German University in Cairo

Mechatronics Engineering (MCTR601)

SELF-BALANCING ROBOT

Name	ID	Lab Number
Amr Wael Fathallah	52-12977	T-34
Ashraf Ahmed	52-21208	T-32
Nabil Hesham Hassan	52-11233	T-28

Table of Contents

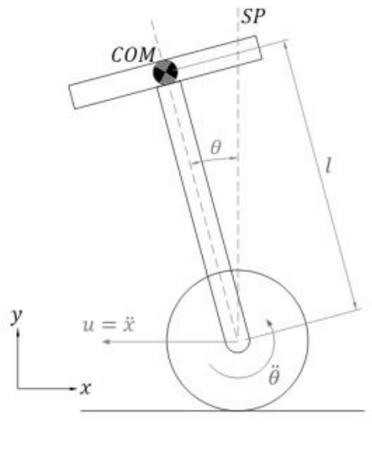
1. Project Description 3
2. Methodology 5
2.1 Mechanical design 5
2.2 Electrical design 6
2.3 Control 7
  2.3.1. Modeling 7
  2.3.2. Analysis 8
  2.3.3. Controller Design 9
2.4 Programming 12
3. Design Evaluation 13
4. Appendix 14

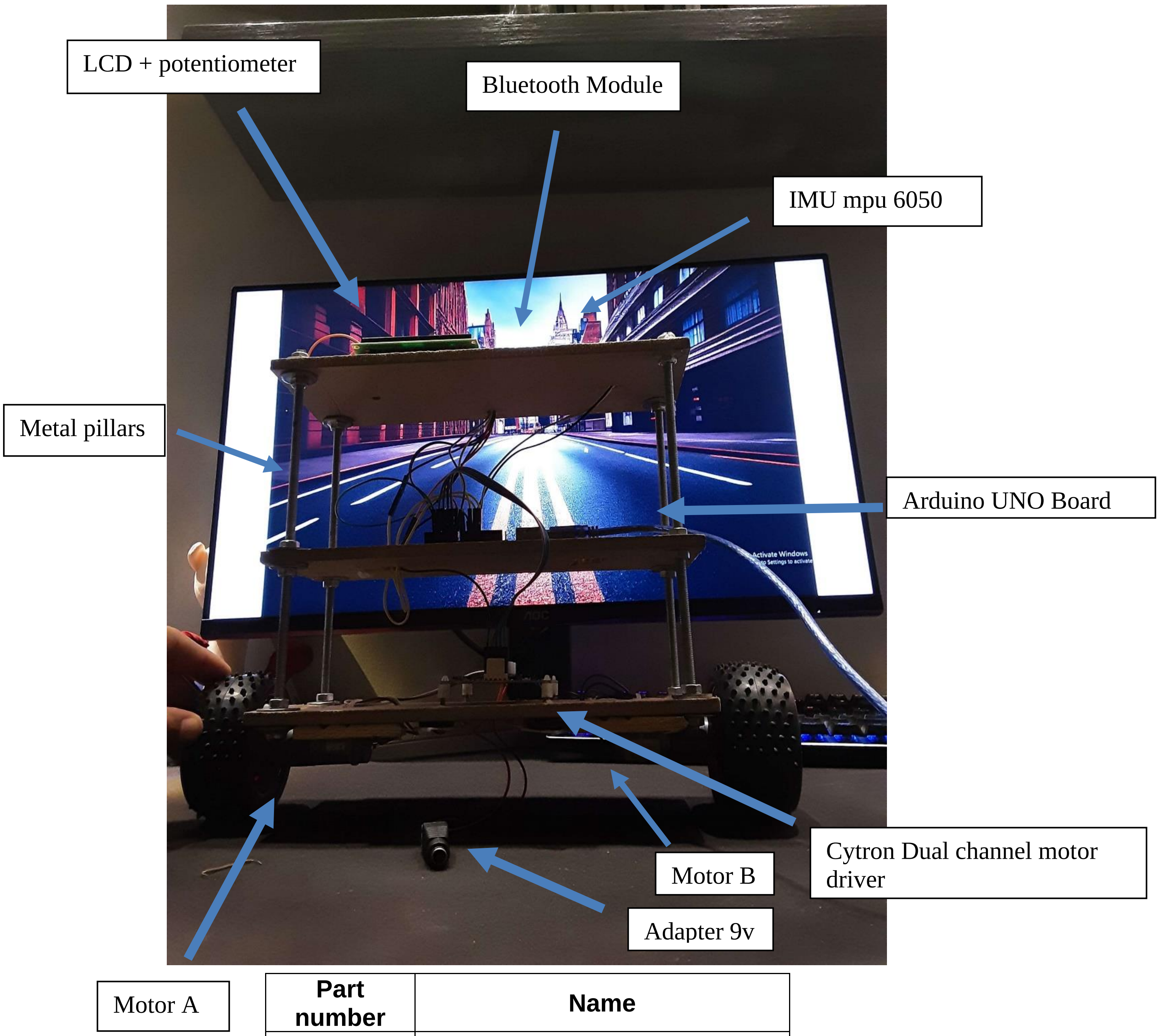
1. Project Description

- A self-balancing robot is a type of robot that can balance itself autonomously without external support. It typically uses a combination of sensors, motors, and algorithms to maintain balance, allowing it to move and interact with its environment. Self-balancing robots have a wide range of applications, from educational toys to industrial automation. They are often used in research, such as studying locomotion or as a base for developing humanoid robots. Some popular examples include the Segway personal transporter and various drone models.

