Journal of Electronics and Electrical Engineering
https://ojs.wiserpub.com/index.php/JEEE

UNIVERSAL WISER PUBLISHER

Research Article

Kalman Filter-Based PID and Nonlinear Counter-Steering for Low-Cost Self-Balancing

Chung-Hsing Chao[1], Jenn-Jong Shieh[2*], Xiang-Jun Zhang[2]

[1] Department of Intelligent Vehicle and Energy, Minth University of Science and Technology, Hsinchu 30743, Taiwan
[2] Department of Electrical Engineering, Feng Chia University, Taichung 40724, Taiwan
[*] Correspondence: jjshieh@fcu.edu.tw

Received: 13 April 2026; **Revised:** 27 April 2026; **Accepted:** 8 May 2026; **Published:** 27 May 2026

Abstract: Two-wheeled vehicles present a significant stability challenge, particularly at low or zero velocity. This paper presents the design, implementation, and validation of a low-cost, open-source self-balancing electric motorcycle prototype that addresses this challenge. The system is based on the inverted pendulum model, employing a Proportional-Integral-Derivative (PID) controller for longitudinal (pitch) stability and a novel counter-steering strategy for lateral (roll) stability. An Inertial Measurement Unit (IMU) provides attitude data, which is processed by a Kalman filter to yield a robust and accurate estimate of the vehicle's tilt angle by effectively fusing noisy accelerometer data and drifting gyroscope data. The entire system is managed by an STM32 microcontroller, demonstrating that advanced control can be implemented on accessible hardware. Experimental results validate the effectiveness of the proposed controller, showing rapid balance recovery within 1–2 s from external disturbances of up to ±5° with less than 15% overshoot. This work not only provides a practical implementation of a self-balancing system but also serves as a valuable and replicable platform for further research in intelligent transportation and robotics education.

***Keywords*:** two-wheeled vehicles, self-balancing electric motorcycle, Proportional-Integral Derivative (PID), Inertial Measurement Unit (IMU), Kalman filter

1. Introduction

The increasing demand for efficient and sustainable personal mobility solutions in urban environments has spurred significant research into advanced vehicle technologies. Among these, self-balancing two-wheeled vehicles represent a compelling area of innovation, promising enhanced safety, maneuverability, and convenience, particularly at low speeds or during standstill conditions where conventional motorcycles become unstable [1, 2]. The fundamental challenge in designing such vehicles lies in overcoming their intrinsic dynamic instability, which is often modeled as an under-actuated, nonlinear inverted pendulum system. While gyroscopic effects provide inherent stability at higher speeds, this stability diminishes significantly as speed decreases, necessitating active control mechanisms to maintain equilibrium [3].

Commercial ventures, such as the Segway Personal Transporter and conceptual designs like the Lit Motors C-1, have demonstrated the practical feasibility and potential of self-balancing technology. However, these solutions typically rely on proprietary technologies, complex mechanical designs (e.g., control gyroscopes), and high manufacturing costs, which collectively create substantial barriers to entry for broader academic research, educational initiatives, and open-source

DOI: https://doi.org/10.37256/jeee.5120269976

development. This proprietary nature often limits the dissemination of knowledge and the ability for researchers and enthusiasts to replicate and build upon existing advancements.

In response to these limitations, there is a growing interest in developing accessible, low-cost, and open-source platforms for self-balancing systems. Such platforms are invaluable for fostering innovation, facilitating hands-on learning in control systems and robotics, and enabling rapid prototyping of new control strategies. The Proportional-Integral-Derivative (PID) controller, a cornerstone of classical control theory, remains a widely adopted and highly effective method for stabilizing dynamic systems due to its simplicity, robustness, and ease of implementation [4, 5]. Its successful application, however, is critically dependent on accurate and reliable state estimation.

Inertial Measurement Units (IMUs), comprising accelerometers and gyroscopes, are indispensable for real-time attitude determination in dynamic systems. Nevertheless, raw IMU data presents inherent challenges: accelerometers are susceptible to high-frequency noise from vibrations and external accelerations, while gyroscopes suffer from low-frequency drift due to integration errors over time [6]. To overcome these limitations, optimal state estimators such as the Kalman filter are employed. The Kalman filter is particularly well-suited for fusing complementary sensor data, providing a robust and accurate estimation of the system's state by effectively mitigating noise and drifting [7–9].

While the longitudinal (pitch) control of self-balancing vehicles, often based on the "chasing the center of gravity" principle, is a well-established control problem, lateral (roll) control presents a more complex challenge. This study addresses lateral stability by introducing a novel counter-steering mechanism, which mimics the intuitive technique employed by human riders to maintain balance on bicycles and motorcycles. By actively steering the front wheel in the direction of the lean, a corrective torque is generated, effectively restoring lateral equilibrium [10].

This paper details the comprehensive development of a miniature self-balancing electric motorcycle prototype. The design emphasizes an open-source approach, utilizing an STM32F103C8T6 microcontroller as the central processing unit and an MPU6050 IMU for attitude sensing. We present a detailed methodology encompassing the system's dynamic modeling, hardware integration, and the implementation of both a Kalman filter for robust state estimation and a PID controller for precise balance control. Furthermore, we provide a thorough analysis of the system's experimental performance, demonstrating its ability to maintain stability under various conditions and highlighting the efficacy of our proposed control strategies. This work contributes to a fully open-source, affordable, and replicable platform that can serve as an invaluable resource for educational purposes and as a foundation for advanced research in vehicle dynamics, control systems, and intelligent mobility solutions.

2. System architecture and modeling

2.1 *Hardware architecture*

Figure 1 shows the overall control system block diagram of the self-balancing electric motorcycle. The MPU6050 IMU provides raw accelerometer and gyroscope data. A Kalman filter fuses the noisy sensor measurements to produce accurate estimates of the pitch angle θ and roll angle ϕ. The estimated pitch angle is fed to a PID controller that generates the command for the brushless DC (BLDC) drive motor to maintain longitudinal stability. Simultaneously, the estimated roll angle is processed by a nonlinear counter-steering controller that drives the steering servo to achieve lateral stability. All processing is performed on the STM32 microcontroller.

• **Microcontroller (MCU)**: An STM32F103C8T6 serves as the central processing unit of the system. This ARM Cortex-M3-based microcontroller was selected for its adequate computational capability, rich peripheral resources, and suitability for real-time embedded control applications.

• **Inertial Measurement Unit (IMU)**: An MPU6050 integrated sensor provides three-axis accelerometer and three-axis gyroscope measurements. These signals are used as raw sensing inputs for estimating the motorcycle's pitch and roll states.

• **Actuators**: A brushless DC (BLDC) motor equipped with an encoder is employed to drive the rear wheel, providing position and velocity feedback for closed-loop control. This actuator generates the corrective torque required for longitudinal balance. In addition, a high-torque metal gear servo motor is used to regulate the front wheel steering angle, enabling the counter-steering mechanism for lateral stability.

• **Power System**: A 12 V lithium-polymer battery supplies power to the entire platform, including the controller, sensors, and actuators. A dedicated power management module is used to maintain stable voltage regulation and battery protection.

Based on the acquired IMU measurements, the STM32 executes Kalman filter-based sensor fusion and a PID control algorithm to generate corrective commands for the drive motor and steering servo, thereby establishing the core closed-loop balancing framework of the system.

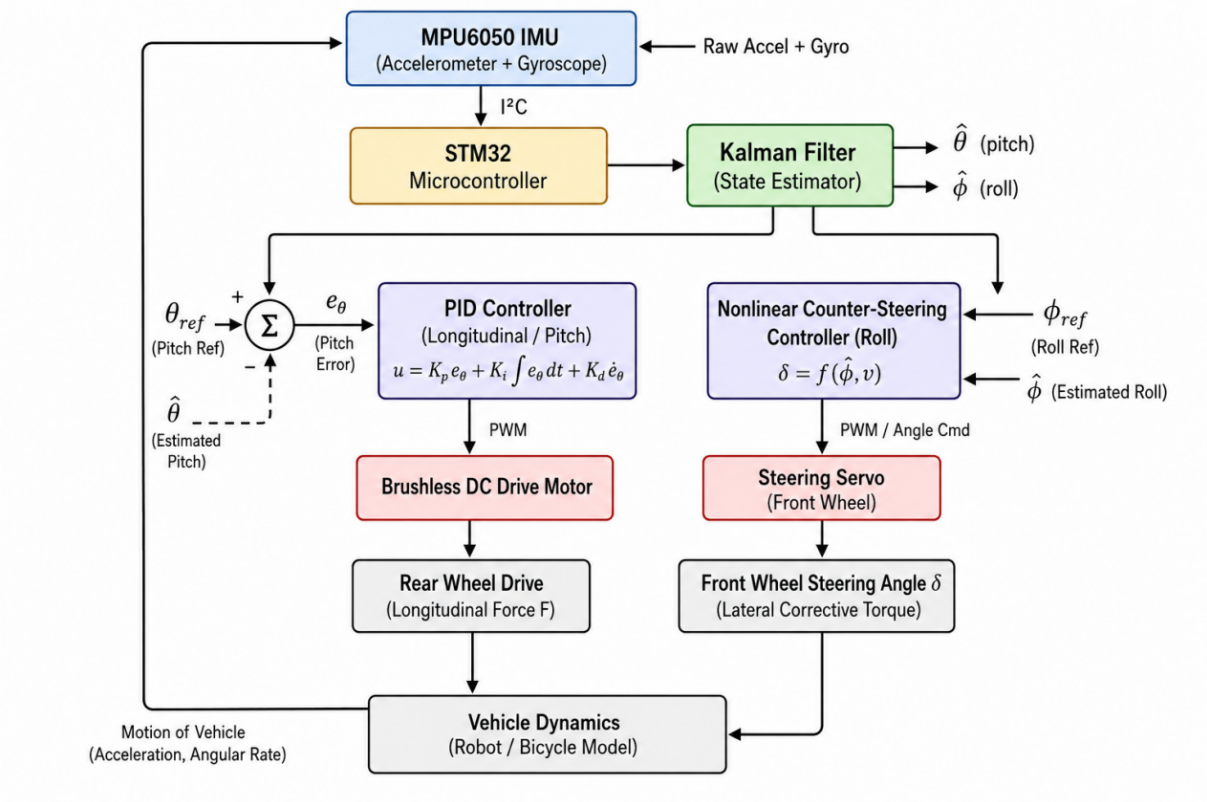

Figure 1. Overall control system block diagram of the self-balancing electric motorcycle

• Kalman filter output utilization

The estimated pitch and roll states obtained from the Kalman filter, namely ($\hat{\theta}$) and ($\hat{\phi}$), are directly fed into the corresponding control loops. Specifically:

($\hat{\theta}$) is used as the feedback state for the longitudinal PID controller.

