pins. The ESP32 board is cheaper than the Arduino Uno.

- Smart industrial devices, including Programmable Logic Controllers (PLCs)
- Smart medical devices, including wearable health monitors.
- Smart energy devices, including HVAC and thermostats.
- Smart security devices, including surveillance cameras and smart locks.

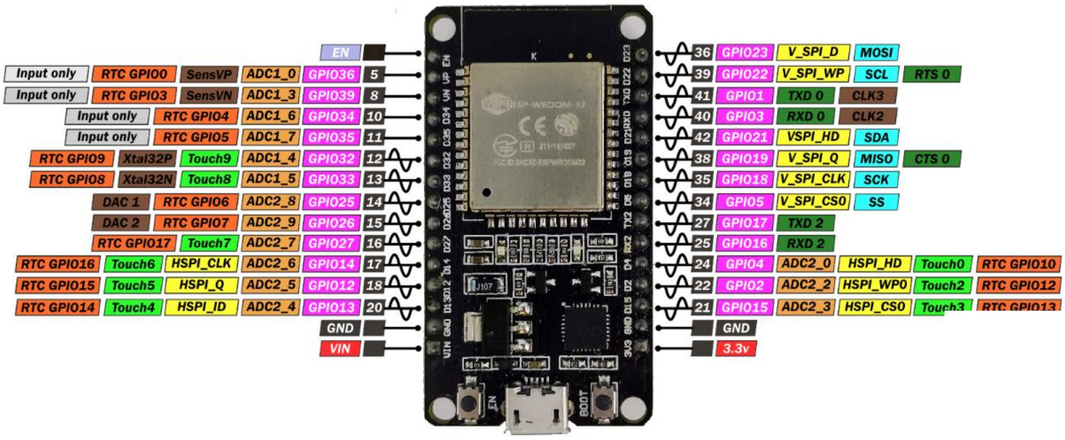

Figure 3: ESP32 Development Board Pinout

4.1.2. Shaft Bearings

Bearings are the parts that assist objects rotation. They support the shaft that rotates inside the machinery. Machines that use bearings include automobiles, airplanes, and electric generators and so on. Bearings are designed to enable rotational or linear movement in a device, bearings are machine elements that are used to reduce friction between moving parts and to enhance the speed and efficiency of a system. They serve three main functions while it facilitates motion: it carries loads, reduces friction and positions moving machine parts. It has many uses. Some of them are as follows:

- Machines that use bearings include automobiles, airplanes, electric generators and so on.
- They are even used in household appliances that we all use every day, such as refrigerators, vacuum cleaners and air-conditioners.
- Bearings support the rotating shafts of the wheels, gears, turbines, rotors, etc.

Figure 4: 8mm Shaft Bearing

4.1.3. Linear Rail Bearing

A linear-motion bearing or linear slide is a bearing designed to provide free motion in one direction. Linear bearings are designed to enable linear movement in a device, bearings are machine elements that are used to reduce friction between moving parts and to enhance the speed and efficiency of a system. They serve three main functions while facilitating the motion: it carries loads, reduces friction, and positions moving machine parts.

- In Machine-tool ways.
- In Sliding doors.
- In 3D printers.
- In a variety of automation settings were reducing friction and guiding linear motion is needed.

Figure 5: Linear Bearings

4.1.4. Timing Belts:

A toothed or synchronous belt is a flexible belt with teeth molded onto its inner surface. Toothed belts are usually designed to run over matching toothed pulleys or sprockets. A Synchronous belt, also known as a timing belt, resembles a flat belt with evenly spaced teeth perpendicular to the belt's axis. The belt teeth are designed to mash with pulley teeth, similar to chains. These belts operate with a consistent efficiency of 98% and maintain their efficiency over a wide load range. The timing belt is there to ensure that the movement of the camshaft or camshafts is synchronized with the movement of the crankshaft.

- Toothed belts are used widely in mechanical devices, including sewing machines, photocopiers, and many others.
- A major use of toothed belts is as the timing belt used to drive the camshafts within an automobile or motorcycle engine.

- As toothed belts can deliver more power than a friction-drive belt, they are used for high-power transmissions.

Figure 6: Timing Belt

4.1.5. DC Encoder Motor:

A motor encoder is a rotary encoder mounted to an electric motor that provides closed-loop feedback signals by tracking the speed and/or position of a motor. Where an encoder is a sensor that detects rotation angle or linear displacement. Encoders are used in devices that need to operate at high speed and with high accuracy. The following motor is a 24-36V encoder motor which can output 200PPR via encoder for rotation sensing, the current of the motor for maximum load s 1.3A and the maximum velocity is 6000RPM, the maximum load torque of the motor is 71mN.m.

- DC motor encoders are used for speed control feedback in DC motors where an armature or rotor with wound wires rotates inside a magnetic field created by a stator.
- The DC motor encoder provides a mechanism to measure the speed of the rotor and provide closed loop feedback to the drive for precise speed control.

Figure 7: Nisca Encoder Motor

4.1.6. MPU5060 IC:

MPU6050 is a Micro Electro-mechanical system (MEMS) that consists of a three-axis accelerometer and three-axis gyroscope. Also, it has the additional feature of the on-chip Temperature sensor. The MPU6050 sensor module is a complete 6-axis Motion Tracking Device. It combines 3-axis Gyroscope, 3-axis Accelerometer, and Digital Motion Processor, all in small package. It has I2C bus interface to communicate with the microcontrollers. It has Auxiliary I2C bus to communicate with other sensor devices like 3-axis Magnetometer, Pressure sensor etc. It helps us to measure velocity, orientation, acceleration, displacement and other motion like features.

Note that, gyroscope and accelerometer sensor data of MPU6050 module consists of 16-bit raw data in 2's complement form.

- In the security and authentication systems, MPU6050 is used for gesture recognition.
- For "no-touch" UI application control and navigation MPU6050 is used.
- In motion command technology for gesture short-cuts, this module is used.
- This module has also found applications in motion-enabled gaming and application frameworks.
- Due to its small size, this module is used in handsets and portable gaming equipment.
- 3D remote controllers, and 3D mice also use this module.

- Wearables used for health, fitness, and sports also contain MPU6050.
- In drones and quadcopters, MPU6050 is used for position control.
- This module has also found application in self-balancing robots.
- MPU6050 is highly preferred for robotic arm control.

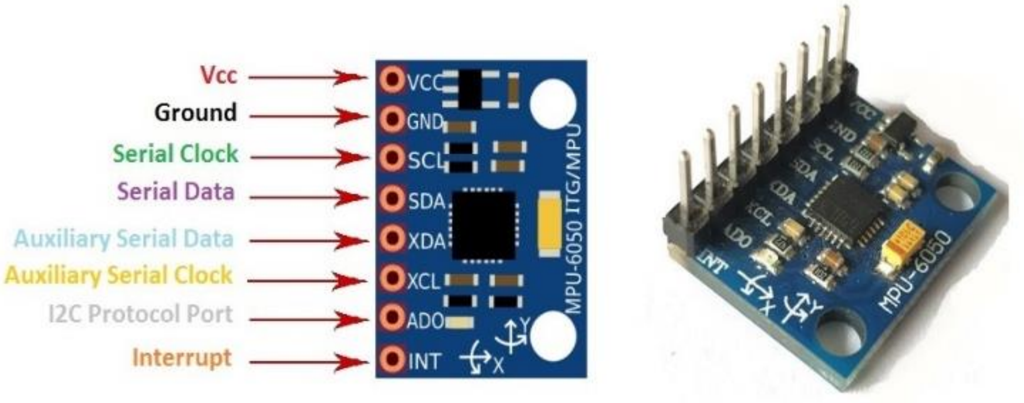

Figure 8: MPU-6050 Breakout Board

4.1.7. L293D Motor Driver:

The L293D Motor Driver IC is a controller that can control a set of two DC motors simultaneously in any direction and speed. Motor drivers act as an interface between the motors and the control circuits. Motors require a high amount of current, whereas the controller circuit works on low-current signals.