- Self-balancing robots have numerous applications, ranging from toys for entertainment purposes to complex machinery for inspection and surveillance tasks. In the industrial field, self-balancing robots can be utilized in assembly lines to carry out repetitive duties. In agriculture, they can be utilized to monitor crop fields. In healthcare, self-balancing robots can help deliver medication to patients, while in the entertainment industry, these robots can be used for a variety of tasks, including providing rides at amusement parks. Additionally, self-balancing robots are being developed for use in search and rescue, military, and even space exploration applications.

- A self-balancing robot typically consists of a microcontroller (Arduino in our case), motor driver, DC motors, battery, IMU (inertial measurement unit), and various sensors such as gyroscopes and accelerometers. The microcontroller receives data from the sensors and calculates precise motor movements to maintain balance. The motor driver controls the speed and direction of the motors. The battery (Adapter in our case) provides power to the system. Various sensors help the robot detect its surroundings and respond accordingly. Overall, the combination of these components allows the self-balancing robot to maintain stability and move seamlessly.





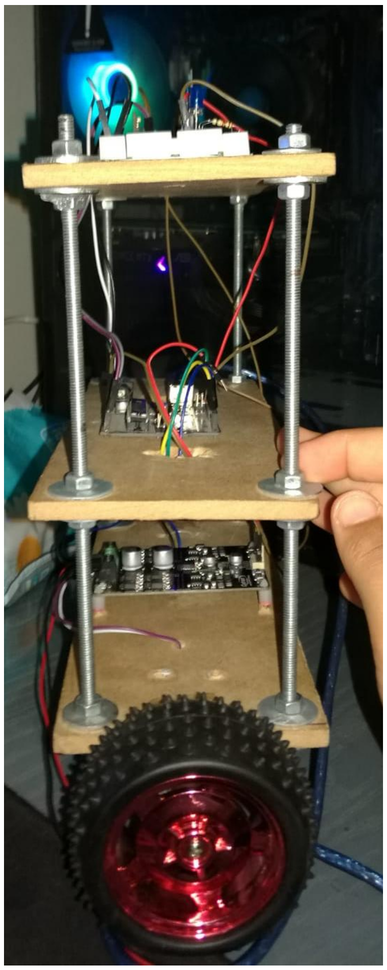
Part number	Name
1	Cytron Dual Channel 10A DC motor driver
2	DC motors
3	IMU MPU 6050
4	Arduino UNO
5	9V Adapter
6	Bluetooth Module
7	Potentiometer