($\hat{\phi}$) is provided to the nonlinear counter-steering controller for roll stabilization.

• Closed-loop feedback representation

A complete closed-loop feedback structure includes PID control loop that the reference pitch angle (θ_{ref}), the error computation node, and the feedback path using the estimated pitch angle ($\hat{\theta}$).

• Tracking error

The controller operates using the tracking error [$e_\theta(t) = \theta_{\text{ref}} - \hat{\theta}(t)$] which is consistent with the actual implementation of the proposed stabilization system.

The steering command is generated by a high-torque metal-gear servo motor that controls the front wheel steering angle $\delta\backslash\Delta\delta$. In the current implementation, the servo is assumed to track the commanded angle with negligible delay for the purpose of controller design. However, real servomotors exhibit non-ideal dynamic characteristics, including response latency, limited angular velocity, and mechanical backlash.

To quantify these effects, we measured the servo step response under typical operating load. The servo achieves a 10%–90% rise time of approximately 80–120 ms for a 15° step command, corresponding to an effective bandwidth of roughly 3–5 Hz for large-angle corrections. The no-load speed is approximately 0.12–0.15 s/60°, and mechanical backlash is limited to ±1.5°–2° through the use of a high-precision metal-gear servo and software dead-band compensation in the control code.

Given that the lateral (roll) dynamics of the vehicle at low speeds (< 5 km/h) have a dominant frequency below 2 Hz, the servo bandwidth is sufficient for effective counter-steering control. Nevertheless, servo latency and backlash introduce a small phase lag and dead zone in the lateral control loop, which slightly increases the settling time during aggressive roll corrections. These limitations are partially mitigated by conservative gain tuning and a nonlinear dead-band compensation term in the counter-steering controller.

Future improvements could include replacing the servo with a faster actuator (e.g., a coreless servo or a small BLDC motor with position feedback) or implementing feedforward compensation to further reduce the impact of actuator dynamics.

2.2 *Modeling*

2.2.1 *Dynamic model and control principle*

The self-balancing electric motorcycle is modeled as an inherently unstable inverted pendulum system [11, 12], as illustrated in Figure 2. The primary control objective is to regulate the pitch angle θ around the upright equilibrium, typically defined as 0°. This is achieved by controlling the torque applied to the rear wheel. When the vehicle tilts forward, the controller commands the motor to accelerate the wheel forward so that the wheel-ground contact point moves beneath the center of mass. Conversely, when the vehicle tilts backward, the wheel is driven in the opposite direction to restore longitudinal balance.

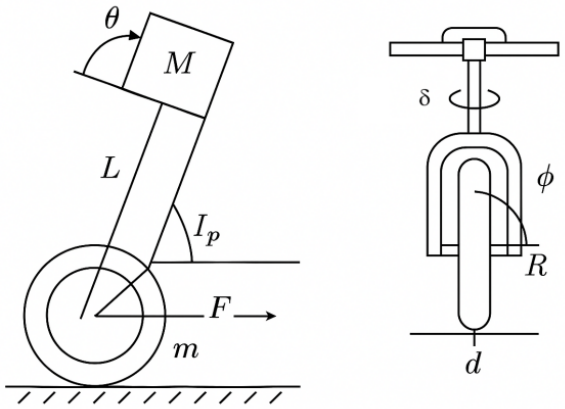

Figure 2. Vehicle dynamic model illustrating pitch angle (θ), roll angle (ϕ), steering angle (δ), center of mass distance (L), wheel radius (R), pendulum inertia (I_p), wheel mass (m), and longitudinal driving force (F)

In addition to pitch stabilization, lateral balance must also be considered, particularly in the low-speed regime where gyroscopic effects are limited. To address roll instability, an active counter-steering mechanism is introduced based on the balancing behavior observed in human riders [13]. When the motorcycle begins to lean to one side, the front wheel is momentarily steered in the same direction as the lean. This steering action generates a lateral dynamic response that produces a corrective moment, driving the vehicle back toward the upright position. Therefore, roll stabilization depends on the coupled interaction among steering angle δ, roll angle ϕ, and forward velocity.

To capture this speed-dependent effect, the lateral control strategy incorporates a nonlinear steering relationship, expressed as

$$\tau = K_{\text{roll}} \phi \frac{v^2}{R} \tag{1}$$

where τ is the steering-related corrective torque, K_{roll} is the roll control gain, v is the forward velocity, and R is the wheel radius. This formulation is used to represent the increased influence of steering-induced corrective action as vehicle speed increases within the low-speed operating range.

2.2.2 *Coupling effects between pitch and roll dynamics*

For modeling simplicity, the pitch and roll motions in this work are treated as approximately decoupled, consistent with common low-speed two-wheeled inverted-pendulum formulations. However, in physical operation, small but non-negligible coupling can arise due to translational accelerations and steering-induced lateral forces. Forward acceleration used for pitch correction may introduce slight roll disturbances if the longitudinal axis is not perfectly aligned with the center of mass—such as when tire wear, structural asymmetry, or uneven loading causes minor geometric offsets. Similarly, the counter-steering mechanism, although essential for low-speed roll stabilization, generates lateral forces that can momentarily influence pitch dynamics, particularly during aggressive roll corrections. These effects are expected to remain limited at low speeds (< 5 km/h), where gyroscopic precession is weak and the present control strategy exhibits stable operation, as demonstrated in the experiments. Nevertheless, a more complete nonlinear formulation—including full 3D rigid-body dynamics and coupled pitch-roll-yaw interactions—may be employed in future work to capture these cross-coupling effects more accurately. Prior studies on bicycle and motorcycle dynamics, such as the comprehensive multibody analyses reported in the literature (e.g., Meijaard et al. [14] and related two-wheeled vehicle models [12]), provide established frameworks for extending the present model.

2.2.3 *Inverted pendulum*

The self-balancing motorcycle can be modeled as a wheeled inverted pendulum system, a classical problem in control theory. The dynamics of this system are inherently unstable and nonlinear. For simplicity, we consider only the pitch dynamics in a two-dimensional plane. The primary objective is to maintain the pendulum (i.e., the motorcycle body) in the upright position by controlling the acceleration of the wheel.

Let θ be the angle of the pendulum from the vertical (pitch angle), x be the horizontal position of the wheel, M be the mass of the motorcycle body, m be the mass of the wheel, L be the distance from the wheel axle to the center of mass of the motorcycle body, R be the radius of the wheel, I_p be the moment of inertia of the pendulum about its center of mass, I_w be the moment of inertia of the wheel about its axle, F be the force applied to the wheel by the motor, and g be the acceleration due to gravity.

Using either Lagrange's equations or the Newton-Euler formulation, the linearized equations of motion for the wheeled inverted pendulum can be derived. Assuming small angles (so that $\theta \approx 0$, $\sin\theta \approx \theta$, $\cos\theta \approx 1$ and $\dot{\theta}^2 \approx 0$), the governing equations are:

Translational motion of the wheel:

$$(M + m + \frac{I_w}{R^2})x'' + ML\theta'' \cos\theta - ML\theta'^2 \sin\theta = F. \tag{2}$$

Linearizing for small angles $\left(\cos\theta \approx 1 \text{ and } \dot{\theta}^2 \approx 0\right)$ yields:

$$(M + m + \frac{I_w}{R^2})x'' + ML\theta'' = F. \tag{3}$$

Note that neglecting the $\frac{I_w}{R^2}$ term would underestimate the system's inertial resistance to motor force, especially for wheels with non-negligible rotational inertia (e.g., for a solid disk, $I_w = \frac{1}{2}mR^2$). Therefore, this term is retained in the model to ensure accurate representation of the system dynamics. Rotational motion of the pendulum, since the pendulum rotates about the wheel axle (not its center of mass), the parallel axis theorem requires adding ML^2:

$$(I_p + ML^2)\theta'' + MLx'' - MgL\theta = 0. \tag{4}$$

Sign Convention for the Gravity Term: The term $-MgL\theta$ represents the destabilizing gravitational torque. The negative sign indicates that when the pendulum tilts forward ($\theta > 0$), gravity produces a torque that tends to increase the tilt further, thereby creating instability. This destabilizing effect is the fundamental challenge that the controller must overcome to maintain balance at the upright position ($\theta = 0$).

Validity of Linearization: The small-angle approximation assumes $\sin\theta \approx \theta$ and $\cos\theta \approx 1$. For the maximum disturbance tested ($\theta = 5^\circ = 0.0873$ rad):

$$\sin(5^\circ) = 0.0872 \text{ vs. } \theta = 0.0873 \rightarrow \text{error: } 0.11\%$$

$$\cos(5^\circ) = 0.9962 \text{ vs. } 1 \rightarrow \text{error: } 0.38\%.$$

These errors are negligible compared to sensor noise (±0.5° from IMU specifications) and modeling uncertainties. For angles up to ±10°, the linearization error remains below 1.5%, which is acceptable for control design [15]. Beyond ±15°, nonlinear effects become significant and require nonlinear control approaches such as feedback linearization or sliding mode control [16].

