UNIVERSITY OF TURKU

Design and Analysis of a Self-Balancing Motorcycle

Department of Mechanical and Materials Engineering

Bachelor's Thesis

Author:
Aatu Rantanen

9.6.2026
Turku

Bachelor's thesis

Subject: Mechanical Engineering

Author(s): Aatu Rantanen

Title: Design and Analysis of a Self-Balancing Motorcycle

Supervisor(s): Anam Tahir, D.Sc. Tech.

Number of pages: 31 pages

Date: 9.6.2026

Abstract

As the urban environment becomes increasingly congested, the spatial efficiency of two-wheeled vehicles makes them an attractive alternative to conventional four-wheeled transportation. However, motorcycles exhibit an inherent physical limitation, such as being unstable when stationary. This thesis investigates the design, analysis, and implementation of a feedback control system for a self-balancing motorcycle equipped with an inertia wheel. Operating as an underactuated inverted pendulum, the system maintains upright equilibrium by generating reaction torque through the rotation of the inertia wheel.

A Model-Based Design methodology is utilized to develop and validate the controller in two stages, such as (1) simulation and (2) real-time physical implementation. A Proportional-Derivative controller combined with a Low-Pass Filter is designed and evaluated using a digital twin developed in Simscape Multibody. The virtual prototyping process proves effective in identifying design issues such as coordinate-frame mismatches before hardware deployment.

Nevertheless, the transition from simulation to physical implementation highlights the influence of real-time conditions and unmodeled dynamics on system performance. While the near-ideal simulation achieves stable equilibrium with relatively low controller gains, the physical prototype requires significantly more aggressive tuning to compensate for gravity effects, sensor noise, wire-induced disturbances, and mechanical slippage between the inertia wheel and axle. The findings demonstrate the effectiveness of Model-Based Design for embedded control systems while emphasising that digital control performance remains fundamentally constrained by physical implementation factors.

Keywords: Arduino, motorcycles, pendulums, vehicle stability control systems, embedded systems

Kandidaatintutkielma

Koulutusohjelma, oppiaine: Konetekniikka

Tekijä(t): Aatu Rantanen

Otsikko: Itsetasapainottuvan moottoripyörän suunnittelu ja analyysi

Ohjaaja(t): Anam Tahir, D.Sc. Tech.

Sivumäärä: 31 sivua

Päivämäärä: 9.6.2026

Tiivistelmä

Kaupunkialueiden ruuhkautuessa kaksipyöräisten ajoneuvojen tilatehokkuus tekee niistä houkuttelevan vaihtoehdon perinteisille nelipyöräisille kulkuvälineille. Moottoripyörillä on kuitenkin luontaisia fyysisiä rajoituksia, kuten epävakaus ollessaan paikallaan. Tässä opinnäytetyössä tutkitaan vauhtipyörällä (engl. inertia wheel) varustetun itsetasapainottuvan moottoripyörän säädinjärjestelmän suunnittelua, analysointia ja toteutusta. Järjestelmä toimii aliohjattuna käänteisenä heilurina (engl. underactuated inverted pendulum) ja ylläpitää pystysuoraa tasapainoa tuottamalla reaktiovoiman vauhtipyörän pyörimisen avulla.

Ohjaimen kehittämisessä ja todentamisessa käytetään mallipohjaista suunnittelumenetelmää (engl. Model-Based Design) kahdessa vaiheessa: (1) simulointi ja (2) reaaliaikainen fyysinen toteutus. Suhteellis-Derivoiva (engl. Proportional-Derivative) ohjain yhdistettynä alipäästösuodattimeen suunnitellaan ja arvioidaan käyttämällä Simscape Multibody -ohjelmistossa kehitettyä digitaalista kaksosta. Virtuaalinen prototyyppausprosessi osoittautuu tehokkaaksi suunnitteluongelmien, kuten koordinaatistojen yhteensopimattomuuksien, tunnistamisessa ennen laitteiston käyttöönottoa.