2. Methodology

2.1 Mechanical design

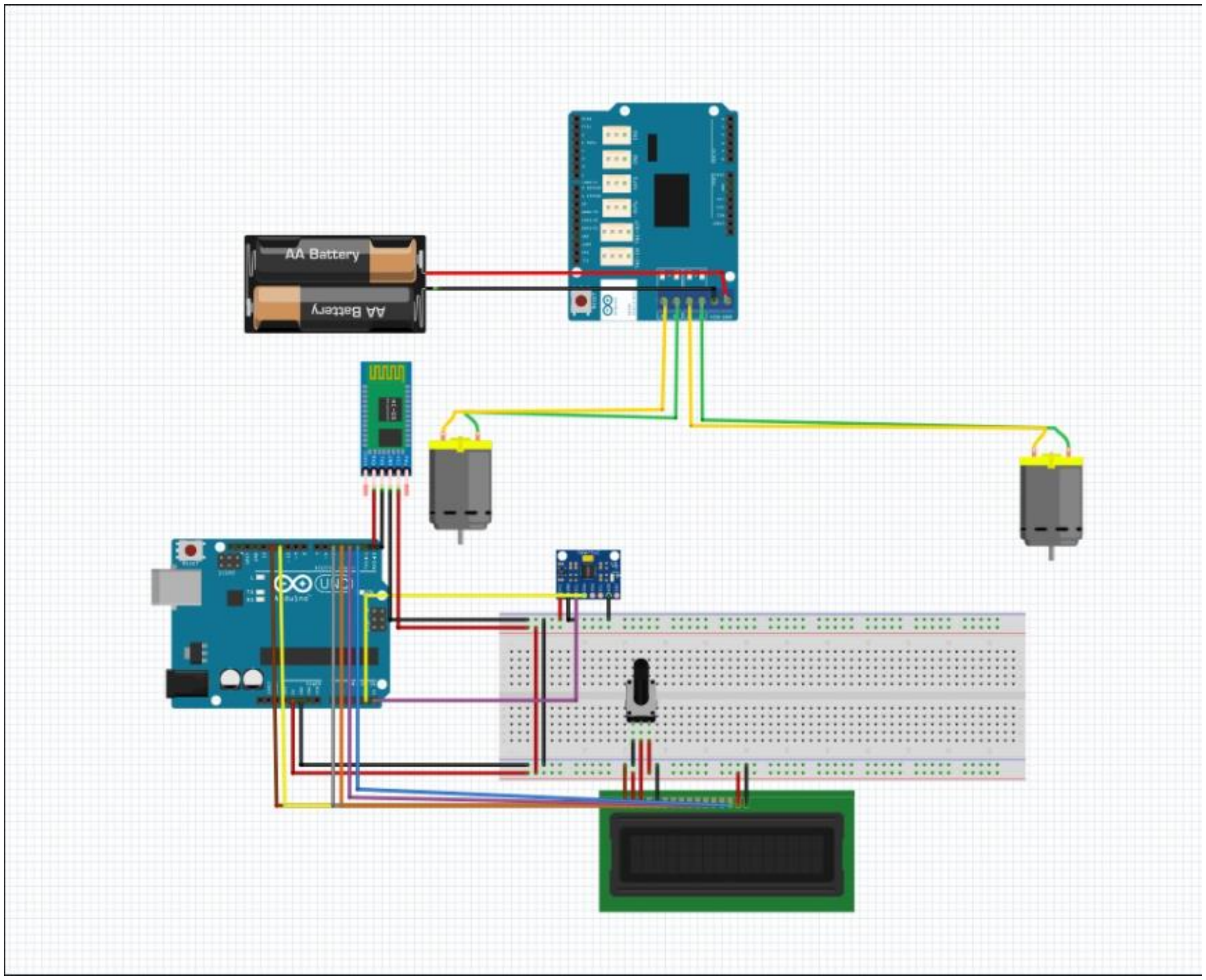
- The project's mechanical component consists of three wooden plates that are mounted on top of one another using four threaded metal posts. The first two plates are separated by 15 cm, and the second and third plates are separated by 10 cm, and the project is powered by two dc motors with a maximum torque of 8.8 kg/cm and a Cytron dual channel motor driver and two 80mm diameter wheels. Nuts, and washers are used to attach the plates on top of one another with the pillars.

- For the electromechanical section, the DC motors are driven by a dual-channel Cytron motor driver that can handle 10 Amps of current and a 30-volt voltage differential. The power for this device is supplied by a 9-volt, 3-amp adapter, and the PWM and direction channels are controlled by an Arduino UNO.



2.2 Electrical design

The electrical section of the projects includes LCD, MPU 6050, Potentiometer, Bluetooth module, Motor driver and Arduino UNO where the driver should be mounted on Arduino in the schematic but in reality, it's connected like normal motor driver.



1.Circuit Schematic

2.3 Control

2.3.1. Modeling

- Mathematical models of the mechanical system of the self-balancing robot yield the transfer function. A TWBMR operates on the same mechanical principle as an inverted pendulum. When the balancing force is absent, the pendulum will fall. The notion leads to the conclusion that the system is unstable. In contrast, the pendulum position is steady when it is normal or unreversed**.**

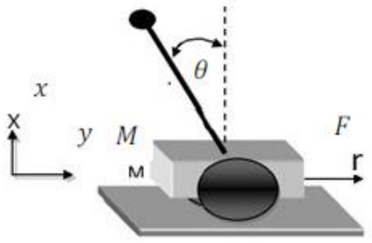


Figure 2

- The dynamic Newton II law equation for TWBMR is used. The mechanism's Free Body Diagram (FBD) is depicted in Figure 2. The pendulum's slope has a number of angles that combine the two components of force with the horizontal and vertical directions.

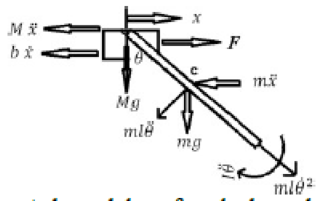
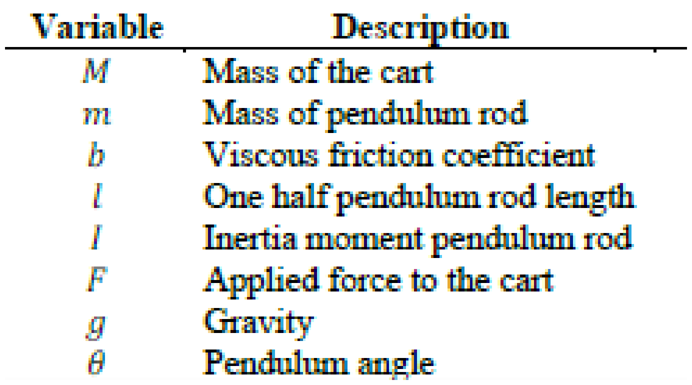