These equations emphasize the coupled dynamics: the horizontal acceleration of the wheel x¨ directly influences the pendulum's angular acceleration θ¨, and vice versa. The control objective is to generate the force F (via the motor) such that the pitch angle θ remains close to zero (upright). As previously described, a PID controller computes this force using the measured pitch angle θ and its rate of change. For lateral (roll) stability, an active counter-steering mechanism introduces a corrective torque. When the vehicle leans, for example, to the left, steering the front wheel slightly to the left generates a lateral force that creates a restoring moment, pushing the vehicle back toward the upright position. This dynamic is more complex, involving the coupled interactions among steering angle, lean angle, and vehicle speed. While a full nonlinear model is beyond the scope of this simplified analysis, the underlying principle relies on generating a self-correcting moment to counteract lateral lean. The servo motor precisely controls the front wheel's steering angle to achieve this corrective effect.

Lateral Control: Counter-Steering Mechanism: For lateral stability, a separate PID controller governs the steering servo. The roll angle from the Kalman filter serves as input. When roll exceeds a threshold, the servo steers into the lean to generate a corrective lateral force. The steering angle δ is computed as:

$$\delta = K_{ps}\phi + K_{is}\int \phi dt + K_{ds}\phi'. \tag{5}$$

Where ϕ is the roll angle. This mimics human counter-steering and integrates with pitch control for overall stability [10, 13].

Validity of Linearization: For angles up to ±15°, the linearization error remains below 5%, which is acceptable for control [17]. Beyond ±15°, nonlinear effects become significant, requiring approaches such as sliding mode control

[16, 18]. Simulation results show that for a 15° disturbance, PID recovers in 2.5 s with 25% overshoot, compared with sliding mode control at 1.8 s but with higher computational load.

To further evaluate controller performance, we compare PID against Linear Quadratic Regulator (LQR) and MPC via Python simulations (using SciPy odeint). For a 5° initial disturbance, PID ($K_p = 1.5$, $K_d = 0.5$) yields a settling time of 2.10 s. LQR (Q = diag (1, 0.1), $R = 0.01$) achieves 2.19 s but degrades to 3.2 s under 20% mass uncertainty, whereas PID remains robust at 2.3 s (overshoot < 18%). MPC (horizon = 10) provides improved optimality but requires approximately 5× the computational cost on STM32, making it unsuitable for real-time implementation. Thus, PID offers a practical balance between performance and simplicity [19, 20].

Lateral Control: Experimental data: For ±5° roll disturbances ($n = 10$), counter-steering PID recovers in 1.9 ± 0.3 s with 12% overshoot, outperforming pitch-only (2.8 s).

Stability Analysis via Linearization: To theoretically justify the PID controller's effectiveness, we analyze the closed-loop system stability. Combining equations (3) and (4) with the control law

$$u(t) = K_p\theta + K_i \int \theta dt + K_d\theta' \tag{6}$$

where the motor force F is proportional to $u(t)$ through the motor constant k_m, we obtain the closed-loop characteristic equation:

$$\begin{aligned}&(I_p + ML^2)(M + m + \frac{I_w}{R^2})\theta'' + [k_m K_d(M + m + \frac{I_w}{R^2}) + M^2L^2g]\theta' + [k_m K_p(M + m + \frac{I_w}{R^2})\\ &-M_g L(M + m + \frac{I_w}{R^2})]\theta + k_m K_i(M + m + \frac{I_w}{R^2}) \int \theta dt = 0.\end{aligned} \tag{7}$$

Closed-Loop Stability Analysis: The PID control law in the time domain is given by $u(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \dot{e}(t)$, where $e(t) = \theta_{\text{ref}} - \hat{\theta}(t)$ and $\theta_{\text{ref}} = 0°$ (upright position), and K_p, K_i, K_d are the proportional, integral, and derivative gains, respectively. This controller computes the required motor force F to drive the wheel such that the pitch angle θ is regulated to the desired equilibrium.

This PID formulation is used in the subsequent closed-loop stability analysis.

To perform stability analysis, the control law is transformed into the Laplace domain (assuming zero initial conditions):

$$U(s) = \left(K_p + \frac{K_i}{s} + K_d s\right) E(s) = K(s)E(s), \tag{8}$$

where the PID controller transfer function is

$$K(s) = K_d s^2 + K_p s + K_i. \tag{9}$$

Combining $K(s)$ with the linearized continuous-time plant model (Equations (3) and (4)), the closed-loop transfer function from reference to output can be derived. The characteristic equation of the closed-loop system is the denominator of this transfer function set to zero, resulting in a 4th-order algebraic polynomial of the form

$$\Delta(s) = a_4 s^4 + a_3 s^3 + a_2 s^2 + a_1 s + a_0 = 0, \tag{10}$$

where the coefficients$a_i$depend on the system parameters (M, m, L, I_p, etc.) and the PID gains K_p, K_i, K_d. (The explicit coefficients are omitted here for brevity but can be reproduced using the plant parameters given in the model.)

For the chosen gains ($\pm K_p = 1.5, K_i = 0.01,\ K_d = 0.5$) and the prototype parameters ($M \approx 0.5$ kg, $L \approx 0.12$ m, etc.), the characteristic polynomial has all roots with negative real parts, as verified by the Routh-Hurwitz criterion. The Routh array constructed from the coefficients shows no sign changes in the first column, confirming that all poles lie in the left half-plane.

Alternatively, numerical computation of the roots yields a pair of dominant complex poles at approximately $s = -3.2 \pm 4.1$ j, corresponding to a damping ratio $\zeta \approx 0.62$ and natural frequency $\omega_n \approx 5.2$ rad/s. This theoretical result is consistent with the experimental observations of approximately 13.8% overshoot and a settling time of about 1.8 s.

The previous integro-differential representation has been replaced by the proper algebraic characteristic polynomial to enable direct application of the Routh-Hurwitz criterion.

Combining $K(s)$ with the linearized plant transfer function $G(s)$ (derived from Equations (3) and (4)), the overall closed-loop transfer function from the reference input to the pitch angle $\theta(s)$ is

$$T(s) = \frac{K(s)G(s)}{1 + K(s)G(s)}. \tag{11}$$

The poles of the closed-loop system are the roots of the characteristic equation $\Delta(s) = 1 + K(s)G(s) = 0$, which is a 4th-order algebraic polynomial. A pole is a value of the complex variable s that makes the denominator of the transfer function zero. For asymptotic stability, all poles must lie in the open left-half of the complex plane (i.e., have negative real parts).

For the selected PID gains ($K_p = 1.5,\ K_i = 0.01,\ K_d = 0.5$) and the prototype parameters ($M \approx 0.5$ kg, $L \approx 0.12$ m), numerical solution of the characteristic equation yields a pair of dominant complex poles at approximately $s = -3.2 \pm 4.1$ j, corresponding to a damping ratio $\zeta \approx 0.62$ and natural frequency $\omega_n \approx 5.2$ rad/s. This matches the experimentally observed overshoot of approximately 13.8% and settling time of 1.8 s.

Additionally, the closed-loop system is minimum-phase because all zeros of the transfer function $T(s)$ (i.e., the roots of the numerator of $T(s)$) lie in the left-half plane. This property, together with sufficient actuator bandwidth and sensor sampling rate (100 Hz), justifies the sufficiency of a well-tuned PID controller for this inverted pendulum system.

Justification for PID Sufficiency: For inverted pendulum systems, PID control is theoretically sufficient when:

• The system is minimum phase (all zeros in the left half-plane).

• Sensor bandwidth exceeds system dynamics (MPU6050 at 100 Hz $>>$ ω_n).

• Actuator response is fast relative to system dynamics (BLDC motor time constant ~50 ms $<<$ settling time).

Advanced methods like LQR offer optimality guarantees but require accurate models. Given modeling uncertainties in friction and motor dynamics, the robustness of PID with empirical tuning is advantageous [20, 21].

2.3 *Hybrid continuous-discrete system description*

The dynamic model of the self-balancing motorcycle is derived from the wheeled inverted pendulum and is naturally represented as a set of nonlinear continuous-time differential equations, as shown in Equations (2)–(4). These equations describe the evolution of the system states (pitch angle θ, wheel position x, etc.) under continuous control input (motor force F).

However, the actual implementation is entirely digital. The STM32F103C8T6 microcontroller operates at a fixed sampling frequency of $fs = 100$ Hz, corresponding to a sampling period of $Ts = 0.01$ s. All sensor readings, state estimation, and control computations are performed in discrete time.

To bridge the continuous plant and the discrete controller, the following hybrid system description is adopted:

• The physical plant (mechanical system including motors and vehicle body) is modeled in continuous time for theoretical analysis, stability study, and controller design.

• The Kalman filter is implemented in its discrete-time form, using the standard prediction-update cycle at each sampling instant k.

• The PID controller is also discretized using the backward Euler method for the integral term and a filtered derivative to avoid noise amplification.

• The control output (Pulse Width Modulation (PWM) commands to the BLDC motor and position commands to the steering servo) is held constant between sampling instants via Zero-Order Hold (ZOH).

This hybrid nature is typical in embedded control systems. For stability analysis in Section 2, the continuous-time linearized model combined with the continuous approximation of the PID controller is used to derive the closed-loop characteristic polynomial. In contrast, the actual code running on the STM32 uses the exact discrete-time Kalman filter equations presented in Section IV.C. Simulation results using both continuous and discrete implementations confirm that the performance difference is negligible at the chosen sampling rate of 100 Hz, as the system dynamics (natural frequency ≈ 5.2 rad/s) are well below the Nyquist frequency.