- The function of motor drivers is to take a low-current control signal and turn it into a higher-current signal that can drive a motor.
- The L298N Motor Driver module consists of an L298 Motor Driver IC, 78M05 Voltage Regulator, resistors, capacitor, Power LED, 5V jumper in an integrated circuit.
- The L298N Motor Driver Module is a high-power motor driver module for driving DC and Stepper Motors.

The L298 Dual H-Bridge Motor Driver IC has multiple applications. It is generally used to control the direction of the motor as well as motor speed. It is applicable in different fields like robotics, embedded, etc. Here are a few areas where L298 IC is applicable.

- It is basically used in applications where H-bridge is used. i.e., in H-bridge-based applications
- It is applicable to applications where current control and PWM operable IC are required.
- It is also applicable to several real-life applications like Relay drivers, Robotics, automatic door control systems, Weight lifters, etc.
- It is used in applications where a high-power motor driver is required. Since the microcontrollers work on very little voltage and current, L293 motor IC is preferred for high voltage and current applications.

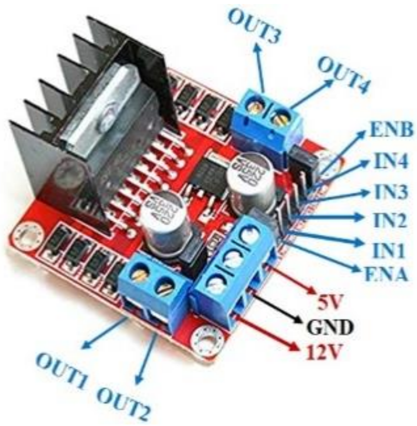

Figure 9: L293D Motor Driver

4.2. Circuit Diagram

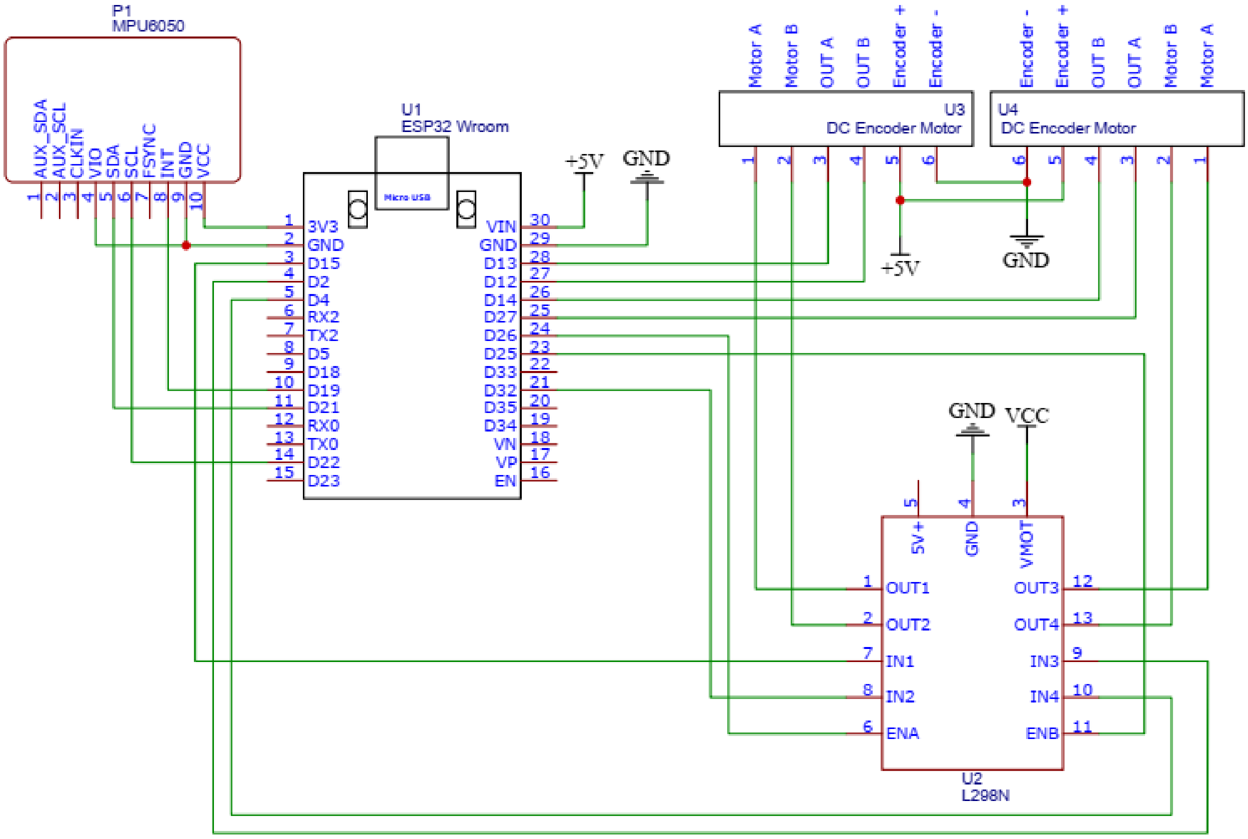

Figure 10: Circuit Diagram

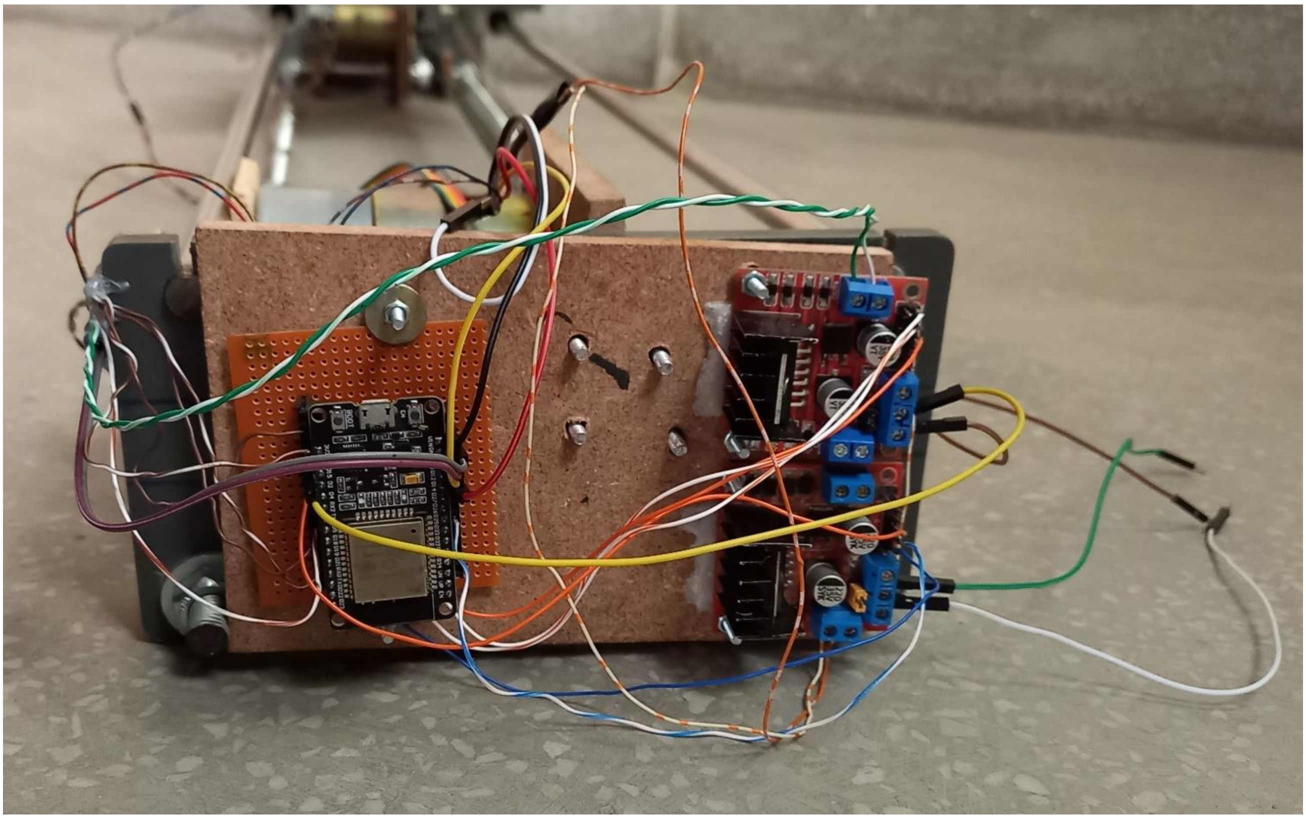

Figure 11: Physical Circuit of the Inverted Pendulum and Connections

4.3. Project Time Line

Table 1: Project Timeline

Sr. No.	No. of Week	Task to Achieve
1.	Week 6	Issuing the Hardware
2.	Week 8	Busying Hardware Components
3.	Week 10	Testing the DC Motor
4.	Week 11	Feedback Sensor
5.	Week 12	Hardware Completion
6.	Week 13	Controller Selection and MATLAB implementation
7.	Week 14	Controller and Implementation
8.	Week 15	Final Submission

4.4. Cost Estimation

Table 2: Project Cost Estimation (21 Dec, 2022)

Sr. No.	Name	Amount Required	Price/Piece	Total Price
1.	8mm Horizontal Linear Bearings	2	250	500
2.	ESP32 Controller	1	1350	1350
3.	MPU-6050 (IMU Sensor)	1	950	950
4.	8mm-8mm Shaft Coupler	1	220	220
5.	8mm Linear Rail Shaft (400mm)	1	580	580
6.	8mm Linear Rail Shaft (250mm)	1	380	380
7.	Hole to Hole Jumpers	1	130	130
8.	2 Core Twisted Wire	1	40	40
9.	Close Timing Belt	2	290	580
10.	Open Timing Belt & 20 Teeth Timing Pulleys (8mm Bore)	1	950	950
11.	40 Teeth Timing Pulleys (8mm Bore)	2	508	1016
12.	Carbon Wound Pot (1k)	1	150	150
13.	DC Motor Driver	2	350	700
14.	Frame	1	--	2000
15.	Encoder Motors	2	2500	5000
16.	Others (Screws, Nuts, Bolts)	--	--	500
	Total Project Price		**Rs. 15000 (Approx.)**	

4.5. Applications

There are many applciations of inverted pendlum and cart model. SO me of those applciaitons are given below: -

- Rocket propellers works on the principle of inverted pendulum and cart model, the robkect engines acts as motor as we are using here and adjust the rocket angle.
- Self balancing mobile robots uses the same principl of inverted pendulum to be stand on two wheels or two legs.
- Humanoid robots use this principle to stablize and to be stand on the ground.

5. Methodology

5.1. System Modelling

