Experimental Design and Performance Analysis of a Self-Balancing Robot Using PID Control

Bhargav Prajwal Pathri[1], Krishna Vamshi Ganduri[2], Sravan Sripadi[3], Sanka Sai Darahas[4], Boggula Lohith[5], Boggula Lokesh[6], Sheik Shoaib Akhtar[7],Pabbisetty Manideep[8]
[1,3,4,5,6,8]School of Technology, Woxsen University, Hyd, India
[2,7]AI Research Centre, Woxsen University, Hyd, India.
Mails: bhargav.pathri@gmail.com [1], kvganduri@gmail.com [2], sravansripadi6@gmail.com [3], saidarahas75@gmail.com [4], lohithboggula22@gmail.com [5], lokeshboggula06@gmail.com [6],munnashoaib.sk@gmail.com [7], pabbisettymanideep258@gmail.com[8]

Abstract

Self-balancing robots are a significant group of autonomous systems that capture the classical inverted pendulum problem, and as such, they are both a control design benchmark and a real-world mobile robot platform. This work describes the development and construction of an affordable two-wheeled self-balancing robot based on an ESP32 microcontroller, MPU6050 inertial measurement unit (IMU), and Proportional–Integral–Derivative (PID) controller. The main goal is to provide real-time stability using continuous feedback control under the condition of making hardware affordable and reproducible for educational and research purposes. The tilt angle of the robot is estimated from sensor fusion of accelerometer and gyroscope measurements by a complementary filter, which allows for stable orientation estimation in noise and drift. Experimental tests show that the PID controller with the adjusted gains (Kp = 25, K_i = 0.8, K_d = 10) stabilizes within 1.5 – 2 seconds after initiation and recovers from small and moderate perturbations within 300 – 700 ms. The findings support the efficacy of PID control for low-cost dynamic stabilization as it provides a good balance between simplicity, computational cost, and robustness. This work provides a reproducible scheme that can be applied to other advanced control methods like Linear Quadratic Regulator (LQR), fuzzy logic, or reinforcement learning, opening doors to future uses in education, service robotics, and assistive mobility.

Keywords: Dynamic stabilization, ESP32, Inverted pendulum, Kalman filter, microcontroller, PID control, Self-balancing robot

I. INTRODUCTION

The quest for dynamic stabilization in robotic platforms has taken center stage in the past decade as it plays a pivotal function in making agile, adaptive, and intelligent mobility possible[1][2]. Of all platforms, self-balancing robots present the most challenging but also the most appropriate benchmark problem because they mimic the traditional inverted pendulum, a naturally unstable system that requires ongoing feedback to achieve equilibrium[3]. These robots illustrate the real-time control, sensor fusion, and embedded computation principles, and hence, they are not just worth exploring academically but also critical to apply practically in assistive robotics, autonomous delivery systems, and human–robot interaction. Traditional stability methods that are based on static support are not sufficient for mobile platforms where compactness, efficiency, and adaptability to disturbances are demanded[4]. With the incorporation of an ESP32 microcontroller and an MPU6050 inertial measurement unit and a Proportional–Integral–Derivative (PID) controller, this research provides a low-cost efficient strategy for accomplishing dynamic balance robustness in a two-wheeled robot. To proving stability during perturbation, the system defines PID as an efficient and trustworthy control method for embedded robotic systems, thus offering a way forward towards unifying theoretical control models with practical robotic use[5].

A. Background and Motivation

The quest for balance ranks among the most rudimentary problems in robotics. From humanoid robots to agile mobile platforms, the capacity to preserve stability during interaction with the physical environment is a fundamental need of autonomy. One of the classical problems in control and robotics, the two-wheeled self-balancing robot (TWSBR) stands as a simplification and rich testbed for dynamic stabilization among the classical control and robotics problems. Its dynamics are those of an inverted pendulum, a classical underactuated system, which is unstable except with ongoing corrective action. This capability renders self-balancing robot's superior vehicles for learning, experimentation, and innovation in feedback control, estimation, and embedded system design[6][7].

There has been a recent revival of interest in building low-cost reliable prototypes that are usable for research purposes as well as for education. Low-cost hardware like the ESP32 microcontroller, combined with low-cost

S. Uppapalli and D. Chakraborty (eds.), *Proceedings of the International Conference on Mechanics, Materials and Mechatronics (ICM³ 2026)*, Atlantis Highlights in Engineering 47,
https://doi.org/10.2991/978-94-6239-764-4_15

inertial sensors like the MPU6050, allows students and researchers to play around with real-time control techniques on embedded devices. Not only do such systems give insights about traditional controllers like PID but also provide ways to explore sophisticated techniques like Linear Quadratic Regulators (LQR), fuzzy logic, and reinforcement learning. The impetus for the current work is at the crossroads of education, accessibility, and scientific rigor. There are high-end robots with sophisticated stabilization systems, but they are still pricey, sophisticated, and sometimes proprietary, relegating them to classrooms and laboratories with limited resources[8]. Our self-balancing robot seeks to seal the gap between theory and practice by providing a reproducible, low-cost, and scalable platform. It is both a proof-of-concept for control theory in practice and a basis upon which further investigations into adaptive and AI-based control techniques can proceed. Self-balancing robots have real-world applications beyond mere scholarly interest. The underlying principles apply to real-world systems including Segway transporters, elderly assistive mobility devices, last-mile logistics robotic delivery platforms, and agile inspection robots in warehouse and industrial settings. Understanding these dynamics at a more profound level, especially in cost-sensitive settings, solidifies the pipeline from classroom to real-world innovation[9].

