IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, March 2026
Impact Factor: 8.2

Self-Balancing Robot

Mr. Y. V. S. Durga Prasad, V. Charan, B. Ajay, C. Manaswini
Associate Professor, ECE
Department of Electronics and Communication Engineering
ACE Engineering College, Hyderabad, India

Abstract: *In this paper, a self-balancing robot using an in- verted pendulum principle with an Arduino microcontroller is proposed. Self-balancing robots are widely used to study control systems, robotics, and embedded system integration. The robot balances by constantly monitoring its orientation and taking corrective action to maintain stability.*

The robot is equipped with an MPU6050 Inertial Measure- ment Unit (IMU), which is a sensor that combines a three- axis accelerometer and a three-axis gyroscope to sense the tilt angle, angular velocity, and linear acceleration of a body. The data is then processed by an Arduino microcontroller to calculate the orientation of the robot. To maintain stability in the robot, a Proportional-Integral-Derivative (PID) control method is employed.

The system is a closed-loop system in which all processes are carried out in real time. The results of the experiment show that the robot is able to maintain stability and balance in an upright position..

Keywords: Self Balancing Robot, Arduino, PID Controller, MPU6050, Robotics

I. INTRODUCTION

Self-balancing robots are dynamic robotic devices that are able to maintain their vertical position using feedback control mechanisms. Self-balancing robots are often represented by inverted pendulums, which are naturally unstable and require prompt control actions to maintain stability. Self-balancing robots are becoming increasingly popular, driven by the success of products such as the Segway and their use for educational and research purposes.

Two-Wheeled Balancing Robots: An Interesting Case ”Two- wheeled balancing robots are interesting because they must detect tilt and control the wheels to correct the imbalance. In other words, the robot must detect tilt and drive the wheels forward to correct for forward tilt and restore the center of mass. Similarly, the robot must drive the wheels backward to correct for backward tilt.”

Self-balancing robots are used extensively for research and educational purposes in the domain of robotics. In the litera- ture, several sensor fusion and control algorithms are used for balancing robots. Some popular algorithms used are Kalman filtering and other sophisticated control algorithms other than PID control. In our research, we are using a popular approach for balancing robots using MPU6050 and PID control, as shown in other research works.[1]

II. LITERATURE REVIEW

There are a number of methods that researchers have suggested for increasing stability and performance in self-balancing robots. The first self-balancing robot utilized ac- celerometers to calculate tilt by gravity. However, accelerom- eters are not very effective in a moving system. The addition of a gyroscope would improve performance, as it would be possible to calculate angular velocity and integrate it to obtain a short-term estimate of tilt. The combination of data from accelerometers and gyroscopes using a filter would provide a more accurate tilt measurement. Several methods have been compared for use in self- balancing robots. While linear quadratic regulators and fuzzy logic controllers have been used, a Proportional-Integral- Derivative (PID) controller is also a popular choice. Sani et al. have successfully used a PID controller in conjunction with a complementary filter to balance a robot using an Arduino platform. The method proposed in this paper is also a part of this trend, using a PID loop to control tilt error.[2][3]

DOI: 10.48175/IJARSCT-32741

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, March 2026
Impact Factor: 8.2

III. BLOCK DIAGRAM

BLOCK DIAGRAM

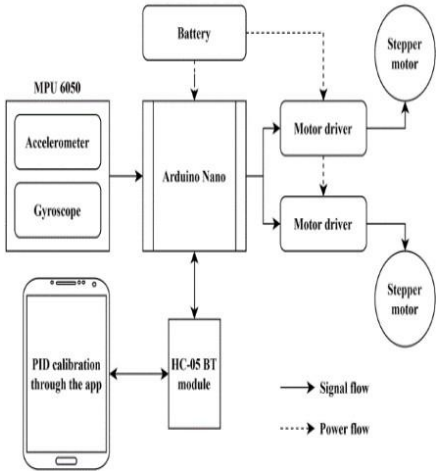

Fig. 1. Block Diagram of Self Balancing Robot

The block diagram illustrates the interaction between system components. The MPU6050 sensor measures tilt angle and angular velocity and sends data to the Arduino microcontroller through the I2C interface. The Arduino processes the data using a PID control algorithm and sends motor control signals to the motor driver. The motor driver controls the DC motors (wheels) to move the robot. This high-level architecture is similar to other self-balancing robot designs.