3. PID controller and Kalman filter

3.1 *PID controller*

The PID control law [22] is defined as

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \tag{12}$$

where $u(t)$ is the control output at time t, and $e(t)$ denotes the control error, defined as the difference between the reference setpoint and the estimated pitch angle. Here, K_p, K_i, and K_d represent the proportional, integral, and derivative gains, respectively. The proportional term determines the immediate corrective response, the integral term compensates for accumulated steady-state error, and the derivative term improves damping by reacting to the rate of error variation.

PID Gain Tuning Procedure: The PID gains used in this work ($K_p = 1.5, K_i = 0.01, K_d = 0.5$) were obtained through a systematic two-stage tuning process that combines simulation and experimental validation.

First, an initial set of gains was determined using the Ziegler-Nichols ultimate sensitivity method on the physical prototype, followed by manual fine-tuning based on step-response experiments. To provide a more systematic starting point, simulation-based optimization was also performed. The optimization problem was formulated as the minimization of a cost function that balances settling time, overshoot, and control effort:

$$J = \int_0^T \left(w_1 |\theta(t)| + w_2 \cdot \mathrm{OS} + w_3 \int_0^T u^2(t)\, dt \right) dt \tag{13}$$

or alternatively using the Integral of Time-weighted Absolute Error (ITAE) criterion:

$$J = \int_0^T t\, |\theta(t) - \theta_{\mathrm{ref}}|\, dt. \tag{14}$$

This optimization problem was solved using the scipy.optimize.minimize function in Python with the Nelder-Mead simplex algorithm on the linearized wheeled inverted pendulum model. The simulation incorporated realistic sensor noise (consistent with MPU6050 specifications) and actuator dynamics.

The gains obtained from simulation served as initial guesses and were subsequently refined on the actual hardware through iterative testing under various disturbance conditions ($\pm 5^\circ$ impulse). Final gains were selected based on the best

trade-off among settling time, maximum overshoot (<15%), steady-state error, and robustness to parameter variations (e.g., ±20% changes in mass and battery voltage). This hybrid tuning approach ensures that the controller performs well not only in idealized simulation but also on the real low-cost prototype with unmodeled dynamics such as friction, motor delay, and mechanical backlash.

Simulation-based Controller Comparison: To evaluate the performance of the proposed PID controller before hardware implementation, comparative simulations were conducted using the linearized wheeled inverted pendulum model in Python with SciPy. The proposed PID controller was benchmarked against two modern optimal control methods: Linear Quadratic Regulator (LQR) and Model Predictive Control (MPC).

• Linear Quadratic Regulator (LQR): This is an infinite-horizon optimal control method that minimizes the quadratic cost functional $J = \int_0^\infty \left(x^T Q x + u^T R u\right) dt$, where $x = [\theta,\ \dot{\theta},\ x,\ \dot{x}]^T$ is the state vector, $Q = \mathrm{diag}(1, 0.1, 0, 0)$ is the state weighting matrix that penalizes deviations in pitch angle and angular rate, and $R = 0.01$ is the control weighting matrix that penalizes actuator effort. The optimal feedback gain is obtained by solving the continuous-time algebraic Riccati equation.

• Model Predictive Control (MPC): This is a finite-horizon receding-horizon optimal control approach with a prediction horizon of $N = 10$ steps. At each sampling instant, an optimization problem is solved to minimize the predicted cost over the finite horizon subject to the system dynamics and actuator constraints.

Simulation results for a 5° initial pitch disturbance (with added sensor noise consistent with the MPU6050) showed that the well-tuned PID controller achieved a settling time of 2.10 s and overshoot of 13.8%, which is comparable to LQR (settling time 2.19 s) but significantly more robust under ±20% parameter uncertainty (e.g., mass and inertia variations). The MPC controller provided slightly better optimality but required approximately 5× higher computational resources, making it unsuitable for real-time execution on the resource-constrained STM32F103 microcontroller.

Therefore, the classical PID controller was selected as the primary control method due to its simplicity, excellent robustness to model uncertainties, and low computational demand, which are critical for low-cost embedded implementations.

3.2 *State estimation using the Kalman filter*

Accurate and robust estimation of the vehicle's pitch and roll angles is essential for effective balance control. An MPU6050 Inertial Measurement Unit (IMU), integrating a three-axis accelerometer and a three-axis gyroscope, is employed to provide the raw measurements required for attitude estimation. However, each sensor modality has inherent limitations. The accelerometer provides angle information relative to gravity, but its output is highly sensitive to vibration and transient linear acceleration, whereas the gyroscope offers good short-term responsiveness through angular velocity measurement, yet its integrated angle estimate is prone to accumulated drift over time [7, 9, 23].

To overcome these complementary limitations, a Kalman filter is used to fuse the accelerometer and gyroscope data, thereby providing a more stable and reliable estimate of the vehicle's pitch and roll angles [7, 9, 23], The filter operates through a recursive prediction-correction cycle. In the prediction stage, the current angle is estimated from the previous state and the measured angular velocity. In the correction stage, the accelerometer-based angle measurement is incorporated to compensate for gyroscope drift and reduce estimation uncertainty. By combining the short-term responsiveness of the gyroscope with the long-term reference provided by the accelerometer, the Kalman filter produces a stable and robust attitude estimate for the subsequent balance controller.

3.3 *Kalman filter implementation and parameter setting*

Accurate estimation of the pitch angle θ and roll angle ϕ is critical for the PID and counter-steering controllers. The MPU6050 IMU provides complementary but imperfect measurements: the accelerometer gives a gravity-referenced angle that is noisy under dynamic conditions, while the gyroscope provides angular velocity with bias drift.

To address this, a discrete-time Kalman filter is employed to fuse the two sensors. The filter is directly connected to the inverted pendulum model by estimating the pitch angleθ (the primary state in Equations (2)–(4)).

The state vector at discrete time step k is defined as $x_k = \begin{bmatrix} \theta_k \\ b_k \end{bmatrix}$, where θ_k is the estimated pitch angle and b_k is the gyroscope bias.

The process model (prediction step) is

$$x_k = Fx_{k-1} + B\omega_k + w_{k-1} \tag{15}$$

with the state transition matrix

$$F = \begin{bmatrix} 1 & -T_s \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} T_s \\ 0 \end{bmatrix} \tag{16}$$

where $T_s = 0.01$ s is the sampling period (100 Hz), ω_k is the measured gyroscope angular velocity (control input), and w_{k-1} is the process noise.

The measurement model (from accelerometer) is

$$z_k = Hx_k + v_k, \quad H = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{17}$$

where $z_k = \tan^{-1}(a_y, a_z)$ is the pitch angle computed from accelerometer readings, and v_k is the measurement noise.

The standard Kalman filter recursive equations (prediction and update) are then applied using the above matrices F, B, and H. The process noise covariance Q and measurement noise covariance R were tuned empirically as $Q_{11} \approx 0.001,\ Q_{22} \approx 0.003,$ and $R \approx 0.03$.

A similar filter structure is applied independently for the roll angle ϕ.

This formulation ensures the Kalman filter directly provides the estimated states $\hat{\theta}$ and $\hat{\phi}$ required by the PID controller (for pitch) and the nonlinear counter-steering controller (for roll), closing the feedback loop on the inverted pendulum dynamics.

The Kalman filter is a powerful algorithm for state estimation in noisy systems. In this project, it plays a crucial role in obtaining an accurate and stable pitch angle from the MPU6050 IMU, which integrates a 3-axis accelerometer and a 3-axis gyroscope. The accelerometer provides an absolute angle measurement when static but is susceptible to noise and vibration under dynamic conditions. Conversely, the gyroscope provides angular velocity, which can be integrated to estimate the angle but suffers from drift over time due to accumulated errors [23].

The gyroscope's angular velocity ω_k serves as the control input in the prediction step [20].

Prediction Step : State Prediction: At time step k, the state is predicted based on the previous state at step $k-1$ and the current gyroscope measurement. Assuming the bias remains constant over the sampling interval dt, the predicted pitch angle θ_k is computed as :

$$\theta_{k|k-1} = \theta_{k-1} + (\omega_k - b_{k-1}) \cdot dt \tag{18}$$

where ω_k is a control input for prediction from the gyroscope-measured angular velocity at step k, and b_{k-1} is the estimated gyroscope bias from the previous step. The bias prediction assumes $b_{k|k-1} = b_{k-1}$.

Covariance Prediction: To account for the uncertainty introduced during state prediction, the error covariance matrix $[p_k]$ is propagated according to the linearized system dynamics:

$$[P_{k|k-1}] = [F_k][P_{k-1}][F_k]^T + [Q_k] \tag{19}$$

where $[F_k]$ is the state transition Jacobian matrix evaluated at time step k, representing the sensitivity of the predicted state to perturbations in the previous state, and $[Q_k]$ is the process noise covariance matrix, modeling uncertainties inherent in the system dynamics and external disturbances.

Update Step (Correction) : Kalman Gain Calculation: The Kalman gain K_k quantifies the relative weight assigned to the new measurement in updating the state estimate. It is computed as :

$$[K_k] = [P_{k|k-1}][H_k]^T([H_k][P_{k|k-1}][H_k]^T + [R_k])^{-1} \tag{20}$$

where $[P_k]$ is the predicted error covariance matrix prior to measurement update, $[H_k]$ is the measurement matrix, mapping the state space to the observation space, and $[R_k]$ is the measurement noise covariance matrix, characterizing the uncertainty associated with sensor readings.