B. The Inverted Pendulum Problem

The inverted pendulum problem is perhaps the most extensively analysed model within control theory and robotics, due to its unstable nature. In essence, the system is a rigid rod with its center of mass positioned above a pivot. Under a small perturbation, the pendulum is moved away from equilibrium, and without feedback, the pendulum will fall because of gravity[10]. The task is thus to create a controller that will impose forces to return the pendulum to balance and maintain it upright. For more than half a century, the inverted pendulum has been an industry-standard problem for testing control algorithms. It has been utilized to compare and test controllers from simple proportional–integral–derivative (PID) loops to optimal control, adaptive control, and artificial intelligence methods. The model's significance lies in three properties:

- It requires ongoing corrective control, thus being more difficult than static stabilization issues.
- The control input is not enough to directly control all system states, and indirect stabilization techniques need to be designed cleverly.
- What is learned from the inverted pendulum control applies to real-world applications like rocket stabilization, biped walking, and robotic manipulation.

When the inverted pendulum is mounted on a moving cart or wheels, such as in the self-balancing robot, the system adds additional coupling between translational and rotational dynamics. The wheels need to advance or recede to resist the falling mass and demand coordination of sensor feedback, control algorithms, and actuator response. This makes the two-wheeled robot an optimal, but realistic, realization of the inverted pendulum problem. Though simple in theory, realizing the inverted pendulum in hardware is not straightforward. Physical systems have to contend with sensor noise, actuator delays, friction, battery voltage fluctuations, and mechanical tolerances. Therefore, work on self-balancing robots not only advances theoretical knowledge but also provides researchers with the ability to tackle the grubby realities of physical systems[11].

II. RELATED WORK

Two-wheeled self-balancing robots (TWSBRs) are a textbook realization of the inverted-pendulum problem and an established benchmark for testing control algorithms[12] on hardware. Recent reports focus on their underactuated, nonlinear behavior, and the range of controllers that have been tried out on them - from classic PID and LQR to MPC, sliding-mode, fuzzy[13], and learning-based controllers - and often on small, low-cost platforms for reproducibility and educational purposes[14][15]. These papers also contend that valid comparisons need some standard measurements and open datasets, which are still scarce throughout the field. For teaching and fast prototyping, open platforms reduce the entry cost of controlled experimentation. EduBal is a typical case: an inexpensive, open-source balancing robot employed at RWTH Aachen and ETH Zurich for hands-on laboratories in modelling, identification, and SISO/MIMO control with Simulink-based deployment. Online tuning and live signal plots in the platform documentation allow students to loop in iterations on controllers while monitoring real-time response, something that has been imitated in subsequent teaching platforms[16][17]. Aside from EduBal, a few recent publications offer kits specifically designed for curriculum integration and project-based lessons, with enhanced student participation and skills mastered. Although such platforms vary in bill of materials and software stacks, they both have two drawbacks: (i) qualitative evaluation or a limited set of metrics (time-to-balance, overshoot) is common and (ii) high-rate logs made publicly available for cross-paper benchmarking are uncommon. A second educational thread contrasts model-based vs. data-based control in instructional settings, advocating

side-by-side experiments on the same robot to teach trade-offs in modelling effort, computational cost, and robustness, again underscoring the didactic value of a reproducible platform[18][19].

Hybridizations, LQR with adaptive or robust layers, are therefore becoming increasingly prevalent. Fuzzy and hybrid schemes seek nonlinearities and uncertainty without burdensome modelling. A 2025 work combines fuzzy parallel distributed compensation (PDC) with LQR gains and a sliding neural network to update rule weights online on an STM32 cart with improved stability margins over simple LQR without becoming MCU-unsuitable[20]. These designs self-consciously aim at embedded practicability over compute-rich MPC or deep RL and represent a pragmatic classical-plus-light-AI trend in TWSBR control. On the estimation front, attitude/tilt accuracy has a significant impact on controller performance. Complementary filters continue to be appealing for ESP32-level hardware, with newer IMU papers suggesting adaptive and error-state Kalman extensions that manage bias and high dynamics better, with prospects for minimizing drift-caused steady-state error if computational budgets permit[21].

Learning-controlled control is increasingly being applied to inverted-pendulum-style systems, either to supplant controllers end-to-end or to add to traditional loops. A 2024 paper formulates a deep RL controller with a high-definition reward function and a two-phase learning protocol, announcing enhanced robustness to parameter changes and small steady-state errors in hardware[22]. The approach tackles sim-to-real transfer through framed rewards and defended adaptation methods that can be transferred to TWSBRs. Concurrently, there is an embedded deployment gap: inference and training workload may outpace low-cost MCUs, while long training time or data demands make classroom applications difficult. The latest hybrid controllers, LQR or SMC stabilized cores with learning-based adaptation to uncertainties, strike a balance between giving up some optimality for real-time tractability. Surveys explicitly warn that RL/DL benefits need to be balanced against compute, safety, and repeatability limits on microcontrollers. Lastly, application-driven designs push balancing to perception and navigation. A 2023 inspection robot combines laser SLAM on a two-wheel self-balancing platform for narrow environments, showcasing the necessity to co-design estimation, control, and perception stacks and to test on multi-objective tasks outside of static balancing[23].