Let's start with the system modelling. First of all, split the model into two major parts, here we can say there are two systems, electrical and mechanical. The mechanical model of the inverted pendulum and cart can be represented using the equations,

$$F = (M + m)\ddot{x} - ml\ddot{\theta}$$

$$-mgl\theta = ml\ddot{x} - (I + ml^2)\ddot{\theta}$$

Where F is the force applied on the cart as an input by the actuator or external environment, x is the translational position of the cart, θ is the angular position of the pendulum, l is the half of the length of the pendulum rod or the position of its center of mass from its base, and the I is the moment of inertia of the pendulum the diagram of the cart and pendulum is shown in the Figure 1.

Now the electrical model can be represented using equation of DC motors which are given below, then the two models can be joined together to get a complete model of the system.

$$T_m = K_m i = J_m\ddot{\theta}_m + b_m\dot{\theta}_m + T_L$$

$$\frac{Ld(i)}{dt} + Ri + K_m\dot{\theta}_m = V$$

Here the K_m is the motor constant, J_m is the inertia of the rotor of the motor, θ_m is the position of the motor, i is the motor current, R is the resistance of the armature, L is the inductance of the armature, and V is the input voltage of the motor.

Now we can say that the motor can again be split up into two components one is electrical and other is mechanical as it is also an electromechanical system, and we can also say that the electrical part of the motor is very fast and the dominant poles are only mechanical poles thus the electrical part can be ignored for an approximated motor model which can be represented in first order using equations.

$$Ri + K_m\dot{\theta}_m = V$$

$$K_m i = J_m\ddot{\theta}_m + b_m\dot{\theta}_m + T_L$$

We can relate the motor position θ with the position of the cart x, using a simple relation that is $x = r\theta$, where r is the radius of the pullies attached with the motor and then with the cart using timing belt. Thus, the equation can then be related with the force using relation $T = rF$, and we can say that

$$F = \frac{K_m}{r}V - \frac{K_m}{r^2R}\dot{x} - \frac{J_m}{r^2}\ddot{x} - \frac{b_m}{r^2}\dot{x}$$

Putting it in the equation of the inverted pendulum represented using force input we can get a modified model of inverted pendulum with motor voltage input which is represented using.

$$\frac{K_m}{r}V - \left(\frac{K_m}{r^2R} + \frac{b_m}{r^2}\right)\dot{x} = \left(M + m + \frac{J_m}{r^2}\right)\ddot{x} - ml\ddot{\theta}$$

$$-mgl\theta = ml\ddot{x} - (I + ml^2)\ddot{\theta}$$

The above equations can be solved for the $\ddot{\theta}$ and $\ddot{x}$ using Cramer's tule. Here let us consider some intermediate variables to make the processes easier for us. Let,

$$M_1 = \left(M + m + \frac{J_m}{r^2}\right)$$

$$I_1 = ml$$

$$I_2 = (I + ml^2)$$

$$C_1 = \frac{K_m}{r}$$

$$C_2 = \left(\frac{K_m}{r^2 R} + \frac{b_m}{r^2}\right)$$

$$C_3 = mgl$$

Then the model can be represented as,

$$\begin{bmatrix} M_1 & -I_1 \\ I_1 & I_2 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} C_1 V - C_2 \dot{x} \\ -C_3 \theta \end{bmatrix}$$