Siirtyminen simuloinnista fyysiseen toteutukseen korostaa kuitenkin reaaliaikaisten olosuhteiden ja mallintamattomien dynaamisten ominaisuuksien vaikutusta järjestelmän suorituskykyyn. Vaikka lähes ihanteellinen simulointi saavuttaa vakaan tasapainotilan suhteellisen alhaisilla säätöparametreillä, fyysinen prototyyppi vaatii huomattavasti aggressiivisempaa säätöä kompensoimaan painovoiman vaikutuksia, anturikohinaa, johtojen aiheuttamia häiriöitä sekä vauhtipyörän ja akselin välistä mekaanista luistoa. Tulokset osoittavat mallipohjaisen suunnittelun tehokkuuden sulautetuissa ohjausjärjestelmissä ja korostavat samalla, että digitaalisen ohjauksen suorituskykyä rajoittavat perustavanlaatuisesti fyysiset toteutustekijät.

Avainsanat: Arduino, moottoripyörät, heilurit, ajonvakautusjärjestelmät, sulautetut järjestelmät

Contents

Abbreviations

MBD	Model-Based Design
PD	Proportional-Derivative
CAD	Computer-Aided Design
DoF	Degree of Freedom
DC	Direct Current
PWM	Pulse-Width Modulation
IoT	Internet of Things
IMU	Inertial Measurement Unit
PID	Proportional-Integral-Derivative
PI	Proportional-Integral

1 Introduction

1.1 Background

Researchers in robotics and automation have long been interested in developing and validating advanced control strategies for underactuated and unbalanced mechanical systems [1]. These naturally unstable systems have become a benchmark for making balancing algorithms and their use cases understandable [2]. One of the most attention-seeking subareas is research on self-balancing dynamics, more precisely, the balancing of an inverted pendulum, which, as a mechanical device and a physical phenomenon, is naturally unstable [2].

Compared to a simple pendulum, such as a pendulum clock [3], the difference is that in an inverted pendulum, the center of mass is located above the pivot point of the system, which in turn creates a naturally unstable equilibrium [2]. If it is left uncontrolled, the force of gravity takes over and creates torque about the center of mass, accelerating the system away from its vertical axis, eventually leading to collapse.

The inverted pendulum has attracted considerable attention due to its practical applications, most notably in various vehicular implementations. One well-known concept from the early 1900s, the Gyro Monorail [4], was a tram intended to travel along a single rail with one or more inertia wheels mounted perpendicular to the rail to overcome instability through gyroscopic forces. This never made it into serial production because developing costs were too high, maintenance tasks were too difficult, and the overall capacity to carry passengers was small compared to an ordinary subway [5].

However, in recent years, researchers and engineers have again shifted towards inverted pendulum-based and inertia wheel inverted pendulum-based vehicles, such as self-balancing motorcycles or other two-wheeled vehicles [6]. The interest seems to be in exploring techniques to balance the vehicle without relying on traditional steering geometry or contact forces external to the intended contact surface that provide traction. Vehicles such as the Segway balance themselves by actuating the wheel-

mounted motors using gyroscopic and tilt sensors, while some concepts of a self-balancing motorcycle have used a body-mounted inertia wheel with similar sensors.

The practical motivation for researching and developing self-balancing vehicles lies in the future of urban mobility. As global urbanization increases, so does traffic. Modern-day parking limitations have made traditional four-wheeled vehicles less efficient for single-occupant commuting, not only in cities with a population over a million but also in quiet urban areas.

Two-wheeled vehicles offer a reduced physical footprint and higher energy efficiency. However, they require continuous dynamic balancing by the rider, making them inherently less safe and inaccessible for portions of the population. By successfully implementing inertia-wheel stabilization, the spatial efficiency of a motorcycle can be combined with the stability of a four-wheeled vehicle when stationary.

Furthermore, in this thesis, the methodology presented in [7] is adapted and further developed using the selected Model-Based Design (MBD) workflow and Computer-Aided Design (CAD) based implementation. While the initial modeling framework and motorcycle dynamics are based on this tutorial, all simulations, implementations, and analyses were independently developed by the author using the Arduino Engineering Kit Rev 2, MATLAB, and Simscape.