Figure 2 Inverted pendulum free body and inertia force diagram at the stable equilibrium position



Variable	Description
M	Mass of the cart
m	Mass of pendulum rod
b	Viscous friction coefficient
l	One half pendulum rod length
I	Inertia moment pendulum rod
F	Applied force to the cart
g	Gravity
θ	Pendulum angle

- $\Sigma f_x = 0$
  $F - b\dot{x} - M\dot{x} - m\dot{x} - ml\ddot{\Theta}cos(\theta) + ml(\theta')^2 sin(\theta) = 0$ (1)
- $\Sigma f_y = 0$
  $Mg + mg - ml\ddot{\Theta}\sin(\theta) - ml(\theta')^2 sin(\theta) = 0$ (2)
- $\Sigma M_A = 0$
  $Mglsin(\theta) + I\ddot{\Theta} + ml^2\ddot{\Theta} + ml\ddot{x}cos(\theta) = 0$ (3)

Combining equations (1), (2), and (3), the force equation in both the horizontal and vertical axes is produced.

- $(I + ml^2)\,\theta + mglsin(\theta) = -ml\ddot{x}cos(\theta)$ (4)
- $(M+m)\,\ddot{x} + b\dot{x} + ml\ddot{\Theta}\cos(\theta) - ml(\theta')^2 sin(\theta) = F$ *(5)*

Two linear equations of the transfer function, with q =π, are equations (4) and (5). Using the assumption $\theta = \pi + \phi$

- $cos\theta = -1$ (6)
- $sin\theta = -\phi$ (7)
- $d^2/dt^2 = 0$ (8)

Following the non-linear method of equations (6), (7), and (8), we then derived two versions of the motion equations. The input is represented by the U value,

- $(I + ml^2)\,\ddot{\Phi} - mgl\Phi = ml\ddot{x}$ (9)
- $(M+m)\,\ddot{x} + b\dot{x} + ml\ddot{\Phi} = F$ (10)

By applying Laplace transform on equations (9) and (10),

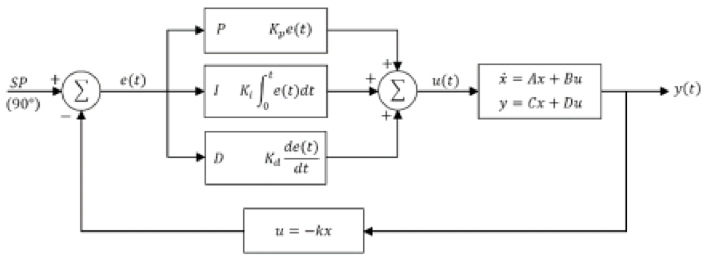
- $(I + ml^2)\,\ddot{\Phi}(s)\, s^2 - mgl\Phi(s) = ml\ddot{x}(s)\, s^2$ (11)
- $(M+m)\,\ddot{x}(s)\, s^2 + b\dot{x}(s)\, s + ml\,\ddot{\Phi}(s)\, s^2 = U(s)$ (12)

we yielded the transfer function for TWBMR from the equations (11) and (12).

$$\frac{\Phi(s)}{U(s)} = \frac{\frac{ml}{q}s}{s^3 + \frac{b(I+ml^2)}{q}s^2 - \frac{(M+m)mgl}{q}s - \frac{bmgl}{q}}, \qquad (13)$$

where $q = [(M+m)(I+ml^2) - (ml)^2]$.