III. SYSTEM DESIGN AND METHODOLOGY

A. Inverted Pendulum Modelling

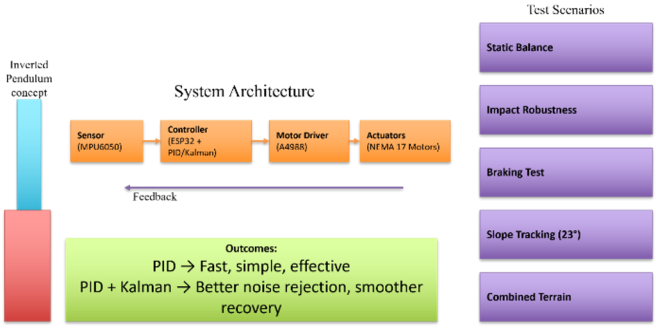

Fig.1 Architecture of self-balancing robot

Kinematics

Body Composition:

$$P_b = \begin{bmatrix} x + lsin\theta \\ R + lcos\theta \end{bmatrix} \quad \ddot{P}_b = \begin{bmatrix} \dot{x} + l\dot{\theta}cos\theta \\ -l\dot{\theta}\dot{sin}\theta \end{bmatrix} \qquad (1)$$

Wheel axle translational velocity is $\dot{x}$. Wheel angular rate $\dot{\emptyset} = \frac{\dot{x}}{R}$

Energies

Kinetic energy T

Sum of wheel translation rotation and body translation rotation[24]:

$$T = \frac{1}{2}(M_w + M_b)\dot{x}^2 + M_b l\dot{\theta} cos\theta + \frac{1}{2}M_b l^2\dot{\theta}^2 + \frac{1}{2}I_b\dot{\theta}^2 + \frac{1}{2}I_w\left(\frac{\dot{x}}{R}\right)^2 \quad (2)$$

$$= \frac{1}{2}\left(M_w + M_b + \frac{I_w}{R^2}\right)\dot{x}^2 + M_b l cos\theta\dot{x}\dot{\theta} + \frac{1}{2}(I_b + M_b l^2)\dot{\theta}^2 \quad (3)$$

Potential energy V

Take zero at axle height:

$$V = M_b g(R + lcos\theta) \Rightarrow \nabla_\theta V = -M_b g l sin\theta \quad (4)$$

(constant term $M_B g$R can be dropped if desired).

Rayleigh dissipation D

$$\mathcal{D} = \frac{1}{2}b_w\dot{x^2} + \frac{1}{2}b_b\dot{\theta^2} \quad (5)$$

Lagrange equations

With $q = [x, \theta]^T$ and input vector $u = [\tau, 0]^T$ (only the x coordinate is directly actuated via wheel torque):

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}}\right) - \left(\frac{\partial T}{\partial q}\right) + \left(\frac{\partial \mathcal{D}}{\partial \dot{q}}\right) + \left(\frac{\partial V}{\partial q}\right) = Bu \quad (6)$$

This yields the standard manipulator form:

$$\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix}\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} + \begin{bmatrix} C_1(q,\dot{q}) \\ C_2(q,\dot{q}) \end{bmatrix} + \begin{bmatrix} 0 \\ M_b g l sin\theta \end{bmatrix} + \begin{bmatrix} b_w\dot{x} \\ b_b\dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{\tau}{R} \\ 0 \end{bmatrix} \quad (7)$$

Where,

$$m_{11} = M_w + M_b + \frac{I_w}{R^2} \quad C_1 == -M_b l sin\theta\dot{\theta}^2 \quad (8)$$

$$m_{12} = m_{21} = M_b l cos\theta \quad C_2 = 0 \quad (9)$$

$$m_{22} = I_b + M_b l^2 \quad (10)$$

Intuition: M(q) is the coupled inertia, C captures centripetal/Coriolis effects, G is gravity, D is viscous loss, and BB maps motor torque to generalized force (τ/R because force at the wheel–ground equals torque over radius).

B. PID Control

The PID[25] controller shown Fig.2, the most popular feedback control approach in robotics because of its simplicity, ease of implementation on embedded devices, and efficacy in a variety of applications, such as stabilization of an inverted pendulum. In our unicycle robot, the PID calculates the corrective motor command as a weighted sum of three terms: the proportional component, which responds immediately to the immediate error; the integral component, which sums up past errors to remove steady-state drift due to sensor bias or mechanical imbalance; and the derivative component, which forecasts the error rate change and introduces damping to prevent oscillations[26].

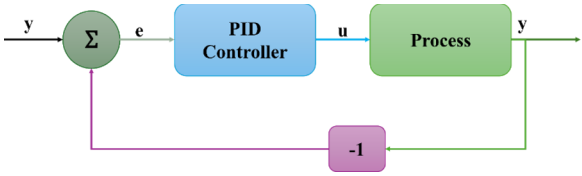

Fig.2 PID controller setup

$$u_i(t) = K_{p,i}\, e_i(t) + K_i \int_0^t e_i(\tau)d\tau + K_{d,i}\frac{de_i(t)}{dt} \quad (21)$$

$$e_i(t) = v_{i,desired}(t) - v_i(t) \quad (22)$$

This closed-loop adjustment allows the robot to recover from minor perturbations within a few hundred milliseconds, showing the responsiveness of the PID algorithm. Although PID control guarantees stable operation for our prototype and finds an optimal balance between performance and computational complexity, it is also limiting in more sophisticated situations. The controller can suffer from overshoot or reduced convergence in the presence of large disturbances, nonlinear payload fluctuations, or uneven ground. These findings set the agenda for future research investigating adaptive PID, Linear Quadratic Regulator (LQR), or sensor-fusion-based improvements to reach beyond controlled laboratory settings[27].