After solving and writing it in state space form the model is represented as follows,

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{M_1 C_3}{Z} & 0 & 0 & \frac{I_1 C_2}{Z} \\ 0 & 0 & 0 & 1 \\ -\frac{C_3 I_1}{Z} & 0 & 0 & \frac{C_2 I_2}{Z} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ -I_1 C_1 / Z \\ 0 \\ -\frac{C_1 I_2}{Z} \end{bmatrix} u(t)$$

Where, $x_1 = \theta, x_2 = \dot{\theta}, x_3 = x, x_4 = \dot{x}, u(t) = V$, and $Z = I_1^2 - M_1 I_2$

The parameters for the described system in this report are shown in the following table,

Table 3: Model Parameters

Sr. No.	Parameter	Value	Units
1.	M	0.8530	Kg
2.	m	0.137	Kg
3.	I	0.0014	$Kg.m^2$
4.	l	0.175	m
5.	J_m	2.85×10^{-5}	$Kg.m^2$
6.	K_m	0.0659	$N.m.A^{-1}$
7.	L_m	0.0630	H
8.	R_m	6.7	Ω
9.	B_m	4.55×10^{-6}	$N.m.rad^{-1}.s$
10.	r	0.005	m

5.2. Non-Linearities

There are certain non-linearities in the model and some unmodeled dynamics also which can cause trouble in our system and should be taken care of when necessary, using control algorithms and other clever techniques. Some of the non-linearities are given below.

- Ignoring the friction in the model, we are using frictionless bearings in the model but this is still a nonlinearity as the friction can't completely be compensated using the bearings thus have to include the friction but with friction the system becomes very complex and can't easily be controller.
- Motors used here comes with no parameters thus the parameters were computed or estimated using the parameter estimation algorithm. These parameters are not accurate but approximated version of the motor. Here is the real motor output and approximated motor output shown in Figure 12.
- The motor stops when the voltage becomes less than 4V during control procedure and the maximum motor voltage must no increase 17V because above voltage will leads towards more current draw and the motor driver will burnout as it can only give 25W of power.

- There is sensor noise which is also a non-linearity, the noise is due to the quantization error of the encoder as it can only give 200PPR.

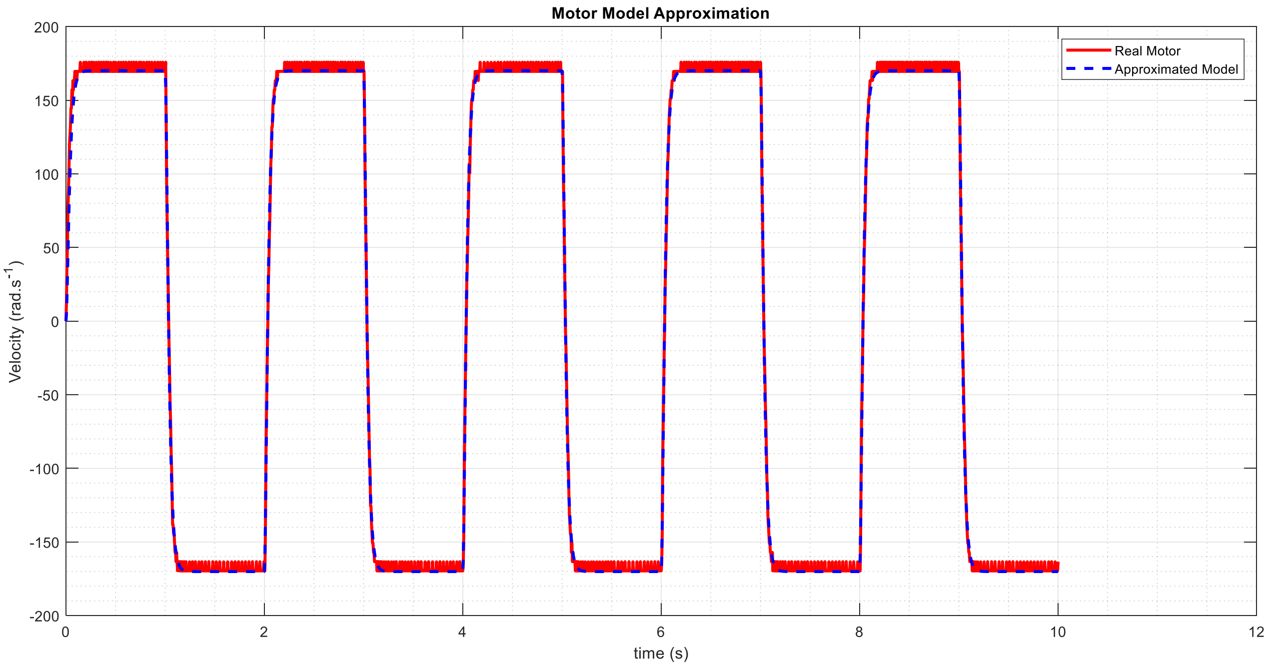

Figure 12: Motor Parameter Estimation

5.3. Control Law

Here we have employed two types of controllers for the inverted pendulum can cart model, one is LQR and second is Robust PID control. The two controllers are explained below, about how they were designed and what gains are used for the model.

5.3.1. LQR Control

Here the LQR control is employed which is abbreviation for Linear Quadratic Regulator. The LQR controller gives us facility to control the system with optimized control parameter. Here the matrices Q and R are responsible for the optimal performance of the controller. The design of optimal control systems is an important function of control engineering. The purpose of design is to realize a system with practical components that will provide the desired performance. The desired performance can be readily stated in terms of time-domain performance indices, such as the integral performance measures. The design of a system can be based on minimizing a performance index, such as the integral of the squared error (ISE). Systems that are adjusted to provide a minimum performance index are called optimal control systems. Here is the MATLAB script for the LQR control for this particular system, the data files and the script itself can be accessed from the GitHub repository https://github.com/sabirhusnain577/Inverted-Pendulum-LQR-PID-Using-Low-Cost-Sensor-MPU6050.git.

```
%% Model
%%%
load('myNewMotor4.mat');
r=0.01; g=9.81;
M=0.770+0.032+0.019+0.032;
m=0.137;
l=0.175;
I=(1/12)*m*(2*l)^2;

M_1=M+m+(J_m/(r^2));
I_1=m*l;
C_1=K_m/(r*R_m);
```

```
C_2=((K_m^2)/((r^2)*R_m))+(B_m/(r^2));
C_3=m*g*l;
I_2=I+m*(l^2);
Z=-(M_1*I_2)+(I_1^2);

A=[0 1 0 0;
    -M_1*C_3/Z 0 0 I_1*C_2/Z;
    0 0 0 1;
    -C_3*I_1/Z 0 0 C_2*I_2/Z];
B=[0; -I_1*C_1/Z; 0; -C_1*I_2/Z];
C=[1 0 0 0;
    0 0 1 0];
D=zeros(2,1);

Q=[50 0 0 0;
    0 1 0 0;
    0 0 50 0;
    0 0 0 1];
R=0.30;
[K,P,E]=lqr(A,B,Q,R);

%% Controller System Bahvaiour
t=0:0.01:5;
sys=ss(A-B*K,[],eye(4),[]);
x_Loop=initial(sys, [0.2;0;-0.20;0], t);

figure(1);
subplot(3,2,1);
plot(t,x_Loop(:,1),'-r', 'LineWidth', 2); grid on; grid minor;
title('Close Loop System State x_1');
xlabel('time (s)'); ylabel('Amplitude');

subplot(3,2,2);
plot(t,x_Loop(:,2),'-g', 'LineWidth', 2); grid on; grid minor;
title('Close Loop System State x_2');
xlabel('time (s)'); ylabel('Amplitude');

subplot(3,2,3);
plot(t,x_Loop(:,3),'-b', 'LineWidth', 2); grid on; grid minor;
title('Close Loop System State x_3');
xlabel('time (s)'); ylabel('Amplitude');

subplot(3,2,4);
plot(t,x_Loop(:,4),'-m', 'LineWidth', 2); grid on; grid minor;
title('Close Loop System State x_4');
xlabel('time (s)'); ylabel('Amplitude');

U=-K*x_Loop';
subplot(3,2,[5 6]);
plot(t,U,'-k', 'LineWidth', 2); grid on; grid minor;
title('Control Signal');
xlabel('time (s)'); ylabel('Amplitude');
```