IV. SYSTEM ARCHITECTURE

The system architecture has three main components, namely:

A. Sensing System

The sensing system consists of an MPU 6050 sensor, which has both an accelerometer and a gyroscope. An accelerometer measures total acceleration, including both gravity and motion. This can be used to determine the tilt of the robot. A gyroscope measures angular velocities, which can be integrated to obtain the robot's orientation. By combining the data from both, the robot can obtain an accurate value for its pitch angle.

B. Processing System

The processing system consists of an Arduino UNO. This reads the data from the sensor, calibrates it, and then calculates the tilt angle. Then, the Arduino UNO performs a PID control function, comparing the desired angle, which is upright, to the actual angle. This is then used to determine the control.

C. Actuation System

The actuation system consists of the motor driver and the motors, which are attached to the wheels of the robot. This consists of DC gear motors, which are capable of producing enough torque to move the robot base quickly to balance the robot. When the robot needs to move, the wheels spin, allowing it to move while balancing itself.[4]

V. HARDWARE COMPONENTS

The hardware components of the robot are as follows:

A. Arduino UNO

The Arduino UNO is the brain of the robot, running the control software. It reads the IMU sensor data, performs the PID computation, and sends PWM control to the motors.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, March 2026
Impact Factor: 8.2

B. MPU6050

The MPU6050 is an IMU sensor, consisting of 3-axis accelerometers and 3-axis gyroscopes. It communicates with the Arduino via I2C protocol. Sensor fusion, or rather a

C. Motor Driver

A dual motor driver IC, such as the L298N or similar, is employed to control the DC motors. It accepts PWM and direction inputs from the Arduino and powers the motors via the battery.

D. Motors and Wheels

Two DC gear motors, rated for 150 RPM, are attached to the wheels of the robot. High torque and moderate speed are necessary for balancing. The wheels are attached to the shafts of the motors for easier construction.

VI. HARDWARE SETUP

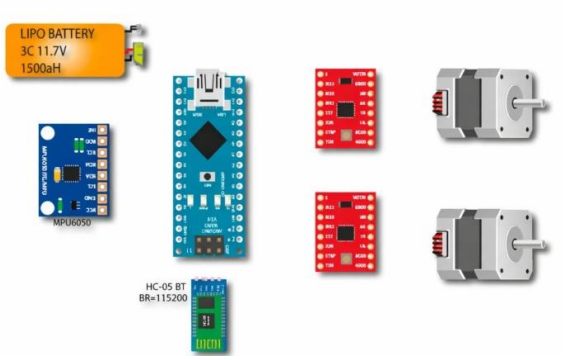

Fig. 2. Hardware Components Used

The hardware setup consists of an Arduino, MPU6050 sensor, motor driver, Bluetooth module (optional), motors, and a power supply in the form of a battery. Figure 2 shows the hardware setup assembled and ready for use on a chassis. The MPU6050 is placed at the center of mass of the robot, with its axes aligned in the forward direction of the robot. The hardware is fixed to a frame to prevent any movement of the sensor relative to the robot body.[5]

VII. MATHEMATICAL MODEL

The two-wheeled robot is considered an inverted pendulum on wheels. The equation of motion about the upright position is:

$$\tau = I\, d2\theta/dt2 + mgL \sin(\theta) \quad (1)$$

This is a nonlinear equation showing the effect of torque on the angular acceleration of the system. In control theory, we often linearize this equation under the assumption of small θ and thus $\sin(\theta) \approx \theta$.

VIII. PID CONTROLLER

The PID controller calculates the control signal u(t) based on the difference between the desired value, or 0 degrees, and the measured value, or the tilt angle, given by

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt} \quad (2)$$

where Kp, Ki, Kd are the proportional, integral, and derivative gains, respectively. In this implementation, the e(t) is the current tilt angle. The proportional term reacts to the tilt, the derivative term reacts to the rate of change of the tilt, and the integral term reacts to biases in the tilt.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, March 2026
Impact Factor: 8.2