C. Kalman Filter Algorithm

The Kalman Filter is a recursive computation to produce an estimate of the state and error covariance in a dynamic system subject to noise and uncertainty. It works on two main steps: prediction and correction. In the prediction step, the algorithm propagates the current state and error covariance in time according to system dynamics. In the correction step, new measurements from sensors are used to update the estimate by reducing the mean square error. The algorithm produces a current estimate of state and covariance, a measure of confidence in the estimate. The algorithm runs in a loop for every time step, rendering the Kalman Filter computationally intensive enough for real-time use.

Despite being frequently employed in nonlinear systems, the Extended Kalman Filter (EKF) was not used in this study since the robot moves in a restricted angular region near the upright equilibrium position, for which system dynamics could be effectively modelled by means of the Linear Kalman Filter. Furthermore, the classical Kalman Filter requires less computational effort and smaller amounts of memory to store filter parameters; hence, it fits better for the real-time implementation on an ESP32 microcontroller. Experiments have shown that the Kalman Filter gave an adequate result in terms of state estimation and balanced operation; thus, employing EKF would have been excessive for this task shown in Fig.3.

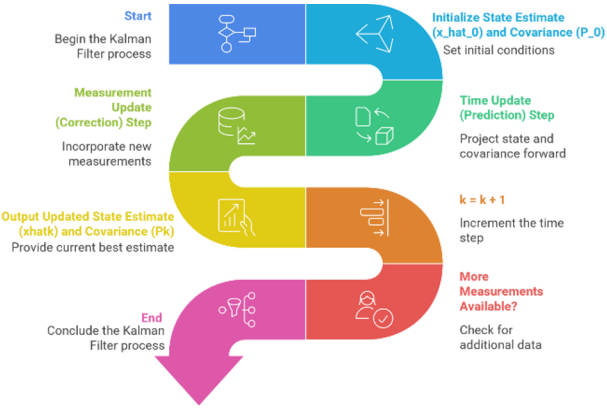

Fig.3 Kalman filter algorithm flow process

D. Hardware Implementation

The suggested hardware configuration of the self-balancing robot is around an ESP32 microcontroller that serves as the brain, communicating with sensors, drivers, and actuators for dynamic stability. An MPU-6050 inertial measurement unit (IMU) with a three-axis accelerometer and gyroscope gives real-time measurements of tilt and angular velocity, which are input into the ESP32 using the I2C interface[28].

ESP32 was chosen as the main microcontroller due to its high-performance capacity, double core CPU, integrated Wi-Fi and Bluetooth connectivity, reduced energy consumption, and sufficient number of GPIO pins for sensor input and real-time motor control. All the above-mentioned parameters allow the implementation of the PID controller with reduced computation time. MPU-6050 IMU is used since it includes a 3-axis accelerometer and a 3-axis gyroscope in one low-cost module, allowing obtaining tilt angle and angular velocity measurements that will be used in dynamic balancing. According to the Fig. 4, the communication between the MPU-6050 IMU and the ESP32 is carried out through the I^2C bus (SDA and SCL). The ESP32 processes data from the sensors with a

complementary filter and PID controller and sends STEP and DIR signals to A4988 motor drivers, which control NEMA 17 stepper motors. Moreover, the ESP32 interacts with the servo motor, LEDs, and buzzer for other purposes, and all components have the same ground, powered by 5 V and 12 V power sources.

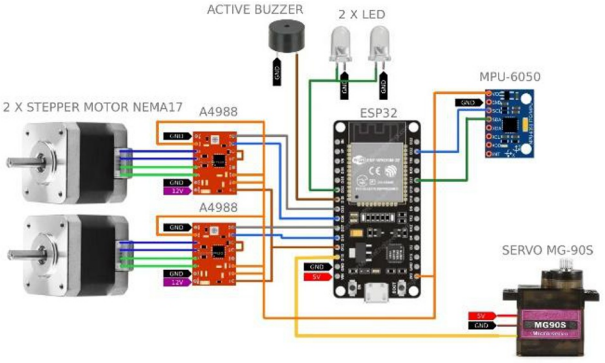

Fig.4 Connections of self-balancing robot

The control algorithm is made by means of a Proportional–Integral–Derivative (PID) algorithm that computes the correction actions as a function of the robot's posture deviation from the vertical reference. The corrective signals are sent as step and direction pulses to two A4988 stepper motor drivers that accurately control the NEMA 17 stepper motors, allowing precise wheel adjustments necessary for real-time balancing. Along with the balancing mechanism, auxiliary components like an MG90S servo motor, active buzzer, and LEDs are incorporated to provide auxiliary actuation, audio alerts, and visual indication respectively. The power distribution is handled by a dual supply system, where the motors are powered by a 12 V supply and the ESP32, IMU, servo, and indicators are powered by a regulated 5 V supply, both sharing a common ground to provide stability. This arrangement not only supports efficient deployment of closed-loop balance control but also offers a modular platform that is easily customizable for sophisticated functionalities like obstacle detection, wireless communication, and AI-based control, thus making it relevant to both research purposes in academia and practical robotic applications.

IV. RESULTS AND DISCUSSIONS

In order to prove the performance of the designed self-balancing robot, a series of systematic tests were carried out under various operating conditions shown in Fig.5.

