Title: Self-Balancing Car with Bluetooth Control
Submitted by: Ronit Shrestha
Submitted to: RAC Club
Department of Computer Engineering

1. Introduction of Your Project

The self-balancing car with Bluetooth control is a two-wheeled robotic system that maintains its balance using a feedback control system. It utilizes sensors like the MPU6050 (which combines a gyroscope and accelerometer) to detect the tilt angle of the car, and adjusts the motors to balance itself in real time using a PID (Proportional-Integral-Derivative) control algorithm. In addition, the system is integrated with Bluetooth functionality, enabling remote control via a smartphone, thus combining automation with manual override.

2. Function or Problem That Your Project Solves

This project addresses the challenge of creating a dynamically stable vehicle that balances itself, similar to real-world applications like Segways or self-balancing personal transporters. It solves the problem of keeping a two-wheeled system upright without external support. It also provides remote control capability via Bluetooth, allowing the user to interact with and control the system in a flexible and wireless manner.

3. Components, Code and Mechanism Used in the Project and Their Contribution/Function

A. Hardware Components:

1. **Arduino UNO** – The main controller that processes sensor data and drives the motors.
2. **MPU6050 Sensor** – Provides real-time orientation (pitch angle) using an accelerometer and gyroscope.
3. **L298N Motor Driver** – Controls the direction and speed of the motors based on Arduino signals.
4. **DC Motors with Wheels** – Provide movement and balancing torque.
5. **Bluetooth Module HC-05** – Enables wireless communication with a mobile phone for remote control.
6. **Battery Pack (12V)** – Powers the entire system.

B. Software and Code:

- The code is written in Arduino IDE using C++.

- The MPU6050 outputs are filtered using a **Kalman Filter** to reduce noise and provide accurate angle measurement.
- A **PID algorithm** is implemented to maintain balance by adjusting motor output based on the tilt angle.
- Bluetooth serial commands are interpreted in code to allow forward, backward, left, and right movement while maintaining balance.

C. Mechanism:

- The MPU6050 continuously measures the pitch of the car.
- The Arduino uses the Kalman filter to get a stable angle reading.
- The PID algorithm calculates an error between the target (usually 0° upright) and current pitch, and sends corrective signals to the motors.
- The L298N driver changes motor direction and speed accordingly.
- Bluetooth commands are received and override motor directions while maintaining balance.

4. Future Scope and Improvement Side

- **Obstacle Detection:** Integrating ultrasonic or infrared sensors to avoid collisions.
- **Autonomous Navigation:** Adding GPS or vision modules for self-driving capabilities.
- **Advanced Balancing Algorithm:** Implementing AI/ML-based adaptive PID or neural network control.
- **Mobile App Integration:** A dedicated smartphone app for smoother control, real-time monitoring, and tuning.
- **Compact Hardware:** Reducing the size and weight for better portability and agility.
- **Battery Optimization:** Improving battery life with better power management systems.

5. Conclusion

The self-balancing car with Bluetooth is a demonstration of control systems, real-time sensor data processing, and embedded system integration. It showcases the application of PID control for dynamic balancing and the use of wireless communication for user interaction. The project not only serves as a great learning tool but also opens pathways for building more advanced robotics systems and autonomous vehicles. With further improvements, it can be developed into a reliable and intelligent transport or robotics platform.