The author declares that the Grammarly AI tool has been used to check and improve the language of the thesis, and the author has reviewed and corrected the text and takes full responsibility for the content.

1.2 Scope and Objective of the Thesis

The primary objective of this thesis is to design and implement an Arduino and MATLAB-based feedback controller that can balance a stationary, scaled-down motorcycle with the use of an inertia wheel. Deriving complex nonlinear equations of motion for the motorcycle is often time-consuming, and human errors are common. In this thesis, the MBD approach is employed, and a digital model with real geometry is developed. MBD shifts the focus from manual coding and equation derivation to simulation and automatic code generation, enabling iterative testing.

The aim is to understand the development of the controller, both in software and on the hardware itself, and to develop an alternative model that isn't based on manual calculation but instead on physical CAD modeling via the Simscape Multibody environment. By directly extracting physical quantities, such as mass, inertia, and rotational speed, from a 3D model, the software automatically infers the system's dynamics.

The scope of this project is limited to balancing the motorcycle in a stationary position using a Proportional-Derivative (PD) control design with a low-pass filter. An important task is also to analyse the differences between an ideal, digital simulation and the mechanically imperfect physical world. This thesis addresses the following research questions:

1. How can an MBD workflow be utilized to develop a self-balancing motorcycle system?
2. How can a stationary motorcycle be stabilized using an inertia wheel and a Proportional–Derivative (PD) controller combined with a low-pass filter?
3. What differences arise between the behavior of the simulation model and the physical motorcycle prototype?
4. Which physical and mechanical factors most significantly affect the balancing performance of the implemented system?

The logical progression of this thesis is visualized in Figure 1.