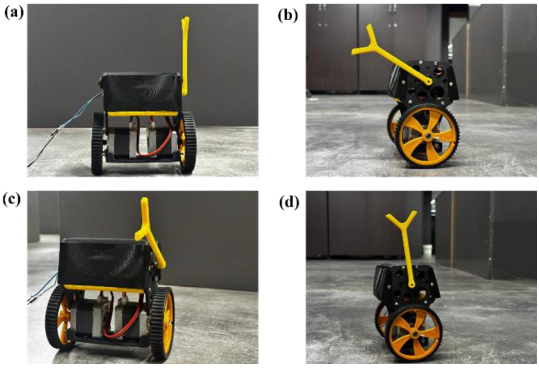

Fig.5 Self Balancing Robot

The experiments were structured to test the robot's capability to achieve static balance, its durability towards external disturbances, and its adaptability to various terrains. Each experiment was repeated several times to check

for reproducibility, and the observations were noted in terms of stabilization time, recovery from disturbances, and trajectory tracking ability. The next subsections outline the results.

A. PID Tuning Process

The focal point of the balancing system would be PID controller. Implementation of PID algorithm is the most obvious portion, yet it is also crucial to tune PID parameters to the optimal; K_p(Proportional), K_i(Integral), and K_d(Derivative). Our original values were more-or-less arbitrarily taken off more recent scholarly literature and publicly shared codebases are K_p= 25, K_i = 0.8, K_d =10. These principles were repeated numerous times. We tried a trial-and-error approach, and we changed the value of one parameter, and we saw the reaction of the robot. Proportional gain (K_p) reigns over the ferocity of responses of the robot to the existing tilt errors. An excessively high setting will result in the robot overshooting making it to be swaying and a low setting will result into a robot being sluggish. Integral gain (K_i) helps to eliminate a cumulative error in a long term, e.g. long run drift. Integral windup however results in instability of high K_i values. Derivative gain (K_d) damps because it is a response to a rate of change of the error. This expression confines the overshooting and makes the robot to move smoothly. The systematic testing showed that the oscillation decreased drastically at higher K_d, showed good recovery at medium K_p and low Ki, and rapid and consistent recovery. Automatic techniques such as auto-tuning such as Ziegler-Nichols were not used since in the practical world mechanical faults may take place and consequently manual tuning is necessary in order to be capable of functioning optimally. In order to check the tuning, we applied perturbations (gently pushing or other uneven floor initiations) and established recovery time and overshoot. We did that at various loads, battery capacity, among others, to determine the most stable PID parameters.

PID control parameters (K_p, K_i, and K_d) were found experimentally by tuning. These parameters were adjusted iteratively while running the system in order to achieve a stable output signal without much overshoot, steady-state error, and very short time for stabilizing. Values of these parameters were chosen according to optimal control system performance while operating the ESP32-based balancing system reliably.

B. Control Design

The graphical output presented above illustrates the sensor measurements, filtered estimation, and control actions of the self-balancing robot being controlled under PID control with optimized parameters (K_p = 25, K_i = 0.8, K_d = 10). The plots of accelerometer and gyroscope angles show the raw tilt measurements, where the accelerometer offers long-term stability but is subject to noise and the gyroscope records smooth short-term dynamics but is plagued by drift. By integrating these signals, the filtered angle plot provides a consistent and precise tilt estimation required for balance control reliability. The omega plot indicates the angular velocity response, capturing robot rotation dynamics upon corrective action. The PWM and U control action plots indicate the controller's effort as motor pulses, illustrating how the PID algorithm continuously varies motor torque to counteract tilting and reestablish equilibrium. The smooth and responsive control curves show that oscillations are well damped, overshoot is reduced, and recovery is fast. Overall, these findings validate the fact that the PID controller, being simple in nature, gives stable real-time stabilization to the self-balancing robot shown in Fig.6.

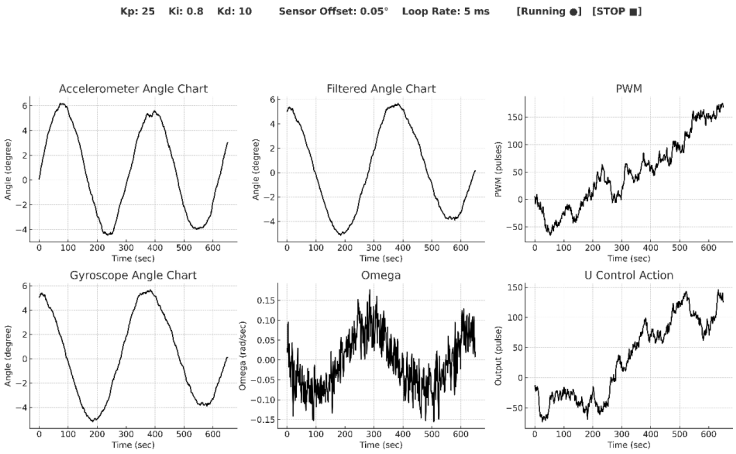

Fig.6 Control design results

C. Static Balance Ability Comparison Test

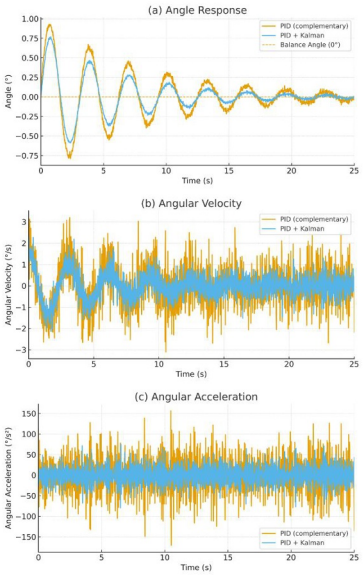

Fig.7 Static stability of self-balancing robot (a) Angle Response, (b) Angular velocity (c) Angular acceleration