Here is the output for the LQR control for the model presented above using equations for the inverted pendulum and cart system, here we can see that the system is applied with some initial conditions and it get backs to zero from those conditions and settles at zero in very less time. One can also observer the control is signal output u(t) in the following plots the u(t) is motor voltages which is less than 15V here which was desired for the model to work properly in our case.

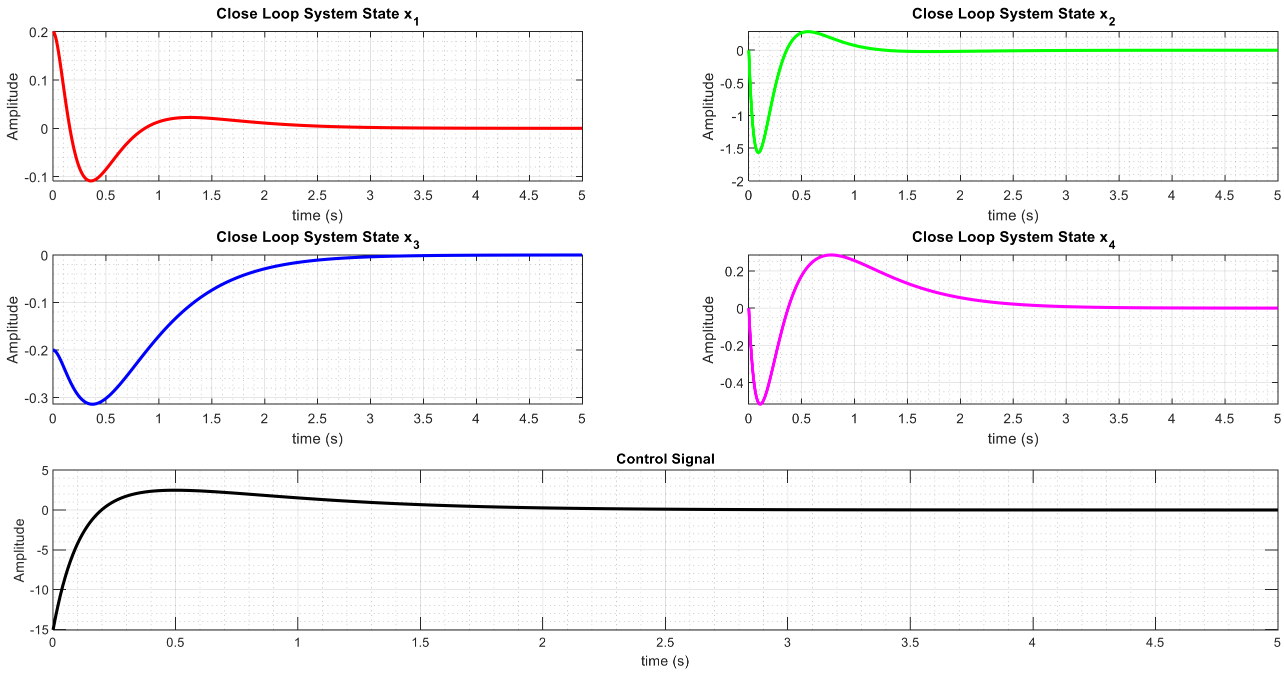

Figure 13: Output Response of LQR Controller

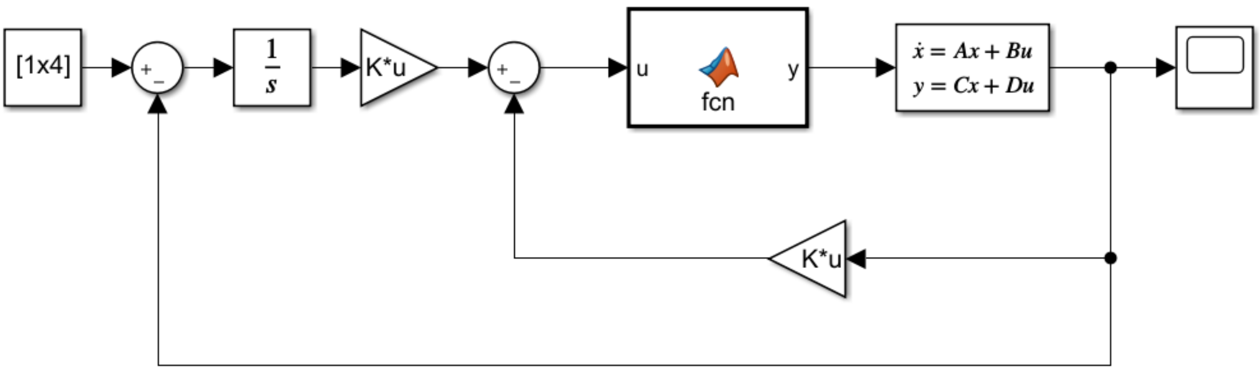

Figure 14: Block Diagram for LQR Control

5.3.2. Robust PID Control

Robust PID control is applied here on the model of inverted pendulum because of its robust behavior, if there is any ambiguity in the modelling of the system the controller will take care of it and will control the system properly. A robust control system maintains acceptable performance in the presence of significant model uncertainty, disturbances, and noise. Here is the MATLAB script for the PID control for this particular system, the data files and the script itself can be accessed from the GitHub repository https://github.com/sabirhusnain577/Inverted-Pendulum-LQR-PID-Using-Low-Cost-Sensor-MPU6050.git.

```
clear; clc; close all;

%% Model
%%%
load('myNewMotor4.mat');
r=0.01; g=9.81;
M=0.770+0.032+0.019+0.032;
m=0.137;
```

```
l=0.175;
I=(1/12)*m*(2*l)^2;

M_1=M+m+(J_m/(r^2));
I_1=m*l;
C_1=K_m/(r*R_m);
C_2=((K_m^2)/((r^2)*R_m))+(B_m/(r^2));
C_3=m*g*l;
I_2=I+m*(l^2);
Z=-(M_1*I_2)+(I_1^2);

A=[0 1 0 0;
    -M_1*C_3/Z 0 0 I_1*C_2/Z;
    0 0 0 1;
    -C_3*I_1/Z 0 0 C_2*I_2/Z];
B=[0; -I_1*C_1/Z; 0; -C_1*I_2/Z];
C=[1 0 0 0;
    0 0 1 0];
D=zeros(2,1);

%% Control Design & Implimentation
%%%
[num den]=ss2tf(A,B,C,D);
G=tf(num(1,:),den);