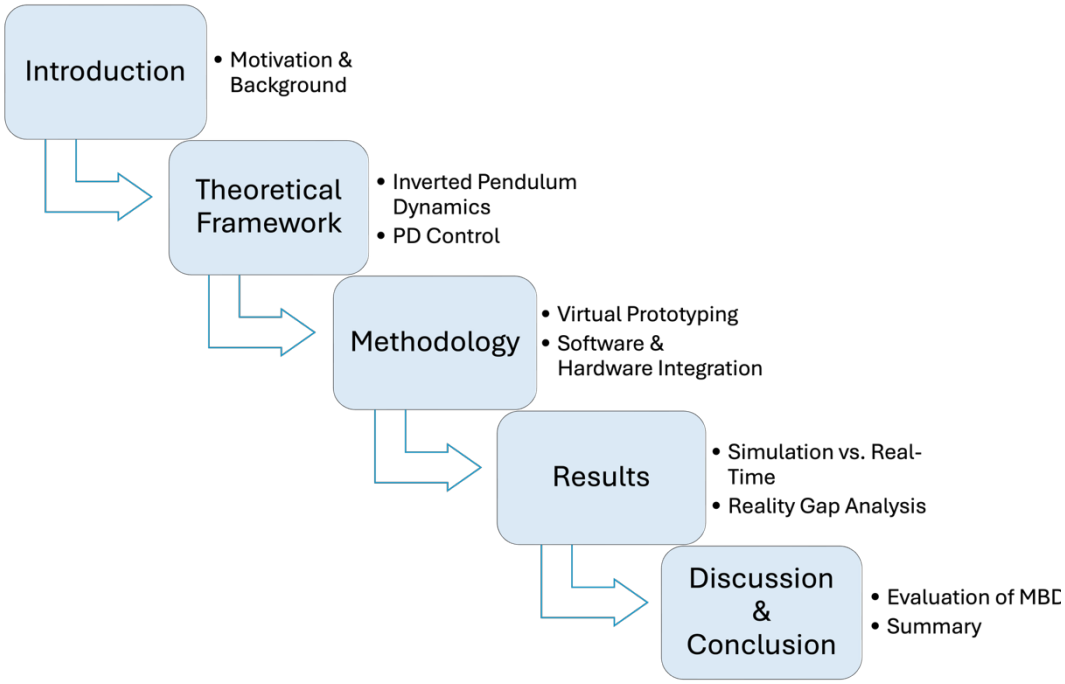

Figure 1. Structure of the thesis

2 Theoretical Framework

2.1 Self-Balancing Dynamics

The self-balancing motorcycle can be considered as an inverted pendulum, as shown in Figure 2. Its frame and wheels form a pendulum rod l_{AD} with one Degree of Freedom (DoF). Attached to the frame via a rotational axis C of a DC motor is an inertia wheel, which also has one DoF. This system is inherently unstable and underactuated because it has one more DoF than actuators [8], [9]. The core challenge lies in the physics of the inverted pendulum because it consists of a mass with its center of gravity B positioned above the pivot point A, which in this case is the axis parallel to the contact patch of the tyres [7]. Unlike traditional, four-wheeled vehicles, the motorcycle cannot maintain its equilibrium without an active controller that intervenes when lateral tilt is detected [10].

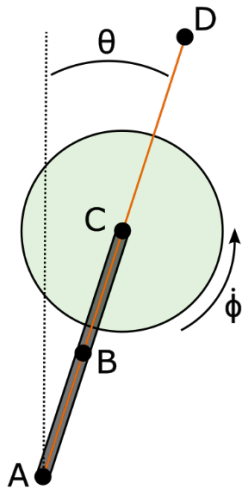

Figure 2. The motorcycle, based on an inverted pendulum approach [7]

When rested in an upright position, the lean angle of the pendulum, defined as the angular deviation from the true vertical axis, should be zero ($\theta = 0$). To dynamically track this state, the control loop also monitors lean velocity ($\dot{\theta}$), which represents the rate of change of the lean angle over time, and wheel speed ($\dot{\phi}$), which is the rotational velocity of the inertia wheel mounted to the actuator shaft. Even a slight angular deviation from the vertical axis can create a gravitational torque

$$\tau_{gr} = m_r g l_{AB} \sin\theta, \tag{1}$$

where m_r is the rod's mass, g is gravitational acceleration, and l_{AB} is the rod's length from A to B. The assumption is that gravitational torque acts only on the center of mass and will continuously accelerate the system away from equilibrium. To counteract this unavoidable instability, the self-balancing motorcycle uses inertia wheel stabilization to maintain equilibrium [8], which is governed by the conservation of angular momentum. An inertia wheel is connected to a Direct Current (DC) motor, and stabilization is achieved through rotational motion.

In a closed system, the total angular momentum L, is defined as the product of the Moment of Inertia I and lean velocity $\dot{\theta}$

$$L = I\dot{\theta}, \tag{2}$$

and it must remain constant. When the motorcycle's frame experiences a lateral disturbance and deviates from equilibrium, it acquires angular momentum. To maintain zero angular momentum, the DC motor must apply torque to accelerate the inertia wheel in the same direction as the deviation. With Newton's third law of rotation, accelerating the inertia wheel generates a reaction torque in the opposite direction [1], [11].

An inertia wheel with higher mass or a larger radius has a higher moment of inertia, allowing it to generate greater reaction torque for the same acceleration. When the inertia wheel spins, it, in turn, generates angular acceleration $\ddot{\theta}$, which is directly proportional to the inertia wheel's balancing power [8]. By Newton's second law for rotation, torque is the time derivative of angular momentum

$$\tau = \frac{dL}{dt} = I\ddot{\theta}. \tag{3}$$

The change in reaction torque is transferred to the entirety of the motorcycle's frame, which pushes the frame back towards its equilibrium. By controlling and optimizing this angular acceleration, the system can produce an equal and opposite torque to the gravitational pull and then tilt the motorcycle back to its upright position [10].

It is critical to note that inertia wheel systems are limited by DC motor saturation. Because torque is only generated is when $\ddot{\theta} \neq 0$, a continuous disturbance will force the

motor to spin the wheel faster and faster [12]. When the motor reaches its maximum rotational speed, it can no longer generate angular acceleration and, at that point, produce reaction torque; the system will inevitably collapse.