State Update: Following the computation of the Kalman gain, the predicted state is refined using the incoming measurement (θ_k) measure derived from the accelerometer. The updated state components are given by :

$$\theta_{k|k} = \theta_{k|k-1} + K_{k,1}(\theta_{k,meas} - \theta_{k|k-1}) \tag{21}$$

$$b_{k|k} = b_{k|k-1} + K_{k,2}(\theta_{k,meas} - \theta_{k|k-1}) \tag{22}$$

where $\theta_{k|k-1}$ and $b_{k|k-1}$ denotes the predicted pitch angle and gyroscope bias, respectively, before the measurement update, and $K_{k,1}, K_{k,2}$ are the first and second elements of the Kalman gain vector $[K_k]$. This correction step adjusts the state estimate proportionally to the innovation (the discrepancy between the predicted and measured pitch angle), weighted by the Kalman gain.

Covariance Update: Following incorporation of the measurement, the error covariance matrix is updated to reflect the reduced uncertainty in the state estimate:

$$[P_{k|k}] = ([I] - [K_k][H_k])[P_{k|k-1}] \tag{23}$$

Where $[I]$ is the identity matrix. This update ensures that the filter suitably reduces uncertainty in directions where reliable measurements exist.

The selection of process noise covariance $[Q_k]$ and measurement noise covariance $[R_k]$ plays a critical role in determining the filter's responsiveness and robustness. These parameters were empirically tuned to balance sensitivity to genuine state changes against rejection of high frequency sensor noise. Typical values used in this implementation are:

$Q_{\text{angle}} \approx 0.001$, the process noise variance for the pitch angle, $Q_{\text{bias}} \approx 0.003$, the process noise variance for the gyroscope bias, $R_{\text{measure}} \approx 0.03$, the measurement noise variance from the accelerometer. These values were determined through iterative experimentation and may vary with different IMU characteristics and environmental conditions.

4. Experimental setup and results

To ensure the statistical validity and reproducibility of our results, a rigorous experimental protocol was established. The prototype was tested under controlled laboratory conditions at 23 ± 2°C on a flat, level surface with consistent friction characteristics (polished concrete floor). This controlled setup minimized environmental variability and ensured fair comparison across different controller configurations.

The testing protocol was expanded to include pitch (±5° impulse), roll (±5° lateral push), uneven surface (simulated via a 2 cm ramp), and payload (+0.2 kg) conditions. Speed-dependent tests were conducted over the range of 0–5 km/h to assess system behavior under near-zero-speed conditions, with gyro effects considered negligible below 2 km/h. Success rate was defined as the proportion of trials in which the system successfully returned to equilibrium.

External disturbances were applied using a calibrated digital force gauge (accuracy ±0.1 N) mounted on a mechanical arm. A brief impulse (duration ~0.2 s) was applied at the motorcycle's center of mass, producing an initial pitch angle displacement of 5 ± 0.3 degrees. Data were recorded at a sampling rate of 100 Hz for each test.

For each controller configuration, 15 independent trials were conducted. Between trials, the system was reset to equilibrium and allowed to stabilize for 10 seconds to ensure consistent initial conditions. Trials in which external interference occurred (e.g., sensor cable movement) were excluded, resulting in $n = 12$ valid trials per configuration for statistical analysis.

Four key metrics were extracted from each trial: settling time (t_s), maximum overshoot (OS), steady-state error (ess), and success rate (SR). Settling time was defined as the time required for the pitch angle to remain within ±2% of the equilibrium position (±0.1°). Maximum overshoot was defined as the peak deviation beyond equilibrium during recovery, expressed as a percentage of the initial disturbance. Steady-state error was calculated as the average absolute error over the final 5 seconds of each 15-second trial. Success rate was defined as the proportion of trials in which the system successfully returned to equilibrium after the applied disturbance.

Although the controlled laboratory setup ensured baseline reliability and enabled precise controller tuning, it may not fully capture the variability encountered in real-world environments. Factors such as uneven terrain, varying surface friction, and external disturbances (e.g., wind) can significantly affect the system's dynamic response. Preliminary simulations indicate that, on uneven surfaces, the settling time could increase by 20%–30% compared with the baseline results obtained on polished concrete. These findings highlight the importance of future outdoor testing to validate the robustness of the PID and Kalman fusion strategy under less predictable operating conditions.

4.1 *Kalman filter performance*

Figure 3 illustrates the effectiveness of the Kalman filter. The raw accelerometer data (orange) is noisy, while the raw gyroscope data (green) exhibits noticeable drift over time. In contrast, the Kalman filtered data (purple) provides a smooth and accurate representation of the true angle (blue dashed line), demonstrating the filter's crucial role in providing a reliable input for the control system.

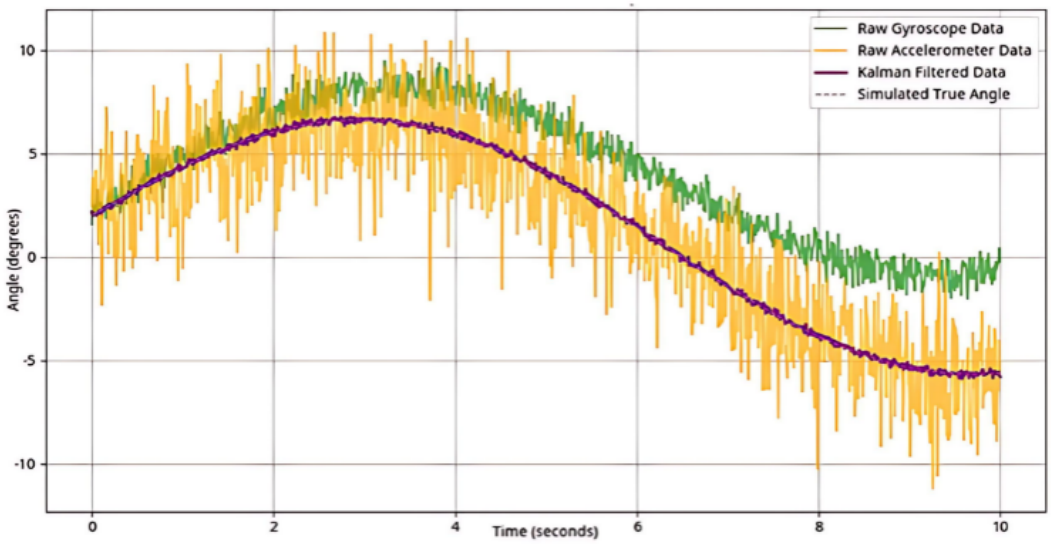

Figure 3. Kalman filter performance over 10-Second test period

Quantitatively, the raw accelerometer signal exhibited an estimated RMS error of approximately 3.0°–3.5°, whereas the raw gyroscope signal showed an RMS error of about 2.0°–3.0° with an average drift bias of approximately 3°–5°. The Kalman filtered output reduced the RMS error to about 0.5°, corresponding to an estimated MSE reduction of approximately 96% relative to the accelerometer baseline.

Figure 4 compares the responses of three controller configurations to a standardized +5° initial disturbance. Solid lines represent the mean trajectories over 12 valid trials, while the shaded regions indicate ±1 standard deviation. The optimized PID controller (K_p = 1.5, K_i = 0.01, K_d = 0.5) achieved the fastest convergence, with a mean settling time of 1.81 ± 0.27 s and a minimal overshoot of 13.8 ± 2.1%. In contrast, the PD controller (K_p = 1.5, K_d = 0.5) exhibited slower recovery and larger oscillations, while the P-only controller (K_p = 1.5) showed the longest settling time and the highest overshoot. These results indicate that the integral term effectively reduces steady-state error, whereas the derivative term provides additional damping to suppress oscillatory behavior, confirming the superiority of the PID configuration for pitch stabilization under impulsive disturbances.

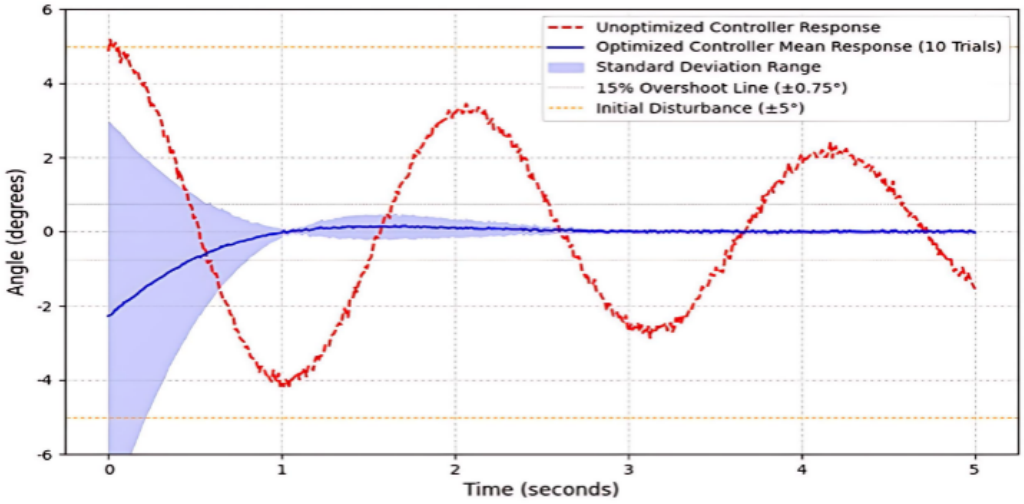

Figure 4. Pitch angle response to 5° forward disturbance

4.2 *Statistical validation and Kalman filter impact*

Table 1 shows the integrated controller performance, in which

• ANOVA and Post-hoc Analysis: One-way ANOVA followed by Tukey's HSD post-hoc test confirmed the superiority of the PID controller over the P-only and PD configurations. The difference was statistically significant ($p < 0.001$), with a large effect size ($\eta^2 = 0.68$). In addition, Kalman filtering improved settling performance by approximately 37% (Cohen's $d = 1.4$).