Num_tf=G.Numerator{1,1};
Den_tf=G.Denominator{1,1};
N1=Num_tf(3); N2=-Num_tf(4); N3=Num_tf(5);
D1=Den_tf(2); D2=-Den_tf(3); D3=-Den_tf(4);

K_D=(28-D1)/N1
K_I1=100000/N3
K_P=(D2/N1)+((N2/(N1^2))*(28-D1))+500/N1

K_I2=(N3*K_P-34000)/N2

K_I3=(5500-N3*K_D+N2*K_P+D3)/N1

% Gc1=tf([K_D K_P K_I1], [1 0]);
% Gc2=tf([K_D K_P K_I2], [1 0]);
Gc3=tf([K_D K_P K_I3], [1 0]);

% sys_Loop1=feedback(series(Gc1,G),1);
% sys_Loop2=feedback(series(Gc2,G),1);
sys_Loop3=feedback(series(Gc3,G),1);

sys_Robust=tf(10^5,[1 2.8*10 5*10^2 5.5*10^3 3.4*10^4 10^5]);

Gp=sys_Robust/sys_Loop3;

sys=series(Gp,sys_Loop3);

[imp_Res,t]=impulse(sys,2);
subplot(1,2,1);
plot(t,imp_Res,'-r','LineWidth',2);
grid on; grid minor;
xlabel('time (s)'), ylabel('Angle (rad)');
title('Impulse Response of the System with Pre-Filter');

[imp_Res,t]=impulse(sys_Loop3);
```

```
subplot(1,2,2);
plot(t,imp_Res,'-b','LineWidth',2);
grid on; grid minor;
xlabel('time (s)'), ylabel('Angle (rad)');
title('Impulse Response of the System without Pre-Filter');

%% Pre Filter Embedded Designing
Gp_z=c2d(Gp,0.0005,'tustin');
Gp_z.Variable='z^-1';
Gp_z
```

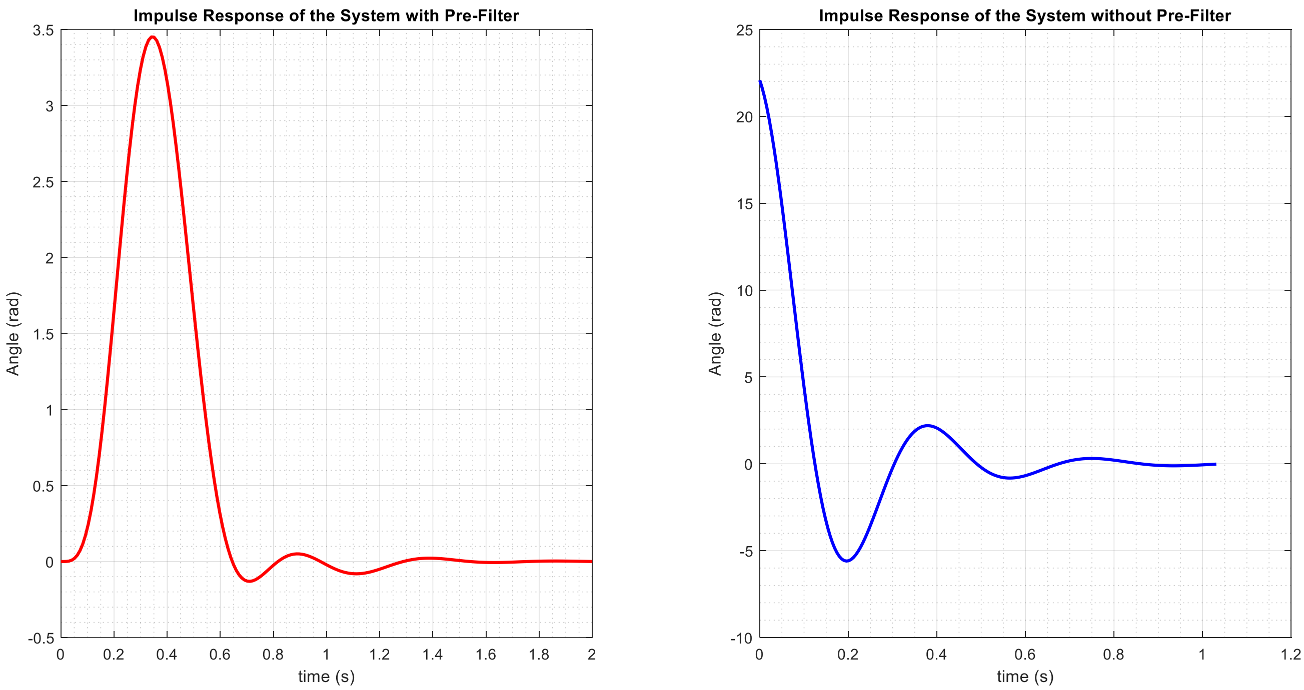

Figure 15: Impulse Response of PID Controller

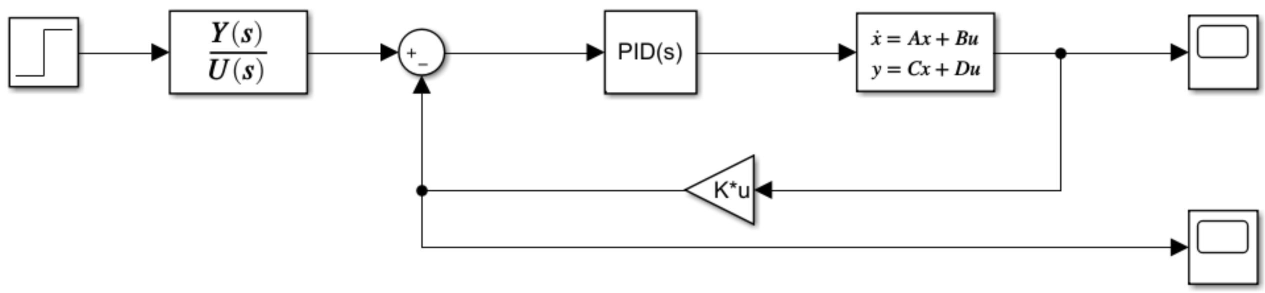

Figure 16: Block Diagram for PID Control

5.4. Embedded Control Algorithm

The flow chart of the embedded controller is shown below, here we are ESP32 Wroom for the embedded system implementation for the controller in C++. The Arduino framework along with FreeRTOS was used here in the program which allows us to run all the tasks in parallel thus giving real time implementation of the control using Linux kernel of RTOS (Real Time Operating System) and time management in a dual core processor. Following flowchart explains the basics of the embedded system code and gives a basic idea how the program works.