Plots of the self-balancing robot under a traditional PID controller and a PID controller with a Kalman filter clearly demonstrate the benefits of sensor fusion for dynamic stabilization. Figure 7. (a) illustrates the angle response, where the Kalman-based system settles quicker and exhibits lesser oscillatory response about the balance point than the complementary filter–based PID. Figure (b) gives the angular velocity, and it is apparent that the Kalman filter attenuates high-frequency noise, leading to less violent velocity profiles and better damping

of the oscillations. Also, Figure (c) shows the angular acceleration where the Kalman filter produces much smaller spikes, validating smoother corrective actions and less control effort. The results reveal that although traditional PID delivers adequate balance, the use of Kalman filtering enhances noise robustness, stability, and overall smoothness of control, especially in the presence of disturbances and prolonged operation.

D. Horizontal Road Braking & Static Balance Test

The underlying theory behind the images demonstrates how the self-balancing robot reacts to acceleration, braking, and steady-state manoeuvres on a flat road. The initial plot demonstrates the tilt angle in terms of the vertical, where both PID and PID along with a Kalman filter are trying to reduce deviations around the balance point. Although the PID-only controller has greater oscillations and drift with sudden braking, the Kalman-enhanced PID has lower angle excursions and more stable convergence to upright posture. The second graph indicates the angular velocity, which is the robot's corrective motion. Here, traditional PID induces high-frequency oscillations while correcting for braking forces, while the Kalman-based PID provides smooth velocity transitions with lower noise due to improved state estimation. The third plot illustrates angular acceleration, where the plain PID creates steep spikes, representing sudden motor torque changes, and these spikes are smoothened out by the Kalman-enhanced method, which captures smoother corrections. The last plot illustrates true speed of movement versus target speed profile, where it is apparent that PID by itself fails to track exact sudden deceleration and acceleration steps, whereas the Kalman-augmented PID tracks the desired path more efficiently. In general, these findings indicate that the inclusion of a Kalman filter enhances static balance and braking stability through decreased sensitivity to noise, minimizing overshoot, and smoother transitions under dynamic horizontal movement shown in Fig.8.

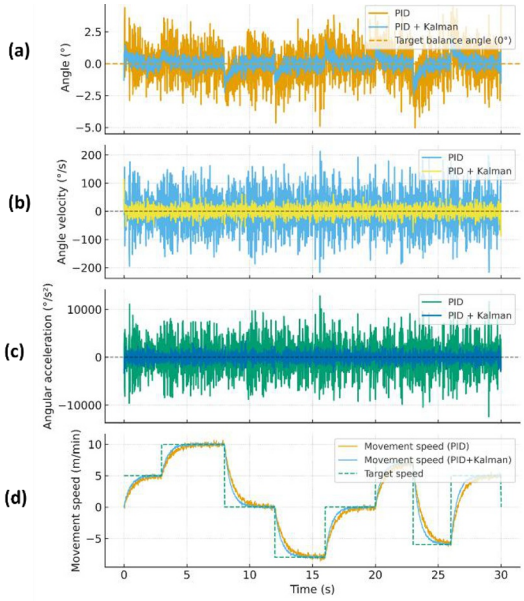

Fig.8 Horizontal road breaking test (a) Angle, (b) Angle velocity, (c) Angular acceleration, (d) Movement speed

E. Uniform & Balanced Movement On Horizontal & Slope

The figure.9 shows the self-balancing robot speed tracking performance on a 23° inclined plane, with the conventional PID controller versus the PID augmented with a Kalman filter. Three target speeds-1.2 m/min, 4.0 m/min, and 5.6 m/min-were tested under the same conditions. The results indicate that although the simple PID controller could track commanded speeds, it had slower rise times, greater steady-state errors, and more pronounced oscillations in both acceleration and braking cycles. Compared to this, the Kalman-augmented PID tracking's were much closer to target profiles with faster stabilization, less noise, and smoother transition. This

enhancement is due to the Kalman filter's optimal blending of gyroscope and accelerometer measurements, thus giving cleaner state estimates to the controller. Consequently, the robot showed more precise control and resilience against the added gravitational load caused by the slope. These results validate that combining Kalman filtering with PID not only improves tracking performance but also enhances static balance under inclined motion, an important consideration for practical use of self-balancing robots

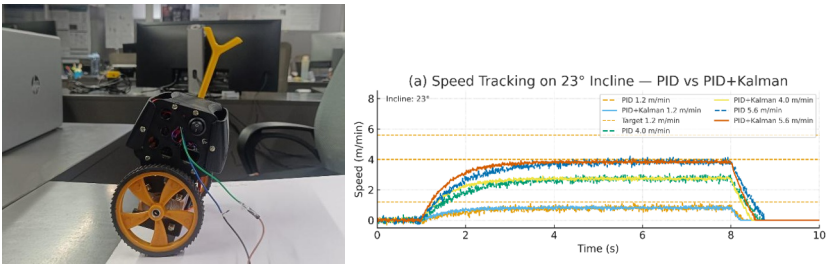

Fig.9 Speed tracking on 23° Incline

Demonstration video of the developed self-balancing robot is available at: https://github.com/kvganduri/Self-Balancing-Robot/tree/main