To achieve a full understanding of the system dynamics, the Euler-Lagrange equations are preferred as the basis for energy-based modeling of the frame and wheels [1], [4], [6], [10]. The Lagrangian operator L is expressed as the difference between the kinetic energy K and the potential energy P, with their generalized coordinates as

$$L = K - P. \tag{4}$$

The total kinetic energy is composed of the translational and rotational components of both the frame and the inertia wheel, where the lean angle is θ, the wheel rotation is ϕ, and l_{AB} and l_{AC} are the corresponding lengths from the inverted pendulum. m_r and m_w are the corresponding masses of the pendulum rod and the inertia wheel. The kinetic energy can be expressed as

$$K = \frac{1}{2}(m_r l_{AB}^2 + m_w l_{AC}^2 + I_{AB} + I_{AC})\dot{\theta}^2 + I_{AC}\dot{\theta}\dot{\phi} + \frac{1}{2}I_{AC}\dot{\phi}^2. \tag{5}$$

The potential energy is expressed as

$$P = (m_r l_{AB} + m_w l_{AC})g\cos\theta, \tag{6}$$

where g represents gravitational acceleration.

The Euler-Lagrange equation of motion is based on a formula

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_i, \tag{7}$$

where q_i represents the generalized coordinates and Q_i the forces not accounted for K and P. With equations 4, 5, and 6, the Euler-Lagrange equation of motion can be calculated, and the following equations can be derived

$$(m_r l_{AB}^2 + m_w l_{AC}^2 + I_{AB} + I_{AC})\ddot{\theta} + I_{AC}\ddot{\phi} - (m_r l_{AB} + m_w l_{AC})g\cos\theta = 0$$

$$I_{AC}\left(\ddot{\theta} + \ddot{\phi}\right) = T_r, \tag{8}$$

where T_r is the torque that the electric motor applies to the inertia wheel [1], [6].

To conclude the theoretical derivation of the motorcycle's dynamics, the Lagrangian results are organized into a state-space format [1], [10]. The nonlinear formulations are linearized about the unstable equilibrium point ($\theta = 0$) using small-angle approximations [4]. The inverted pendulum wheel is derived into a controllable linear state-space form as

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ b/a & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -b/a & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ -1/a \\ 0 \\ (a + I_{AC})/(a - I_{AC}) \end{bmatrix} T_r, \tag{9}$$

where $a = m_r l_{AB}^2 + m_w l_{AC}^2 + I_{AB}$ depicts the total moment of inertia at the centre of mass and $b = (m_r l_{AB} + m_w l_{AC})g$ is the torque response factor [10].

Mass and center of gravity are constants of the motorcycle dynamics, so they can be directly extracted from the CAD geometry to formulate the governing equations of the data-driven model [7]. These fixed parameters and performance variables are shown in Tables 1 and 2.

Table 1. Fixed parameters of the self-balancing motorcycle

Symbol	Definition	Unit	Value
m_r	Mass of the pendulum rod	kg	0.2948
m_w	Mass of the inertia wheel	kg	0.0750
m_b	Mass of the battery	kg	0.0440
m_{fw}	Mass of the front wheel	kg	0.0300
m_{rw}	Mass of the rear wheel	kg	0.0300
m_m	Mass of the motor	kg	0.0610
m_s	Mass of the steering fork	kg	0.0180
g	Gravitational acceleration	m/s^2	9.80665
l_{AB}	Length from A to B	m	0.0650
l_{AC}	Length from A to C	m	0.1300
l_{AD}	Length from A to D	m	0.1300

Table 2. Performance variables of the self-balancing motorcycle

Symbol	Definition	Unit
θ	Lean angle	rad
$\dot{\theta}$	Lean velocity of the pendulum	rad/s
$\ddot{\theta}$	Angular acceleration of the pendulum	rad/s^2
ϕ	Angle of inertia wheel	rad
$\dot{\phi}$	Angular velocity of the inertia wheel	rad/s
$\ddot{\phi}$	Angular acceleration of the inertia wheel	rad/s^2
I	Moment of Inertia	$kg \cdot m^2$
τ	Torque	$N \cdot m$