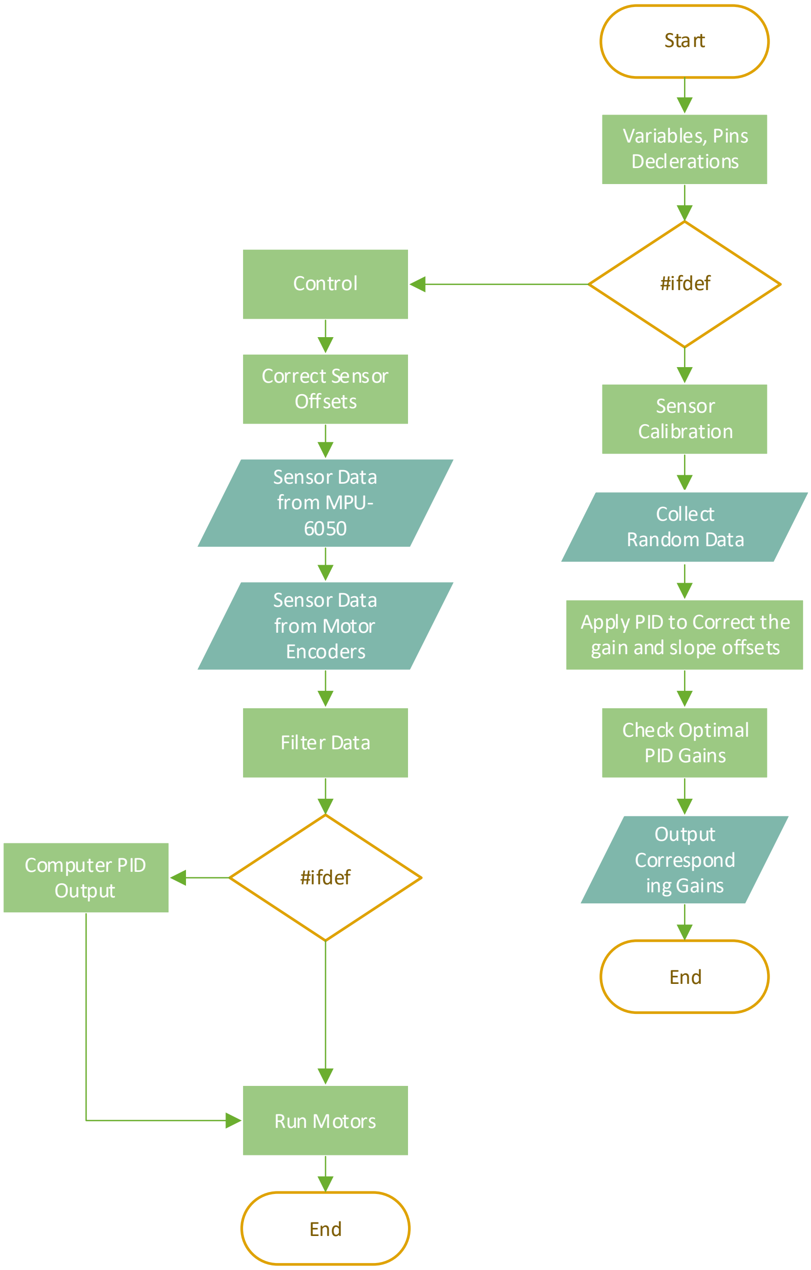

Figure 17: Flow Chart for Embedded Code of the Control

6. Results & Discussion

6.1. Challenges & Their Solutions

The most important and critical task in the system is to obtain the states of the system or in other words to obtain the data using sensors, here we can easily obtain the rotary sensor data using interrupts and some coding in C++ and esp32 but the critical thing here is to get the pendulum angle, for the pendulum angle we are using MPU6050 which consists of 3 accelerometers, and 3 gyro sensors. The gyro sensors provide us rotational velocity and accelerometers gives us the acceleration. We can use these sensors to easily obtain the orientation of a rotating body. The angle can be obtained both from gyro scope an accelerometer as follow.

6.1.1. Gyro Sensor Angle Measurement & Issues with It

One can measure 3 angular velocities with the sensor, now how to measure angle with the angular velocity. So, the solution lies in the definition of velocity. It is stated as “the rate of change of distance with respect to time.” So, if there is a constant angular velocity in certain time “t” one can measure angular displacement easily by dividing velocity by time. Hence, here in this project data is being read by microcontroller in a certain fixed time and then dividing the value of velocity by the time it is added to the previous angle for a certain axis. It can clearly be seen in the below flowchart diagram.

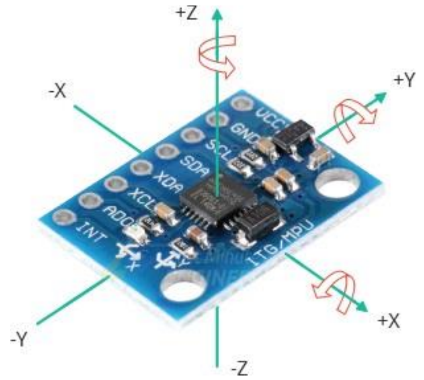

Figure 2: MPU6050 Gyro + Accelerometer with Axis labeled

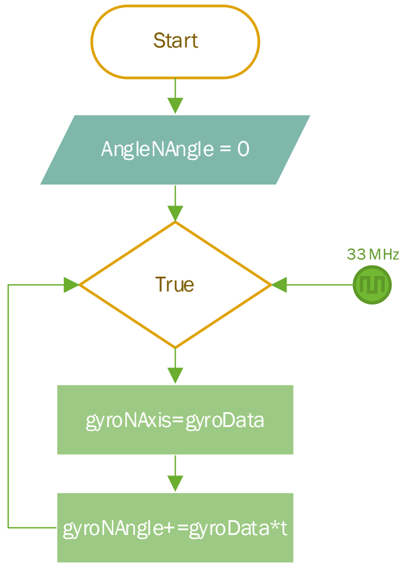

Figure 18: Flow Chart for Angular Motion Sensing Using MPU6050 (Gyro)

So, for every axis we can measure angle easily using gyro by reading the specific axis data and dividing it by loop time. Loop time will be found by microcontroller itself hence it is very easy to measure the angle using gyro. But there is problem I angle measurement with gyro only which is drifting of angle. angle of gyro drifts automatically after some time when it remains stationary on a stationary platform. This drifting of angle is due to the vibrations in the gyro these vibrations cause a minor angular velocity which is being added in the angle after division by time.

Another problem is that assume if gyro is initially at an angle in that case it will read zero because there was no angular velocity sensed by the microcontroller hence if there is no motion sensed by controller then no angle. So, there are two limitations of the gyro sensor and those are:

- Noise (Drift)
- Incremental Output (No Initial Angle)

6.1.2. Angle Measurement using Accelerometer & Issues with It

Now accelerometer is used to measure acceleration of a body but there is a trick by which it can be used to measure the angle. The sensor we are using is a three-axis accelerometer but first let a two-axis accelerometer for simplicity. In below figure a box is being shown which consists of two accelerometers for measuring forces in x and y direction. When a force is applied on the accelerometer an inertial force will be produced in opposite direction of motion which can be used to measure the acceleration in the direction of motion.