- **Block Diagram representation**



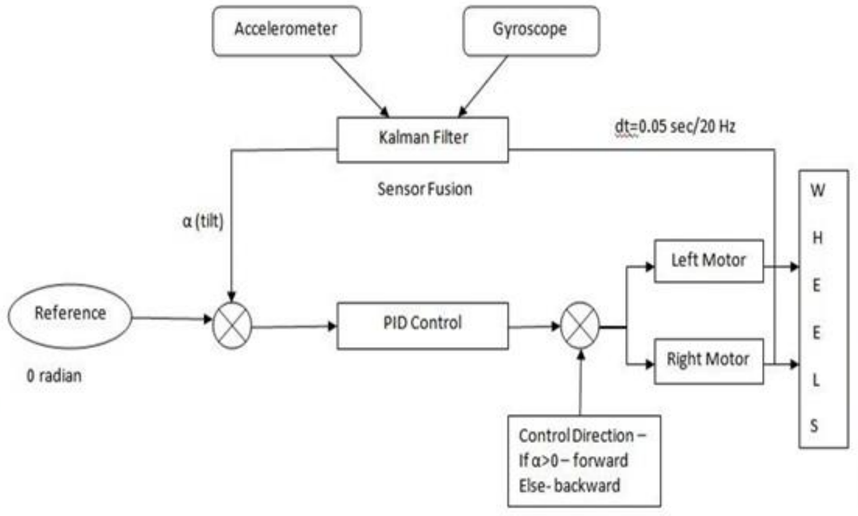
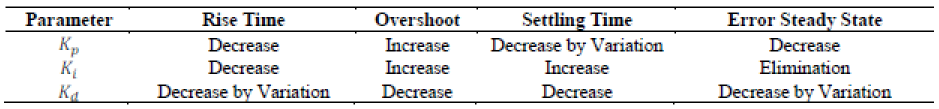
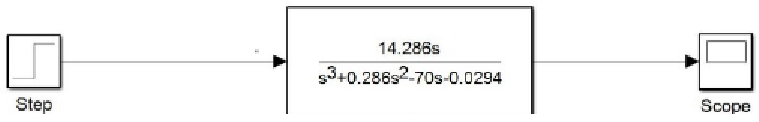


Table 2 Characteristics of PID parameters



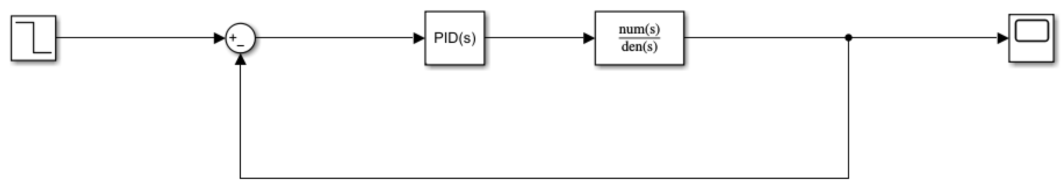
Parameter	Rise Time	Overshoot	Settling Time	Error Steady State
K_p	Decrease	Increase	Decrease by Variation	Decrease
K_i	Decrease	Increase	Increase	Elimination
K_d	Decrease by Variation	Decrease	Decrease	Decrease by Variation



`

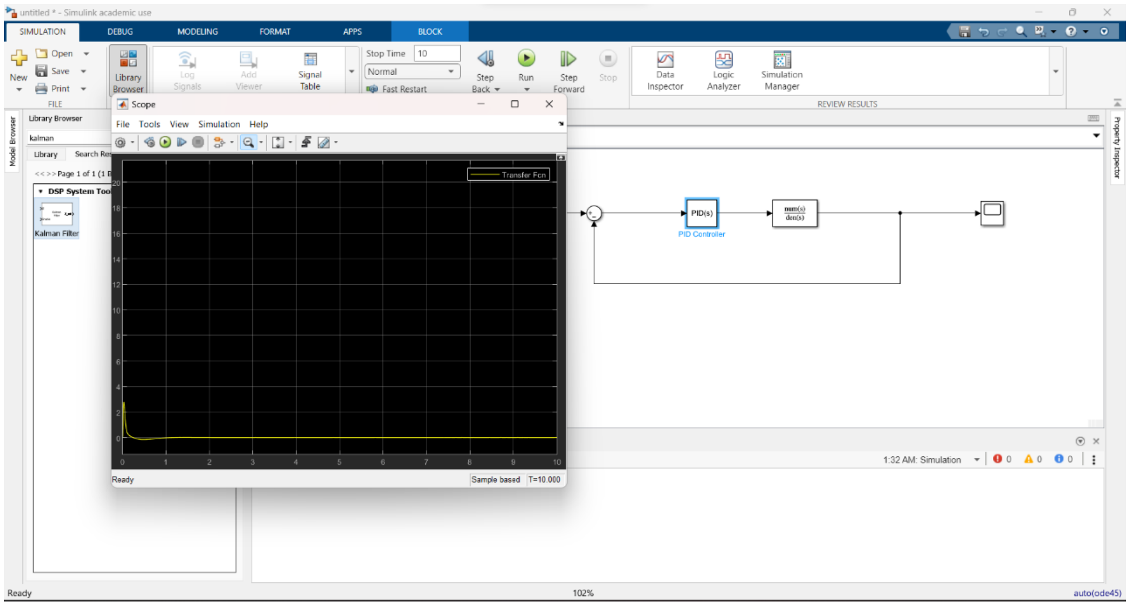
2.3.2. Controller Design

- The controller used in our project is PID controller where we tune the K_p, K_i, K_d to get a desired response.