2.1.1 Proportional-Derivative Control

To stabilize the derived linear state-space system around its equilibrium point ($\theta = 0$), a PD control design is selected. The stabilizing reaction torque τ applied to the DC motor can be expressed as

$$\tau = K_p\theta + K_d\dot{\theta} \tag{10}$$

where K_p is the proportional gain, K_d is the derivative gain, θ is the lean angle, and $\dot{\theta}$ is the angular velocity of the inverted pendulum [11]. The proportional gain provides restorative force directly proportional to the lean angle, while the derivative term provides damping based on the rate of change of the lean angle.

2.2 Model-Based Design

MBD is a mathematical and visual method for addressing control problems by centering development around virtual prototyping and simulation instead of trial and error [13]. The development of embedded control systems has previously relied heavily on manual programming methodologies, which in turn have introduced a risk of syntax errors or damage in hardware [14], [15]. In the context of a self-balancing motorcycle, MBD allows the creation of a digital model using CAD geometry, a virtual model for simulation before any physical implementation.

The fundamental structure of the MBD approach is the V-Model, as illustrated in Figure 3, which describes the development workflow as a distinct V-shape. This model ensures that every part of the development is simulated and verified against the system constraints and requirements.

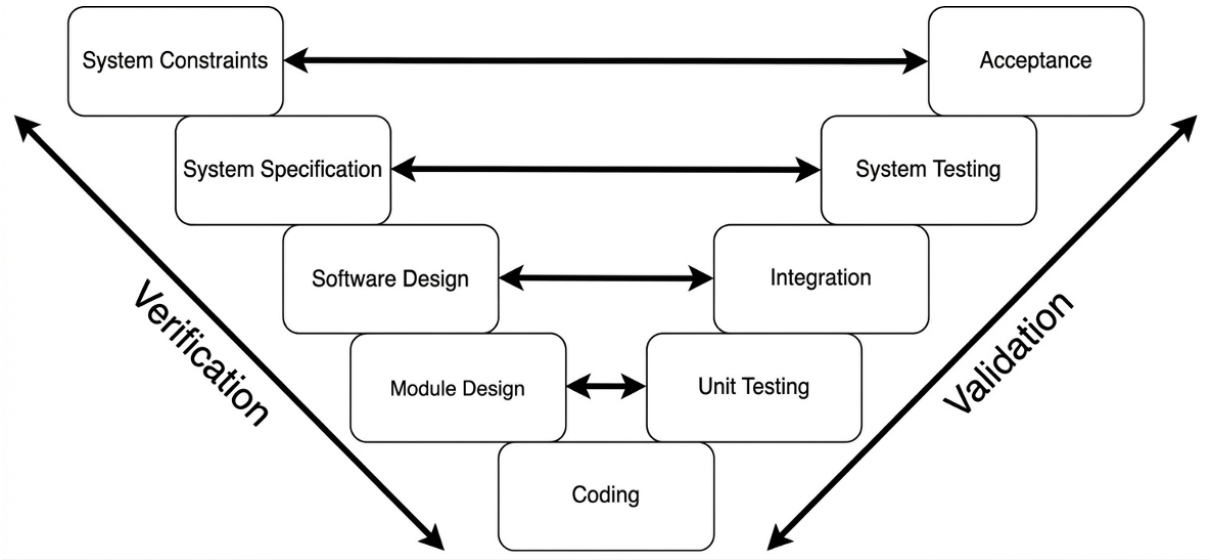

Figure 3. V-model of Model-Based Design [14]

In the MBD approach, specifications are set by creating a digital model of the system in the early stages of development [14]. Going down the V-Model, at the bottom, the feedback controller is refined in the Simulink environment to prevent damaging the physical model. Moving back up the model, the logic control is implemented on the real motorcycle and coupled with the Arduino, tested and evaluated against determined specifications, and simulations are performed.