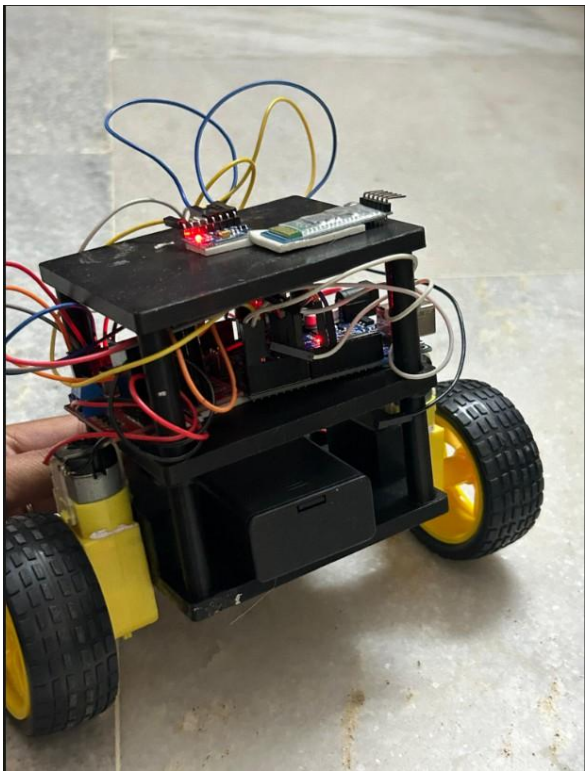

Fig. 3. Self Balancing Robot Prototype

IX. WORKING PRINCIPLE

The robot is operated using continuous feedback control. The MPU6050 sensor is used to continuously sense the tilt angle and angular velocity. The sensed data is fused to estimate the tilt angle of the robot. The Arduino measures the sensed data and compares it with the desired angle (zero degrees), which is the upright position. The error is calculated. This error is passed to the PID controller, which generates control signals. If there is a positive error, it means the robot is leaning forward. The PID controller generates a positive signal, and the wheels are moved forward to balance the robot. The center of gravity is moved to the center of the base. If there is a negative error, it means the robot is leaning backward. The PID controller generates a negative signal, and the wheels are moved backward. Figure 3 shows the prototype of the robot. The above concept has been shown in literature.

X. CONTROL ALGORITHM

1) Initialize sensors and motors
2) Calibrate IMU and set initial setpoint (angle = 0)
3) Read accelerometer and gyroscope values of MPU6050
4) Estimate tilt angle
5) Calculate error value using formula: e(t) = (Setpoint − Angle measurement)
6) Calculate PID value using error value
7) Generate PWM signals proportional to PID value
8) Apply control signals to motors
9) Repeat step 3

DOI: 10.48175/IJARSCT-32741

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, March 2026
Impact Factor: 8.2

This loop will ensure that any angle not equal to 0 is corrected immediately.

XI. FLOWCHART

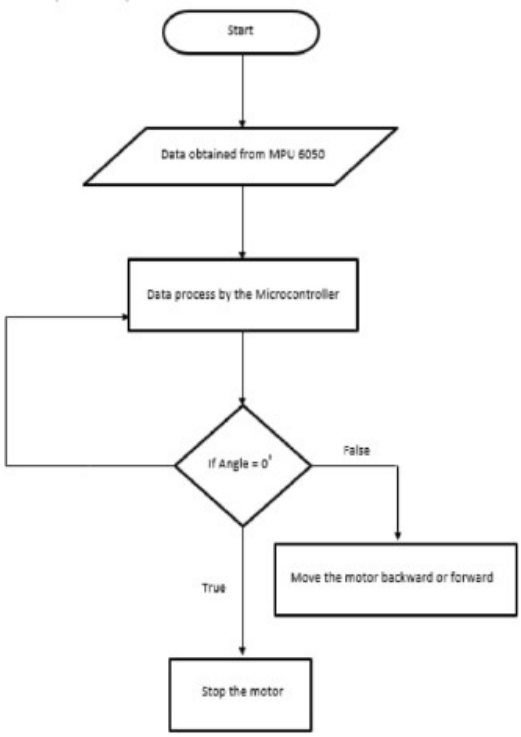

Fig. 4. Control Flow of Robot

The flowchart shown in Figure 4 demonstrates the process of robot balance control. The process involves reading the sensors, calculating the PID control, and adjusting the motors. This process repeats continuously.

XII. EXPERIMENTAL RESULTS

The robot was placed on a flat surface. Figure 7 is a sample plot for the robot's tilt angle over time. If the robot is pushed slightly, the tilt angle will change, and the control loop will make the robot's wheels act to reduce the tilt angle to zero. In the experiment, the tilt angle reached a maximum of 10°, and the robot returned to vertical position in less than 1 second. The PID controller has been able to minimize oscillations and increase stability.