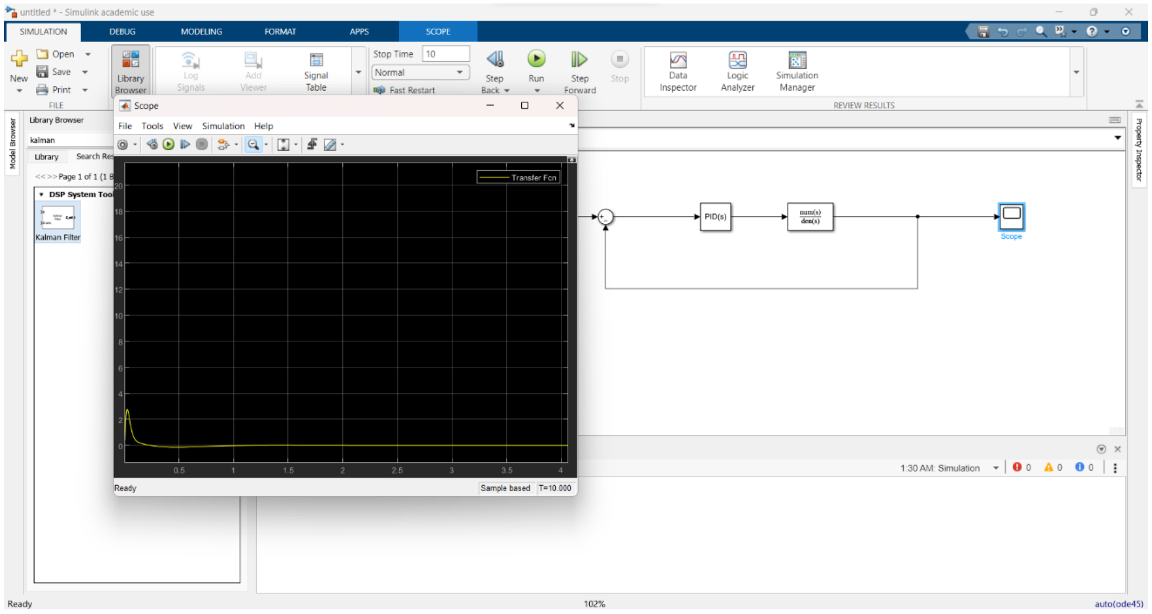


- In the real project we used Kalman filter which is not included in the simulation due to its complexity.

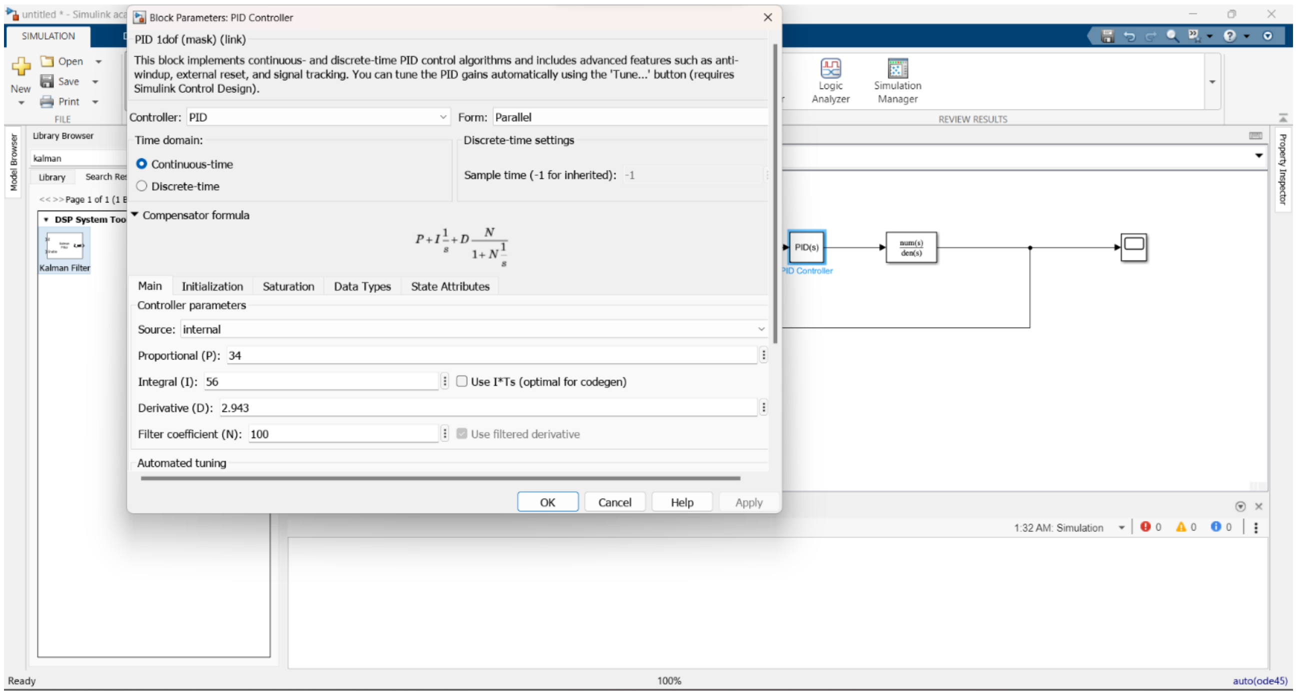
- The simulation was done using Simulink with step input of initial value of 10 and step size of 0.01 and final value of zero and sample time of 0 (continuous).