An important decision within the MBD workflow is creating the detailed “digital twin”, a virtual model of the hardware. By simulating the real-time geometry, joints, and mass distribution of the self-balancing motorcycle, along with the physical limitations of the actuators, the performance of the feedback controller can be carefully tested before any code is deployed to the actual microcontroller. This approach, based solely on virtual iteration and tuning, drastically reduces development time and the risk of hardware damage [16].

In traditional control engineering, development requires hands-on mathematical modeling, where physical principles are translated into complex differential equations. As in Chapter 2.1, this involves deriving the Euler-Lagrange equations and expressing them in state-space form. This equation-based approach serves as a functional concept for feedback controller design. However, modern approaches that utilize MBD provide a different view of physical modeling [16]. Instead of using ordinary Simulink programming blocks to manually specify state-space equations, Simscape Multibody can model the system's dynamics directly from complex geometries and physical properties.

By importing 3D CAD geometry into the Simscape environment, the software can detect inertial properties from the files. This physical modeling bypasses time-consuming manual calculations of equations while handling the real-time geometry of the motorcycle's components far more accurately than analyses of traditional physical systems and simplified mathematical shapes.

2.2.1 Automatic Code Generation

A primary advantage of the MBD approach is the implementation of automatic code generation [14]. The traditional development of an inertia wheel inverted pendulum system requires extensive manual coding in C or C++ to handle precise control-loop timing and Pulse-Width Modulation (PWM) [10], [13], [14]. Through the MBD workflow, the feedback controller is developed visually using block diagrams within the Simulink environment.

Instead of relying on manual translation, the software's integrated support packages are used to automatically compile the control logic into executable C code for the Arduino microcontroller and the digital twin. This method significantly reduces the amount of manual programming errors and accelerates the rapid prototyping phase.

2.2.2 CAD Model

For certain applications, physical systems can be mathematically simplified into basic geometric shapes or point masses to ease the derivation of equations. However, real-time mechanical assemblies have complex mass distributions and asymmetrical

geometries that cannot be represented at the desired resolution with these basic simplifications.

Within the selected MBD framework, this contradiction is addressed by utilizing 3D models of the components to build the physical plant. By analysing the precise volume, mass, and placement of the designed components, Simulink can mathematically extract the exact inertial properties and center of gravity.

This CAD-based parameterization significantly reduces theoretical uncertainty. The gap between the physical hardware and the digital twin is minimized by grounding the model in exact geometric data, including mass distributions, rather than mathematical estimates. Consequently, the feedback controller can be tuned against possibly the most accurate representation of the system's true dynamic resistance to rotation, ensuring stability when the physical deployment is made. The 3D model of the self-balancing motorcycle is shown in Figure 4.

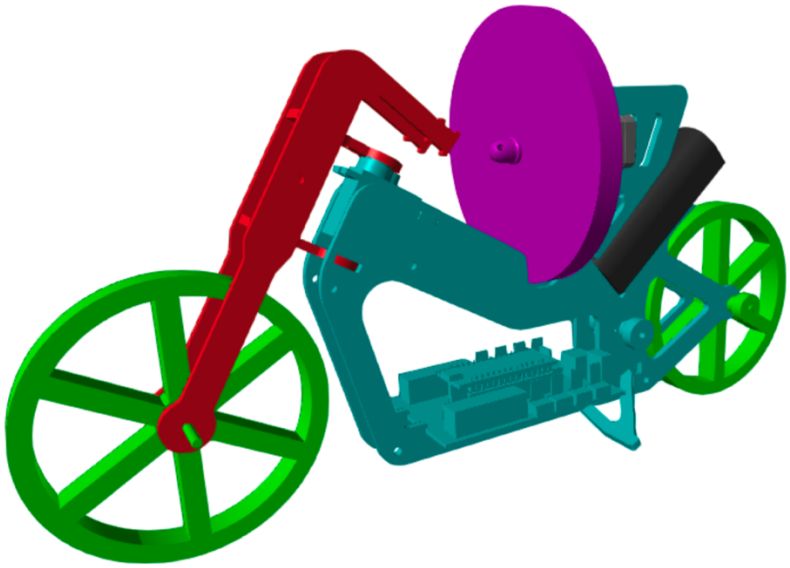

Figure 4. A CAD model of the self-balancing motorcycle

3 Methodology and Implementation

3.1 Hardware Components

The physical implementation of the self-balancing motorcycle is built using a set of components from the Arduino Engineering Kit Rev 2, selected to execute the control logic in real time. The hardware architecture's most important parts are the microcontroller, motor carrier, inertial measurement unit, and electromechanical actuators (DC motors).