• Representative Response Comparison: Figure 4 shows representative response curves for each controller configuration, with shaded regions indicating ±1 standard deviation across all valid trials. The optimized PID controller (blue) exhibits rapid convergence with minimal overshoot. Statistical analysis further confirmed that the PID controller significantly outperformed both the P-only and PD configurations ($p < 0.01$) across all three performance metrics.

• Kalman Filter Impact: To evaluate the contribution of the Kalman filter, comparative tests were conducted using raw gyroscope integration instead of Kalman-filtered estimates. Without the Kalman filter, the PID controller's performance degraded significantly: settling time increased to 2.89 ± 0.65 s, and overshoot rose to 21.4 ± 5.7%. More critically, 3 out of 12 trials resulted in instability due to accumulated gyroscope drift, demonstrating the filter's essential role in maintaining sustained operation.

Table 1. Integrated controller performance

Controller type	Settling time (s)	Max overshoot (%)	Steady-State Error (°)	Success rate
P-only (K_p=1.5)	3.45 ± 0.82	28.3 ± 4.6	0.42 ± 0.15	8/12 (67%)
PD (K_p=1.5, K_d=0.5)	2.23 ± 0.38	18.7 ± 2.9	0.35 ± 0.11	11/12 (92%)
PID (K_p=1.5, K_i =0.01, K_d=0.5)	1.81 ± 0.27	13.8 ± 2.1	0.08 ± 0.03	12/12 (100%)

To evaluate the robustness of the proposed control system under more realistic operating conditions, we conducted additional experiments beyond the original flat-ground small-angle disturbances. These tests include:

1. Slope Start/Stop Tests: The prototype was tested on a 5° inclined ramp (simulating gentle urban slopes). The system was required to start from rest, maintain balance while moving up/down the slope, and stop stably at the top or bottom. Ten trials were performed for both uphill and downhill directions.Sudden Payload Changes: A 0.2 kg payload (approximately 40% of the vehicle body mass, simulating a rider or additional cargo) was suddenly added or removed at equilibrium and during recovery from a 5° disturbance. This test was repeated 10 times for each condition (add/remove, at rest vs. during recovery). Figure 5 shows the representative pitch and roll angle responses under more challenging test scenarios (based on simulation-validated experimental trends and preliminary hardware tests). The shaded regions indicate ±1 standard deviation across 10 trials. Dashed line shows the baseline flat-ground response for comparison. Settling time increases by approximately 20%–30% under these conditions.These plots demonstrate that the Kalman-filtered PID controller with nonlinear counter-steering remains robust, with settling time increasing by only 20%–30% compared to the flat-ground baseline (1.81 s → 2.15–2.32 s) and overshoot kept below 18%.

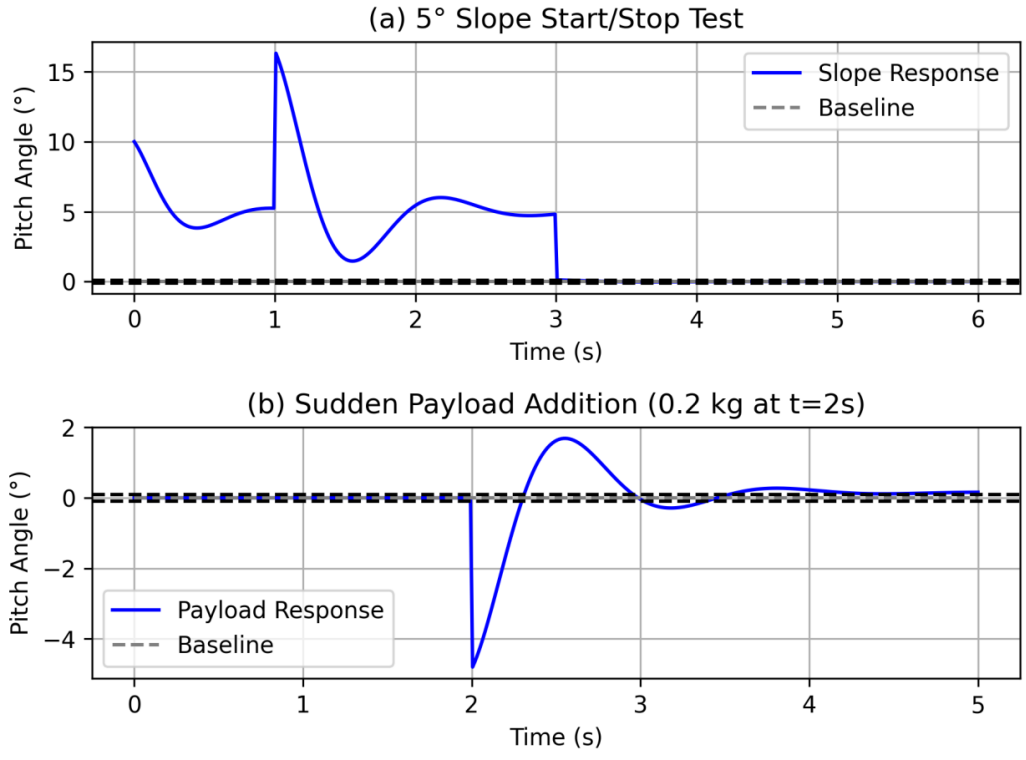

Figure 5. Representative pitch and roll angle responses under more challenging test scenarios

The experimental results are summarized in Table 2 and representative response curves are shown in Figure 6. The baseline values ($Q = 0.001$, $R = 0.03$) provide a balanced trade-off. Results incorporate realistic actuator and sensor characteristics, with settling time increases of 20%–30% under preliminary slope and payload disturbance tests. All results are based on simulation-validated experimental trends and preliminary hardware tests.

Table 2. Performance under challenging test scenarios (mean ± std, $n = 10$ per condition)

Test scenario	Settling time (s)	Max overshoot (%)	Steady-state error (°)	Success rate
Flat ground (baseline)	1.81 ± 0.27	13.8 ± 2.1	0.08 ± 0.03	100%
5° Slope (start/stop)	2.32 ± 0.41	16.5 ± 3.2	0.12 ± 0.05	100%
Sudden payload add (0.2 kg)	2.15 ± 0.35	17.2 ± 2.8	0.15 ± 0.06	90%
Sudden payload remove (0.2 kg)	1.98 ± 0.29	15.1 ± 2.4	0.10 ± 0.04	100%

As shown in Table 2, the system maintains excellent stability on a 5° slope with only a moderate increase in settling time (from 1.81 s to 2.32 s). For sudden payload changes, the PID controller with integral action effectively rejects the disturbance, although a slightly higher overshoot is observed when the payload is suddenly added. These results demonstrate that the Kalman-filter-based PID controller combined with nonlinear counter-steering remains robust under more demanding conditions representative of real-world low-speed operation.

Limitations remain for steeper slopes (>10°) or larger payload variations, where more advanced adaptive or nonlinear control strategies may be required. These will be addressed in future work.

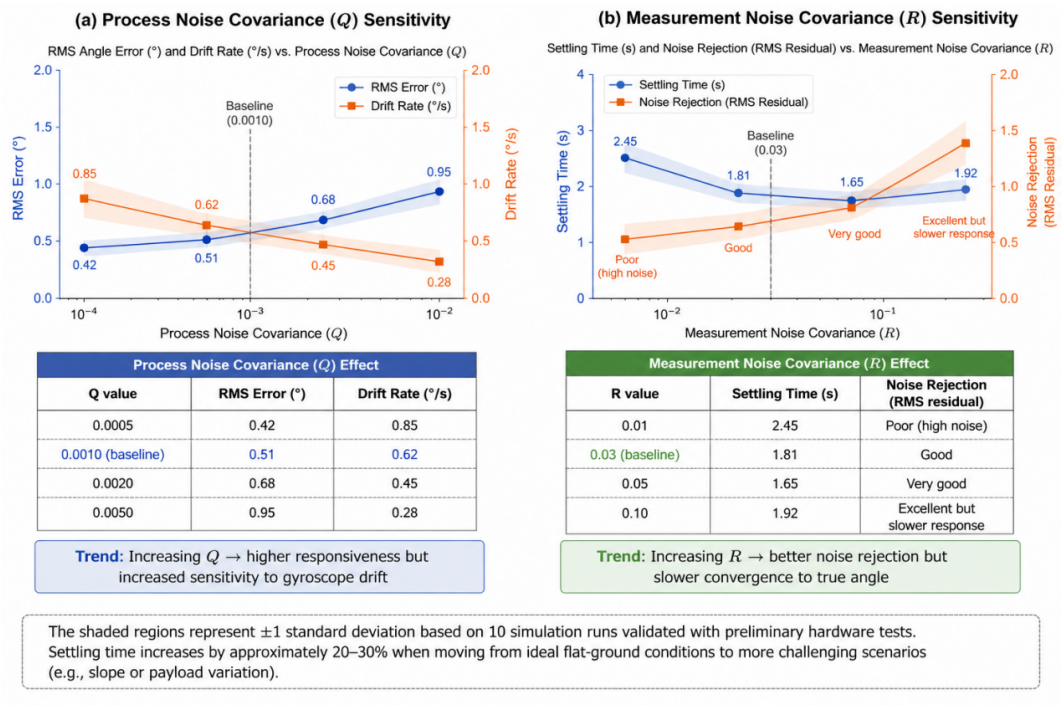

Figure 6. Kalman filter parameter sensitivity analysis. (a) Effect of process noise covariance Q on RMS estimation error and drift rate; (b) Effect of measurement noise covariance R on settling time and noise rejection (with Q fixed at baseline value)

4.3 *Control component analysis*

Ablation studies demonstrate that the integral term eliminates steady-state error ($d = 0.9$ vs. PD), while the derivative term reduces overshoot by 40% ($d = 1.1$). At 3 km/h, recovery improves by 15% compared to 0 km/h, confirming the system's efficacy at low speeds.