- The simulation was done again using Simulink with step input of initial value of 10 and step size of 0.01 and final value of zero and sample time of 0.005 (discrete).



- The K values used in the simulations were $K_p = 34$, $K_i = 56$, $K_d = 2.943$.
- Note: further tunings may output a better response but this one was acceptable.
- The real K values used in the project were $K_p = 84$, $K_i = 150$, $K_d = 2.9$.


Block Parameters: PID Controller

PID 1dof (mask) (link)

This block implements continuous- and discrete-time PID control algorithms and includes advanced features such as anti-windup, external reset, and signal tracking. You can tune the PID gains automatically using the 'Tune...' button (requires Simulink Control Design).

Controller: PID   Form: Parallel

Time domain: Continuous-time

Discrete-time settings: Sample time (-1 for inherited): -1

Compensator formula

$$P + I\frac{1}{s} + D\frac{N}{1+N\frac{1}{s}}$$

Main | Initialization | Saturation | Data Types | State Attributes

Controller parameters

Source: internal

Proportional (P): 34

Integral (I): 56   ☐ Use I*Ts (optimal for codegen)

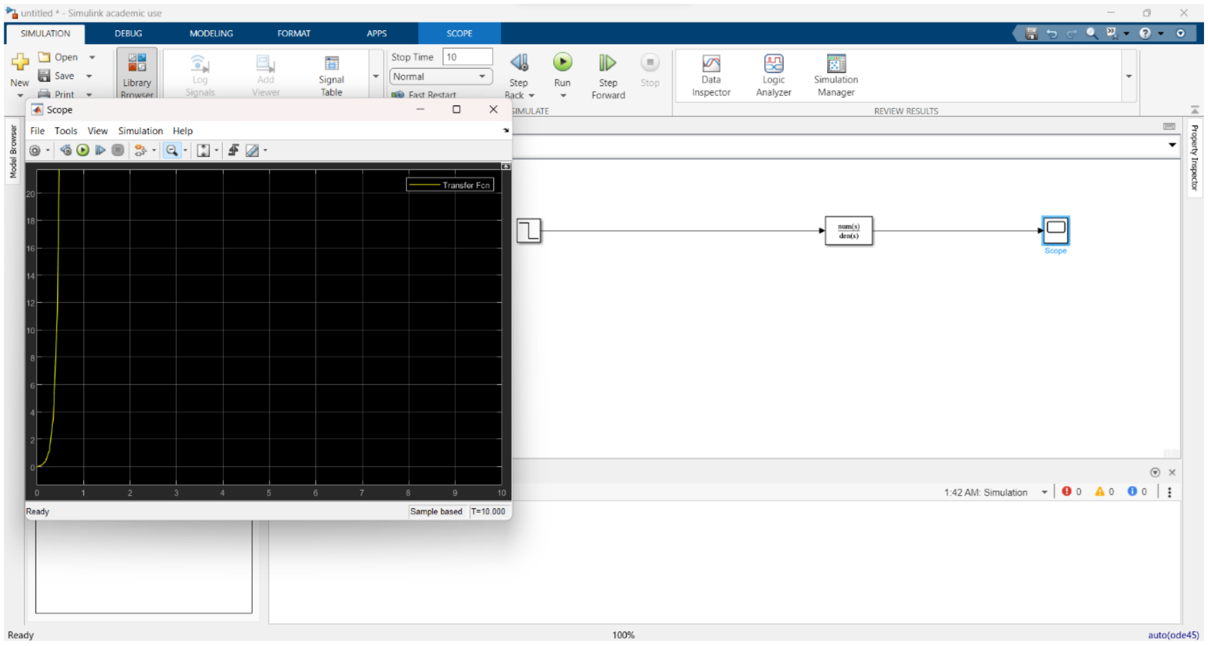
Derivative (D): 2.943

Filter coefficient (N): 100   ☑ Use filtered derivative

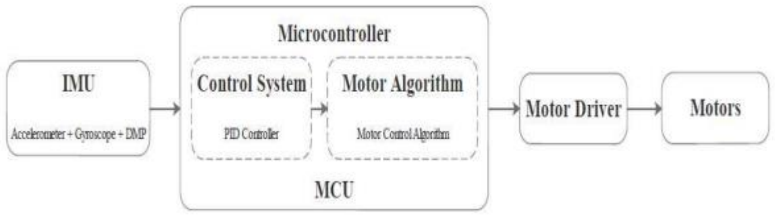
Automated tuning

OK | Cancel | Help | Apply

- The response without controller.