CONCLUSION

The project was able to prove the design, development, and verification of an inverted pendulum model two-wheeled self-balancing robot using a PID control method. Through the integration of the ESP32 microcontroller, MPU6050 IMU sensor, A4988 motor drivers, and NEMA 17 stepper motors into a light-weighted 3D-printed chassis, the system was able to perform dynamic stabilization in real time successfully. Systematic testing verified that the robot was able to achieve stability between 1.5–2 seconds from start-up and recover from medium-sized disturbances in under 500 milliseconds. The PID parameters tuned (K_p = 25, K_i = 0.8, K_d = 10) provided optimum responsiveness-trade-off with smooth damping and reaffirmed the robustness of PID control even on limited-resource embedded platforms. The findings highlight that, as simple as it may be, PID is still a useful and efficient solution for dynamic stabilization problems in robotics, particularly in prototyping and educational settings. In addition to demonstrating the possibility of low-cost self-balance systems, this project is also an educational platform from which to explore sensor fusion, embedded programming, and closed-loop control concepts. Notably, it demonstrates the applicability of the inverted pendulum problem in everyday technology, from Segways to assistive robots and delivery platforms. Future enhancements could include incorporation of Kalman filters for sophisticated sensor fusion, LQR or AI-enabled control programs for adaptive learning, and wireless/mobile interfaces for remote monitoring and control. Extension of the system with obstacle detection, terrain adaptation, and autonomous navigation would significantly increase its relevance for application in real-world scenarios. This work not only confirms the effectiveness of PID-based stabilization but also provides a solid basis for future research and development on cost-effective, quick-response, and smart robotic systems.

REFERENCES

[1] S. Wang, W. M. Lim, J.-H. Cheah, and X.-J. Lim, “Working with robots: Trends and future directions,” *Technol. Forecast. Soc. Change*, vol. 212, p. 123648, Mar. 2025, doi: 10.1016/j.techfore.2024.123648.

[2] M. Javaid, A. Haleem, R. P. Singh, and R. Suman, “Substantial capabilities of robotics in enhancing industry 4.0 implementation,” *Cogn. Robot.*, vol. 1, pp. 58–75, 2021, doi: 10.1016/j.cogr.2021.06.001.

[3] S. S. Kotha *et al.*, “Next generation legged robot locomotion: A review on control techniques,” *Heliyon*, vol. 10, no. 18, p. e37237, Sep. 2024, doi: 10.1016/j.heliyon.2024.e37237.

[4] M. Dhanda, B. A. Rogers, S. Hall, E. Dekoninck, and V. Dhokia, “Reviewing human-robot collaboration in manufacturing: Opportunities and challenges in the context of industry 5.0,” *Robot. Comput. Integr. Manuf.*, vol. 93, p. 102937, Jun. 2025, doi: 10.1016/j.rcim.2024.102937.

[5] J. Chen, N. He, Z. Xu, M. Dou, and L. He, “Design and Implementation of a Novel Two-Wheeled Composite Self-Balancing Robot for Stationary Operations in Unknown Terrain,” *IEEE Access*, vol. 13, no. May, pp.

86032–86045, 2025, doi: 10.1109/ACCESS.2025.3569325.

[6] A. Abdelgawad, T. Shohdy, and A. Nada, “Model- and Data-Based Control of Self-Balancing Robots: Practical Educational Approach with LabVIEW and Arduino,” *IFAC-PapersOnLine*, vol. 58, no. 9, pp. 217–222, 2024, doi: 10.1016/j.ifacol.2024.07.399.

[7] H. P. Santoso, J. Slamet Saputro, H. Maghfiroh, and M. M. Marta Dinata, “Balancing Robot Navigation with Virtual Map and Virtual Sensor,” *Proceeding - 2020 Int. Conf. Radar, Antenna, Microwave, Electron. Telecommun. ICRAMET 2020*, pp. 218–223, 2020, doi: 10.1109/ICRAMET51080.2020.9298584.

[8] R. Shah, A. S. A. Doss, and N. Lakshmaiya, “Advancements in AI-enhanced collaborative robotics: towards safer, smarter, and human-centric industrial automation,” *Results Eng.*, vol. 27, p. 105704, Sep. 2025, doi: 10.1016/j.rineng.2025.105704.

[9] A. K. Kashyap and D. R. Parhi, “Particle Swarm Optimization aided PID gait controller design for a humanoid robot,” *ISA Trans.*, vol. 114, pp. 306–330, 2021, doi: 10.1016/j.isatra.2020.12.033.

[10] O. Boubaker and R. Iriarte, Eds., *The Inverted Pendulum in Control Theory and Robotics: From theory to new innovations*. Institution of Engineering and Technology, 2017. doi: 10.1049/PBCE111E.

[11] R. Petcu and G.-D. Andreescu, “Two-Wheeled Inverted-Pendulum Self-Balancing Robot with LQR or PID Control,” in *2022 IEEE 10th Jubilee International Conference on Computational Cybernetics and Cyber-Medical Systems (ICCC)*, IEEE, Jul. 2022, pp. 000345–000350. doi: 10.1109/ICCC202255925.2022.9922862.