The results obtained from the experiment show stable balancing performance. The performance metrics include the settling time, which refers to the time the robot takes to settle after being subjected to a push, and the maximum tilt, which refers to the maximum angle the robot tilts. This validates the performance of the control strategy, which works as expected, just like the results obtained in the related literature.[7]

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, March 2026
Impact Factor: 8.2

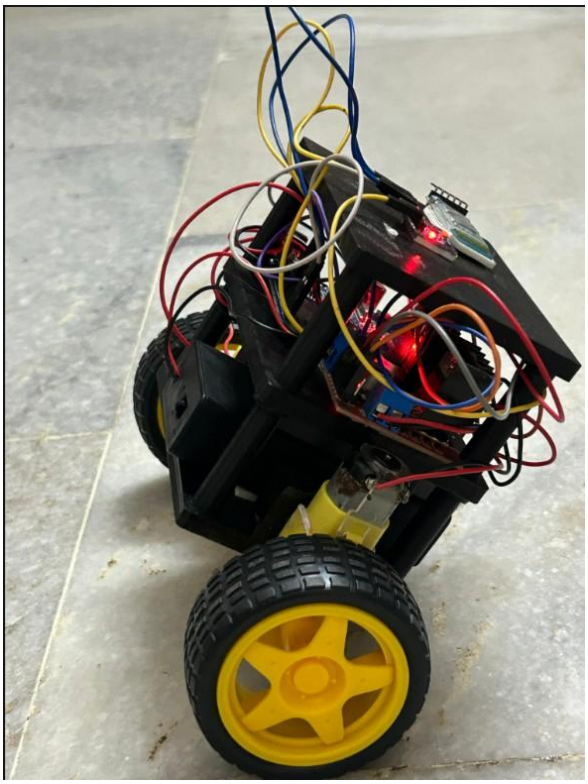

Fig. 5. Tilt Angle vs Time during balancing

XIII. ADVANTAGES

- Low cost system using common components.
- Simple design and assembly.
- Capable of real-time balancing with minimal latency.
- Expandable platform for sensors and wireless control.

XIV. APPLICATIONS

- Personal transportation devices (e.g., Segway-like vehi- cles).
- Educational tool for control systems and robotics.
- Autonomous mobile robots and delivery vehicles.
- Demonstrations of feedback control in engineering courses.

XV. FUTURE SCOPE

Future work may involve using more sophisticated filtering (e.g., using a Kalman filter to improve angle estimation), incorporating obstacle detection sensors for navigation, and using a more powerful controller (e.g., ESP32 for increased update rates and WiFi functionality).

XVI. CONCLUSION

In this regard, a self-balancing robot on two wheels was designed and tested to demonstrate basic control theory and its integration with embedded systems. The MPU6050 sensor, Arduino board, PID control algorithm, and motor control

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 9, March 2026 Impact Factor: 8.2

circuits all work together to ensure the robot balances on its two wheels. This project is an example of the application of control theory in robotics, and the design can be expanded further.

REFERENCES

[1] H. Hellman, "Two-Wheeled Self-Balancing Robot," Master's Thesis, 2015. Available: https://www.diva-portal.org/smash/get/diva2:916184/FULLTEXT01.pdf
[2] "Self-Balancing Robot With PID Controller," ResearchGate, 2023.
Available: https://www.researchgate.net/publication/369833840
[3] C. V. Samak and T. V. Samak, "Design of a Two-Wheel Self-Balancing Robot with Novel PID State Feedback," International Journal of Robotics Research and Development, 2018.
[4] R. M. Shihab et al., "Development and Prototyping of a Two-Wheeled Self-Balancing Robot," Maldives Journal of Engineering and Technol- ogy, 2024.
[5] S. Gupta, "Self-Balancing Mobile Robot with Bluetooth Control," MDPI Robotics, 2025. Available: https://www.mdpi.com/2673-4052/6/3/42
[6] G. Xin, "A Unified Control Framework for Self-Balancing Robots," Sensors, MDPI, 2025. Available: https://www.mdpi.com/1424- 8220/25/23/7144
[7] "Real-Time Implementation of a Self-Balancing Robot Using PID Controller," International Journal of Engineering Research and Technology (IJERT), 2025. Available: https://www.ijert.org/real-time- implementation-of-a-self-balancing-robot-using-pid-controller