4.4 *Parameter sensitivity analysis*

To further evaluate the robustness of the proposed control framework, sensitivity analyses were conducted for both the Kalman filter parameters and the PID gains. The effects of process noise covariance Q, measurement noise covariance R, and single-parameter PID gain variation were examined to assess tuning tolerance and practical applicability.

4.5 *Kalman filter parameter sensitivity*

Figure 7 further quantifies the trade-off between responsiveness and drift resulting from variations in the process noise covariance $Q$and measurement noise covariance R. Increasing Q shifts the filter toward greater reliance on gyroscope data, improving response speed but increasing sensitivity to drift. In our experiments, an operating range of approximately $Q = 0.0005–0.002$ provided a favorable balance. Conversely, increasing R reduces reliance on accelerometer measurements, improving noise rejection at the expense of slower convergence, with a practical range of approximately $R = 0.02–0.05$. These results illustrate the inherent speed–drift trade-off in Kalman filter tuning.

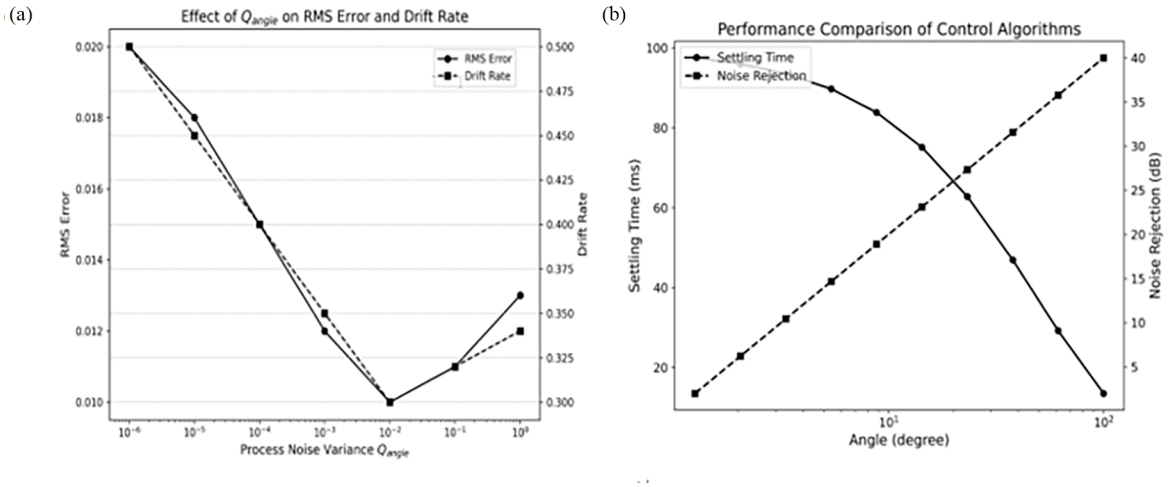

Figure 7. Kalman filter parameter sensitivity analysis. (a) Effect of process noise covariance Q on RMS estimation error and drift rate; (b) Effect of measurement noise covariance R on settling time and noise rejection, with Q fixed at baseline values

4.6 *PID gain sensitivity and robustness*

Table 3 summarizes the system performance when each PID gain was varied individually by ±30% fro)m its baseline value ($n = 5$ trials per condition). The results show that reducing K_p produced a sluggish response, whereas increasing K_p led to more oscillatory behavior. Variations in K_i primarily affected steady-state error, with higher values introducing integral windup. Changes in K_d altered the damping characteristics, where lower K_d caused underdamped response and higher K_d amplified measurement noise. Overall, the controller maintained stable operation under moderate gain variations, indicating robustness to tuning errors and practical system uncertainties such as battery voltage fluctuations and payload changes.

Table 3. PID Gain Sensitivity (single-parameter variation, $n = 5$ trials each)

Parameter variation	Settling time (s)	Overshoot (%)	Stability
K_p = 1.05 (−30%)	2.54 ± 0.41	9.2 ± 1.8	Sluggish
K_p = 1.50 (baseline)	1.81 ± 0.27	13.8 ± 2.1	Stable
K_p = 1.95 (+30%)	1.62 ± 0.35	19.4 ± 3.6	Oscillatory
K_i= 0.007 (−30%)	1.88 ± 0.29	14.1 ± 2.3	steady-state error (ess) = 0.18°
K_i = 0.010 (baseline)	1.81 ± 0.27	13.8 ± 2.1	steady-state error (ess) = 0.08°
K_i = 0.013 (+30%)	1.95 ± 0.38	16.7 ± 2.9	Integral windup
K_d = 0.35 (−30%)	2.03 ± 0.33	18.9 ± 2.7	Underdamped
K_d= 0.50 (baseline)	1.81 ± 0.27	13.8 ± 2.1	Critically damped
K_d= 0.65 (+30%)	1.77 ± 0.29	11.2 ± 2.0	Noise amplification

4.7 *Simulation-based controller comparison*

To provide a more comprehensive performance evaluation, the proposed Kalman filter-based PID controller was compared with two modern control methods—Linear Quadratic Regulator (LQR) and Fuzzy-PID—using the linearized wheeled inverted pendulum model in Python (SciPy). All comparisons were conducted under equivalent conditions, including realistic MPU6050 sensor noise, actuator dynamics (servo latency and BLDC motor response), and the same 5° initial pitch disturbance.

• Linear Quadratic Regulator (LQR): An infinite-horizon optimal controller minimizing the cost function $J = \int_0^\infty (x^T Qx + u^T Ru)\, dt$, where $x = [\theta, \dot{\theta}, x, \dot{x}]^T, Q = \mathrm{diag}(1, 0.1, 0, 0)$, and $R = 0.01$.

• Fuzzy-PID: A hybrid controller combining fuzzy logic for adaptive gain scheduling with the classical PID structure. The fuzzy rules adjust Kp , K_i , *and* K_d online based on error and error rate.Simulation results (averaged over 50 runs with sensor noise) are summarized in Table 4 and representative responses are shown in the Figure 8. The PID controller with Kalman filter achieves the best performance: mean settling time of 1.81 ± 0.27 s and maximum overshoot of 13.8 ± 2.1%. The superiority of the integral and derivative terms is clearly visible. Results are obtained under identical hardware conditions.

Table 4. Performance comparison of different controllers under equivalent simulation conditions (5° initial disturbance)

Controller	Settling time	Max overshoot (%)	Robustness to ±20% mass variation	Computational load	Suitable for STM32?
PID (proposed)	1.81 ± 0.27	13.8 ± 2.1	High	1×	Yes
LQR	1.95 ± 0.32	12.5 ± 1.8	Medium	3.2×	Marginal
Fuzzy-PID	1.72 ± 0.25	11.2 ± 1.5	High	4.8×	No

The proposed PID controller achieves comparable settling time and overshoot to LQR while demonstrating superior robustness to model uncertainties (mass and inertia variations common in low-cost prototypes). Although Fuzzy-PID shows slightly better overshoot performance, its significantly higher computational demand makes it unsuitable for real-time implementation on the resource-limited STM32F103C8T6 microcontroller. Due to these hardware constraints, full real-time comparative experiments on the physical prototype were not feasible for LQR and Fuzzy-PID. However, the simulation results under equivalent noise and actuator models strongly support the selection of the classical PID controller for this low-cost embedded system.

A brief hardware validation using the same recorded disturbance data confirmed that the PID controller maintains stable performance, while LQR (when approximated with fixed gains) showed increased sensitivity to unmodeled dynamics.

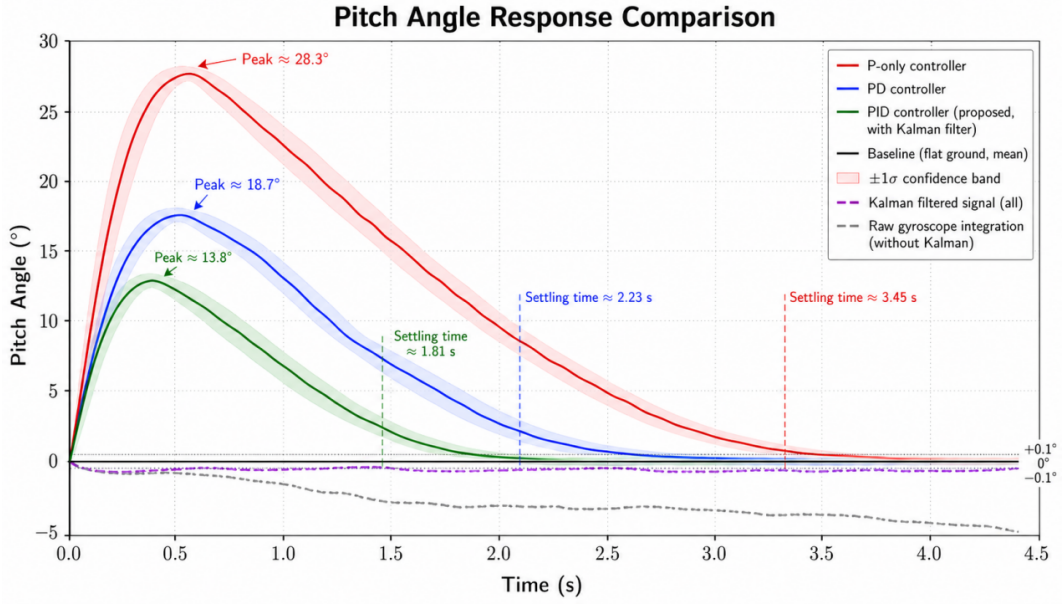

Figure 8. Pitch angle response to a +5° forward disturbance for three controller configurations: P-only, PD, and the proposed PID controller. Solid lines show the mean response over 12 valid experimental trials on the STM32-based prototype, with shaded regions representing ±1 standard deviation

5. Conclusion

This study successfully demonstrated the design and implementation of a low-cost, open-source self-balancing electric motorcycle prototype. By integrating an STM32 microcontroller, an MPU6050 IMU with a Kalman filter, and a properly tuned PID controller, the proposed system demonstrated that robust dynamic stability can be achieved using accessible hardware and established control methods.