[12] K. V. Ganduri and B. P. Pathri, “Swarm Intelligence in Action: Particle Swarm Optimization and Rendezvous Algorithms for Swarm Robotics,” *J. F. Robot.*, Nov. 2024, doi: 10.1002/rob.22466.

[13] G. Yu, “PSO-based Fuzzy Control of a Self-Balancing Two-Wheeled Robot,” pp. 1–5, 2017.

[14] J. Qiu *et al.*, “Two-wheeled self-balancing robot modeling and nonlinear control,” *2017 14th Int. Conf. Ubiquitous Robot. Ambient Intell. URAI 2017*, pp. 734–739, 2017, doi: 10.1109/URAI.2017.7992813.

[15] B. Karthika and V. R. Jisha, “Nonlinear Optimal Control of a Two Wheeled Self Balancing Robot,” *2020 5th IEEE Int. Conf. Recent Adv. Innov. Eng. ICRAIE 2020 - Proceeding*, vol. 2020, pp. 1–6, 2020, doi: 10.1109/ICRAIE51050.2020.9358385.

[16] B. Prajwal Pathri, V. S. H. Borra, K. V. Ganduri, J. M. C. K. Reddy, M. R. Kantipudi, and M. R. Jupalle, “Smart Hostel Management Through the IoT: Reducing Energy Consumption, Securing Access, and Preventing Theft,” in *2024 3rd International Conference on Automation, Computing and Renewable Systems (ICACRS)*, IEEE, Dec. 2024, pp. 483–489. doi: 10.1109/ICACRS62842.2024.10841498.

[17] S. N. Reddy Gottam, B. P. Pathri, K. vamshi Ganduri, T. L. Venkata Prasanna Sathwik, S. T. Reddy, and V. Vaishnavi, “EcoSprout: Machine Learning-Powered Smart Sprinkler System,” in *2024 International Conference on Emerging Smart Computing and Informatics (ESCI)*, IEEE, Mar. 2024, pp. 1–7. doi: 10.1109/ESCI59607.2024.10497210.

[18] A. Chawla and R. Jain, “Developing a Novel Control System for a Two-wheeled Self-balancing Robot using a Dead-weight Pendulum,” *2024 IEEE 3rd World Conf. Appl. Intell. Comput. AIC 2024*, pp. 1192–1197, 2024, doi: 10.1109/AIC61668.2024.10730959.

[19] J. Shakthi Prakash, E. M. Tharun, A. Joseph, V. Elango, M. Madhan, and S. Santhosh, “Smart Embedded Design for Android Application Based Control of Self Balancing Robot,” *2023 Annu. Int. Conf. Emerg. Res. Areas Int. Conf. Intell. Syst. AICERA/ICIS 2023*, pp. 1–5, 2023, doi: 10.1109/AICERA/ICIS59538.2023.10420109.

[20] B. Bekkar and K. Ferkous, “Design of Online Fuzzy Tuning LQR Controller Applied to Rotary Single Inverted Pendulum: Experimental Validation,” *Arab. J. Sci. Eng.*, vol. 48, no. 5, pp. 6957–6972, May 2023, doi: 10.1007/s13369-022-06921-3.

[21] T. Nikita and K. T. Prajwal, “PID Controller Based Two Wheeled Self Balancing Robot,” *Proc. 5th Int. Conf. Trends Electron. Informatics, ICOEI 2021*, pp. 1–4, 2021, doi: 10.1109/ICOEI51242.2021.9453091.

[22] R. Özalp, N. K. Varol, B. Taşci, and A. Uçar, “A Review of Deep Reinforcement Learning Algorithms and

Comparative Results on Inverted Pendulum System," 2020, pp. 237–256. doi: 10.1007/978-3-030-49724-8_10.

[23] D. Chen, Y. Guan, and L. He, "A Self-Balanced Essboard-like Mobile Robot – Essbot," no. December, pp. 160–165, 2019.

[24] P. Prongphimai, L. Poolperm, S. Chaiwas, and U. Kaewmorakot, "Development of a prototype of Self Balancing Robot for Education," *6th Int. STEM Educ. Conf. iSTEM-Ed 2021*, pp. 1–4, 2021, doi: 10.1109/iSTEM-Ed52129.2021.9625106.

[25] R. P. Borase, D. K. Maghade, S. Y. Sondkar, and S. N. Pawar, "A review of PID control, tuning methods and applications," *Int. J. Dyn. Control*, vol. 9, no. 2, pp. 818–827, Jun. 2021, doi: 10.1007/s40435-020-00665-4.

[26] G. A. R. Ibraheem, A. T. Azar, I. K. Ibraheem, and A. J. Humaidi, "A Novel Design of a Neural Network-Based Fractional PID Controller for Mobile Robots Using Hybridized Fruit Fly and Particle Swarm Optimization," *Complexity*, vol. 2020, 2020, doi: 10.1155/2020/3067024.

[27] C. R. Çırak and H. Çalık, "Hotspots in maximum power point tracking algorithms for photovoltaic systems – A comprehensive and comparative review," *Eng. Sci. Technol. an Int. J.*, vol. 43, p. 101436, Jul. 2023, doi: 10.1016/j.jestch.2023.101436.

[28] A. F. M. Sulaiman, I. Siradjuddin, and N. R. Dzakiyullah, "The Design of a Microcontroller-based Self Balancing Robot Employing PID Control and Kalman Filter," *Webology*, vol. 19, no. 1, pp. 1194–1206, 2022, doi: 10.14704/web/v19i1/web19081.