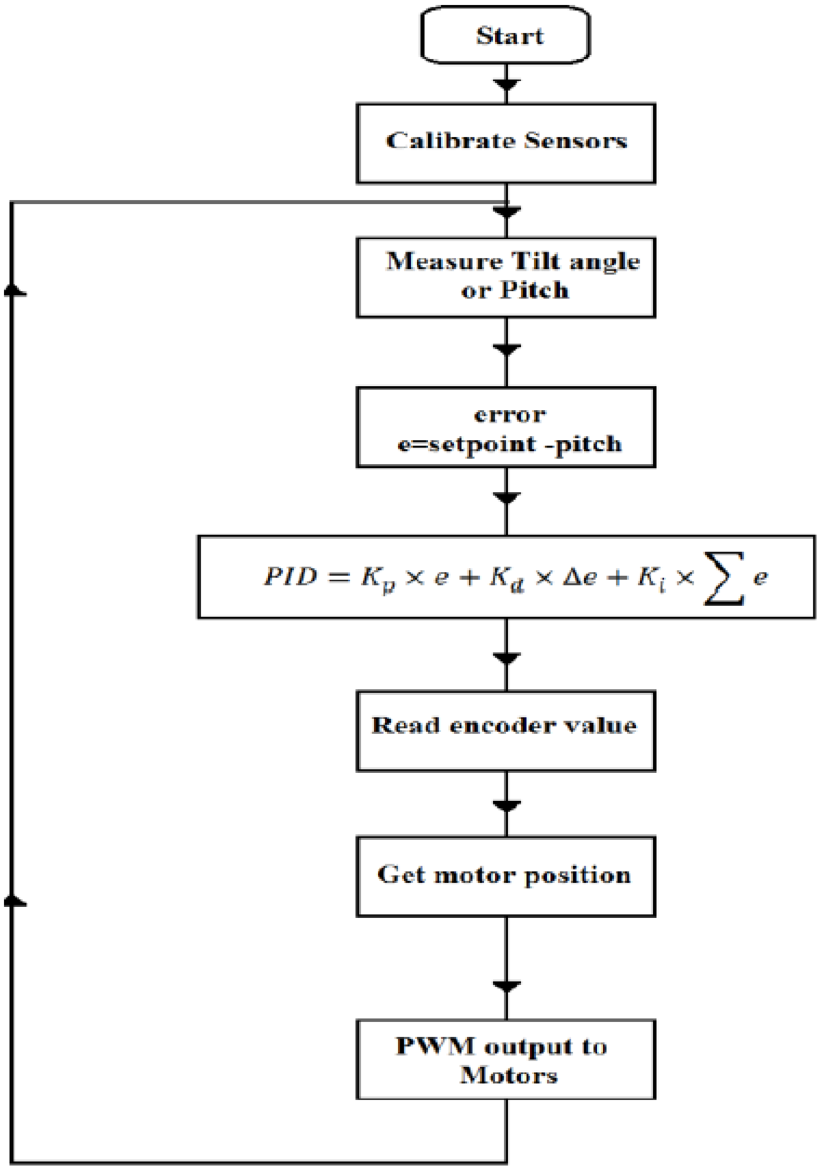


1.Response of the system



2.block diagram

2.4 Programming



3.Flowchart of the program

3. Design Evaluation

We had to change those two parts because the initial hardware implementation in our project was extremely unstable due to the height of the project was out of proportion to the motor's torque because it produces a moment greater than the motor's maximum torque so the system will be unstable no matter what PID numbers we choose and since the standard L298N can only draw 2 Amps as maximum current, we had to utilize the Cytron dual Channel Motor Driver to supply extra power to the new motors.

Thus, after improving those two components, the reaction improved significantly, and after a few tuning adjustments, the system became more reliable and responded better on the Arduino serial plotter.

Due to the inaccuracy of the IMU module sensor readings and the height of the robot, the system became stable with margin errors after some tunings. Therefore, if we changed the IMU module and reduced the height of the robot, the response would be better.

After adjusting the PID parameters to obtain a better actual response, the response of the real system is very comparable to that of the simulation. The system performance is acceptable as long as no large external force is applied on it.

4. Appendix

- https://www.researchgate.net/publication/332739123_Design_of_a_Two-Wheel_Self-Balancing_Robot_with_the_Implementation_of_a_Novel_State_Feedback_for_PID_Controller_using_On-Board_State_Estimation_Algorithm

- B. Y. Suprapto, D. Amri, and S. Dwijayanti, "Comparison of Control Methods PD, PI, and PID on Two Wheeled Self Balancing Robot," in *the International Conference on Electrical Engineering, Computer Science and Informatics*, 2014, pp. 67–71.

- https://www.researchgate.net/publication/323258475_DESIGN_AND_IMPLEMENTATION_OF_A_SELF-BALANCING_ROBOT