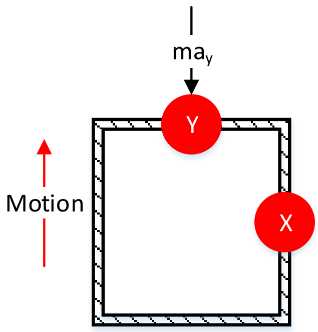

Figure 19: Accelerometer with Movement in Y-Axis

We can see in above figure that the during motion in y-axis a motion is being applied on Y sensor and similarly it will be applied on X sensor when there is a motion in x-axis. The gravitational force is being always applied on the sensors always in every condition if it is in gravitational field of the earth. Now consider following situation and sensor is in motion in a diagonal axis then the forces will be applied on both the sensors shown below.

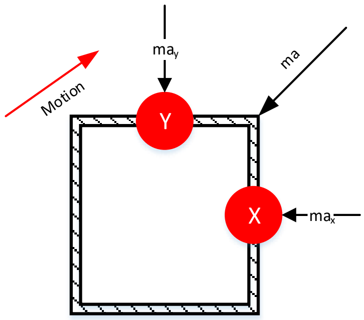

Figure 20: Accelerometer with Movement in Diagonal Axis

As shown in above figure a force will be applied on the Y sensor as well as on X sensor if sensor in under motion in diagonal axis. Hence using trigonometry, it can be observed that;

$$ma = \sqrt{(ma_x)^2 + (ma_y)^2}$$

$$\theta_x = sin^{-1}\left(\frac{a_y}{a}\right)$$

$$\theta_y = sin^{-1}\left(\frac{a_x}{a}\right)$$

The third angle which is with the z-axis can't be measured using accelerometer. Now the question is that why don't we use accelerometer for the angle measurement for x and y axis then the answer is that there is a bigger problem with the accelerometer. If we shake the sensor there will be inaccurate angles measured using accelerometer because there will be other forces rather than gravitational forces. So, if there are vibrations of motor or something like gamble which is going to be in human hand will have a chance of vibrations so we can't use accelerometer at all for angle measurement.

6.1.3. Solution for Angle Drifting

Angle drifting can be stopped using weighted sum of data of both the sensors. If we use complimentary filter, it is such a filter which defines a trust factor and then on that basis we can say that we are trusting this sensor this much and this much for this one, this is very common method of sensor fusion.

6.1.4. Solution for Initial Angle Measurement

Initially angles can be measured only using accelerometer. If the gyro is placed at an angle, then there will be initial angles which are accelerometer angles. When microcontroller starts an angle set flag is set to zero and loop start and there is a condition that if angle set flag is zero then only accelerometer data should have to be read otherwise weighted sum. So following flowchart will explain the logic clearly.

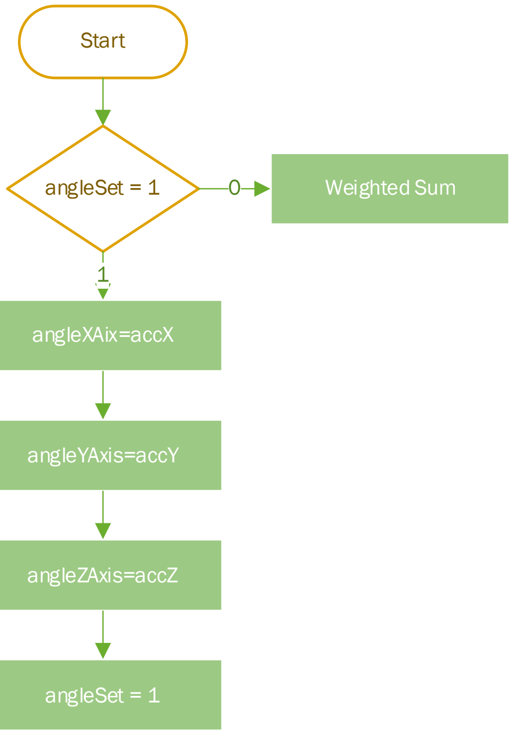

Figure 21: Angle Measurement using Sensor Fusion (Complimentary Filter)

6.2. Issues with The Complimentary Filter & Solution

As we have explained above that the complimentary filter method will work quite well but for rotational bodes but here the cart is being translated and ultimately the pendulum is also with respect to the worlds frame. Here we can easily observer that the angle will always be wrong if we measure it using gravity because now the gravity vector has been added with the world acceleration and cannot be separated out. Here we are using a geometrical solution which can help us to sperate out these two accelerations, it breaks down the acceleration into two components one will be gravity and the other will be world acceleration, in the embedded code we have named it as helper_3dMath library, which helps us to separate out the two readings depending upon the actual motor acceleration and other factors based on statistics beliefs and probabilities. Then at the end we care using Kalman's Filter to mix that two readings that is gyro and accelerometer to get the very precise angle without the effect of world acceleration using gravity method. All that code can be found out in the GitHub repository https://github.com/sabirhusnain577/Inverted-Pendulum-LQR-PID-Using-Low-Cost-Sensor-MPU6050.git.

7. Conclusion

At the end of the day, we were able to control the pendulum using LQR and PID controls, we are getting very accurate pendulum angle which can be used as a state in the LQR or PID control algorithm. The response of the inverted pendulum was good enough and it was stabling the pendulum even if disturbed slightly. The output response of the pendulum can be seen in the figure shown above in the MATLAB section where the control signal and output can be seen for PID and LQR both.

8. References

[1]. Bogdanov, A., 2004. Optimal control of a double inverted pendulum on a cart. Oregon Health and Science University, Tech. Rep. CSE-04-006, OGI School of Science and Engineering, Beaverton, OR.

[2]. Muskinja, N. and Tovornik, B., 2006. Swinging up and stabilization of a real inverted pendulum. IEEE transactions on industrial electronics, 53(2), pp.631-639.

[3]. Fedorov, D.S., Ivoilov, A.Y., Zhmud, V.A. and Trubin, V.G., 2015. Using of measuring system MPU6050 for the determination of the angular velocities and linear accelerations. Automatics & Software Enginery, 11(1), pp.75-80.

[4]. Ooi, K.R., Rosli, M.A.A., Latiff, A.R.A., Othman, W.A.F.W., Alhady, S.S.N. and Wahab, A.A.A., 2021, July. Design of hoeckens linkage-based walking robot with MPU6050 IMU as navigation sensor. In Journal of Physics: Conference Series (Vol. 1969, No. 1, p. 012004). IOP Publishing.

[5]. Jian, H., 2016, June. Design of angle detection system based on MPU6050. In 7th International Conference on Education, Management, Information and Computer Science (ICEMC 2017) (pp. 6-8). Atlantis Press.

[6]. Ngo, H.Q.T., Nguyen, T.P., Huynh, V.N.S., Le, T.S. and Nguyen, C.T., 2017, July. Experimental comparison of complementary filter and kalman filter design for low-cost sensor in quadcopter. In 2017 International Conference on System Science and Engineering (ICSSE) (pp. 488-493). IEEE.