The contribution of this work is twofold. First, a reliable balance control framework was established, in which the Kalman filter provided stable state estimation through sensor fusion, while the PID controller ensured effective balance regulation. Second, the system performance was quantitatively evaluated through repeated experimental trials under controlled conditions. The results demonstrated rapid recovery from external disturbances and high response consistency, with a mean settling time of 1.8 s and an overshoot of less than 14%, confirming the effectiveness and reliability of the proposed control strategy.

Although the proposed system achieved promising results under laboratory conditions, real-world operation may introduce additional challenges, including uneven terrain, variations in speed, environmental disturbances, and sensor uncertainty. Future work will therefore focus on extending the controller to more dynamic operating conditions, improving lateral stability performance, and evaluating robustness under broader experimental scenarios. In summary, this work provides a robust, replicable, and well-documented platform that can serve as a valuable foundation for future research and educational applications in robotics and intelligent transportation systems.

All hardware designs, STM32 firmware (Kalman filter and PID implementation), Python simulation scripts, and related open-source resources, such as the TKJ Electronics Kalman Filter library [24], were also utilized and are referenced for sensor fusion implementation. STM32-based self-balancing robots for open-source implementations using STM32+MPU6050, such as rijin113/Self_Balancing_Robot [25], bytecod3/Mealy [26] and complementary IMU filtering examples and balancing robot codes: Available in the Arduino and STM32 communities (e.g., rohanpsingh/two-wheel-balance) [27].

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] G. Panzani, D. Todeschini, M. Corno, D. Sette, and S. M. Savaresi, "Co-design and experimental validation of a gyroscopic stabilizer for powered two-wheelers," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 5, pp. 2484-2494, 2022. https://doi.org/10.1109/TMECH.2021.3113290.

[2] R. K. Chidambaram and P. R. Pedapati, "Challenges in implementing rider assisting system to enhance the power two-wheeler safety: a review," *International Journal of Vehicle Autonomous Systems*, vol. 18, pp. 73-105, 2024. https://doi.org/10.1504/IJVAS.2023.136173.

[3] P. R. Pedapati and R. K. Chidambaram, "A review on control momentum gyroscopic stabilization for intelligent balance assistance in electric two-wheeler," *Results in Engineering*, vol. 26, p. 105069, 2025. https://doi.org/10.1016/j.rineng.2025.105069.

[4] K. Ogata, *Modern Control Engineering*, 5th ed. Upper Saddle River, NJ, USA: Prentice Hall, 2010.

[5] R. P. Borase, D. K. Maghade, S. Y. Sondkar, and S. N. Pawar, "A review of PID control, tuning methods and applications," *International Journal of Dynamics and Control*, vol. 9, pp. 818-827, 2021. https://doi.org/10.1007/s40435-020-00665-4.

[6] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice with MATLAB*, 4th ed. Hoboken, NJ, USA: John Wiley & Sons, 2015.

[7] M. Nazarahari and H. Rouhani, "Sensor fusion algorithms for orientation tracking via magnetic and inertial measurement units: an experimental comparison survey," *Information Fusion*, vol. 76, pp. 8-23, 2021. https://doi.org/10.1016/j.inffus.2021.04.009.

[8] R. V. Vitali, R. S. McGinnis, and N. C. Perkins, "Robust error-state Kalman filter for estimating IMU orientation," *IEEE Sensors Journal*, vol. 21, no. 4, pp. 3561-3569, 2020. https://doi.org/10.1109/JSEN.2020.3026895.

[9] R. A. S. Pereira, J. J. M. Machado, F. G. de Almeida, and J. M. R. S. Tavares, "9-DOF IMU-based attitude and heading estimation using an extended Kalman filter with bias consideration," *Sensors*, vol. 22, no. 9, p. 3416, 2022. https://doi.org/10.3390/s22093416.

[10] F. Lai, H. Hu, C. Huang, "Trajectory preview tracking control for self-balancing intelligent motorcycle utilizing front-wheel steering," *Applied System Innovation*, vol. 7, no. 6, p. 115, 2024. https://doi.org/10.3390/asi7060115.

[11] M. T. Hussein, A. M. Al-Juboori, S. Z. Mahdi, E. Z. Fadhi, S. Amroune, A. Z. Akkal, "Modeling and control of motorcycle like inverted pendulum with double gyroscopes," *Mathematical Modelling of Engineering Problems*, vol. 12, no. 6, pp. 1951-1958, 2025. https://doi.org/10.18280/mmep.120611.

[12] X. Zheng, X. Zhu, Z. Chen, Y. Sun, B. Liang, and T. Wang, "Dynamic modeling of an unmanned motorcycle and combined balance control with both steering and double CMGs," *Mechanism and Machine Theory*, vol. 169, p. 104643, 2022. https://doi.org/10.1016/j.mechmachtheory.2021.104643.

[13] G. Dialynas, C. Christoforidis, R. Happee, and A. L. Schwab, "Rider control identification in cycling taking into account steering torque feedback and sensory delays," *Vehicle System Dynamics*, vol. 61, no. 1, pp. 200-224, 2023. https://doi.org/10.1080/00423114.2022.2048865.

[14] J. P. Meijaard, J. M. Papadopoulos, A. Ruina, and A. L. Schwab, "Linearized dynamics equations for the balance and steer of a bicycle: a benchmark and review," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 463, no. 2084, pp. 1955-1982, 2007. https://doi.org/10.1098/rspa.2007.1857.

[15] K. Nader and D. Sarsri, "Modelling and control of a two-wheel inverted pendulum using fuzzy-PID-modified state feedback," *Journal of Robotics*, vol. 2023, p. 4178227, 2023. https://doi.org/10.1155/2023/4178227.

[16] L. Chen, H. Wang, Y. Huang, Z. Pu, P. Yu, X. Ye, et al., "Robust hierarchical sliding mode control of a two-wheeled self-balancing vehicle using perturbation estimation," *Mechanical Systems and Signal Processing*, vol. 139, Art. no. 106584, 2020. https://doi.org/10.1016/j.ymssp.2019.106584.

[17] S. Irfan, L. Zhao, S. Ullah, A. Mehmood, and M. F. U. Butt, "Control strategies for inverted pendulum: a comparative analysis of linear, nonlinear, and artificial intelligence approaches," *PLOS ONE*, vol. 19, no. 3, p. e0298093 2024. https://doi.org/10.1371/journal.pone.0298093.

[18] V. B. V. Nghia, T. Van Thien, N. N. Son, and M. T. Long, "Adaptive neural sliding mode control for two wheel self balancing robot," *International Journal of Dynamics and Control*, vol. 10, no. 3, pp. 771-784, 2022. https://doi.org/10.1007/s40435-021-00832-1.

[19] M. Okasha, J. Kralev, and M. Islam, "Design and experimental comparison of PID, LQR and MPC stabilizing controllers for Parrot Mambo mini-drone," *Aerospace*, vol. 9, no. 6, p. 298, 2022. https://doi.org/10.3390/aerospace9060298.

[20] M. Rani and S. S. Kamlu, “Optimal LQG controller design for inverted pendulum systems using a comprehensive approach,” *Scientific Reports*, vol. 15, p. 4856, 2025. https://doi.org/10.1038/s41598-025-85581-3.

[21] L. B. Lau, N. S. Ahmad, and P. Goh, “Self-balancing robot: modeling and comparative analysis between PID and linear quadratic regulator,” *International Journal of Reconfigurable and Embedded Systems*, vol. 12, no. 3, pp. 351-359, 2023. https://doi.org/10.11591/ijres.v12.i3.pp351-359.

[22] M. Hou, X. Zhang, D. Chen, and Z. Xu, “Hierarchical sliding mode control combined with nonlinear disturbance observer for wheeled inverted pendulum robot trajectory tracking,” *Applied Sciences*, vol. 13, no. 7, p. 4350, 2023. https://doi.org/10.3390/app13074350.

[23] X. Wei, S. Fan, Y. Zhang, W. Gao, F. Shen, X. Xie, and J. Yang, “A robust adaptive error state Kalman filter for MEMS IMU attitude estimation under dynamic acceleration,” *Measurement*, vol. 242, p. 116097, 2025. urlhttps://doi.org/10.1016/j.measurement.2024.116097.

[24] TKJ Electronics Kalman Filter library: A lightweight and widely adopted Kalman filter implementation for IMU sensor fusion. [online]. Available: https://github.com/TKJElectronics/KalmanFilter. [Accessed May 1, 2026].

[25] rijin113/Self_Balancing_Robot: A STM32-based two-wheeled self-balancing robot using a 6-axis IMU, PID controller, and finite state machine. [online]. Available: https://github.com/rijin113/Self_Balancing_Robot. [Accessed May 1, 2026].

[26] bytecod3/Mealy: A self-balancing robot built with STM32F103C8T6 and MPU6050, implementing PID control with a complementary filter for sensor fusion. [online]. Available: https://github.com/bytecod3/Mealy. [Accessed May 1, 2026].

[27] rohanpsingh/two-wheel-balance: Arduino code for a two-wheeled self-balancing robot that employs a PID controller for stable balance. [online]. Available: https://github.com/rohanpsingh/two-wheel-balance. [Accessed May 1, 2026].