The central computing unit for this project is the Arduino Nano 33 IoT, a microcontroller designed for Internet of Things (IoT) applications and mechatronic enhancements. The microcontroller has a 32-bit processor capable of executing compiled C code with minimal delay. To interface with the actuators, the microcontroller is mounted onto a motor carrier. This carrier board provides the power distribution to all components, motor driver electronics, and an integrated 9-axis Inertial Measurement Unit (IMU). The IMU, combining an accelerometer, gyroscope, and magnetometer, continuously monitors the physical state of the motorcycle frame and compares it against a calibrated zero point. The absolute lean angle θ is estimated by measuring the gravitational vector, while the lean velocity $\dot{\theta}$ is tracked by the gyroscope.

The mechanical actuation required to spin the actual inertia wheel is provided by a DC motor. To track its own behavior and close the feedback loop, the motor is factory-equipped with a built-in encoder, based on the Hall effect, which continuously measures the rotational position ϕ, and angular velocity $\dot{\phi}$ of the inertia wheel.

3.1.1 Sensor Fusion and Signal Processing

While the integrated 9-axis IMU provides the raw data required for the inverted pendulum's state vector, the individual microsensors within it have physical limitations that prevent their use without processing. The accelerometer is highly sensitive to the gravitational vector, but it is also disturbed by the DC motors around it and the chassis itself, which are observed to produce varying values when testing the IMU as stationary [7]. On the other hand, the gyroscope provides smooth measurements of angular

velocity $\dot{\theta}$, but because the angle is calculated by integrating this velocity over time, it could suffer from integral drift as well as interference from other components. In the gyroscope, small measurement errors accumulate into big angular inaccuracies.

To overcome these hardware limitations, the IMU needs to provide a clean, reliable angle signal for the feedback controller [10]. A filtered-derivative approach is used within the Simulink architecture to process the signals and derive the necessary velocity state without amplifying noise from the surroundings.

To successfully implement the full-state feedback controller, the system requires accurate real-time values for all four state variables ($\theta, \dot{\theta}, \phi, \dot{\phi}$). While the IMU and the motor encoder provide the raw positional data, the respective angular velocities must be derived from these discrete measurements.

A filtered derivative block is implemented in Simulink, which is based on a low-pass filter in connection with a standard derivative calculation. Using this block, the software successfully calculates the correct angular velocities while actively damping the inherent sensor noise.

3.2 Software

The control architecture is developed entirely in MATLAB and Simulink R2025b. To bridge the gap between high-level simulation software and the physically embedded Arduino system, support packages are used. These packages provide pre-configured graphical block sets that allow the software to interface directly with the microcontroller, actuators, and the motor carrier. Consequently, the controller can read 9-axis IMU data, track the encoder's position, and output the required PWM signals to the DC motor without manual programming.

3.3 Virtual Prototyping and Simulation

Before deploying the control algorithms onto the hardware, a virtual prototyping phase is constructed to safely observe the system's dynamics. Using Simscape Multibody, a virtual twin of the motorcycle is constructed from separate 3D files of the components, as shown in Figure 4. The CAD geometry of the system enables the software to

automatically extract inertial properties and mass distributions, bypassing the need for manual mathematical approximations.

To simulate the physical kinematics, virtual revolute joints are defined at the pivot point (the contact point of the wheels) and at the motor shaft that connects to the inertia wheel, as shown in Figure 5. Within the Simscape environment, these joints enable the measurement of relevant state variables, including angular position and angular velocity, which are subsequently used within the control system. The joints are configured to continuously output positional data, simulating the behavior of the IMU and the DC motor encoder.

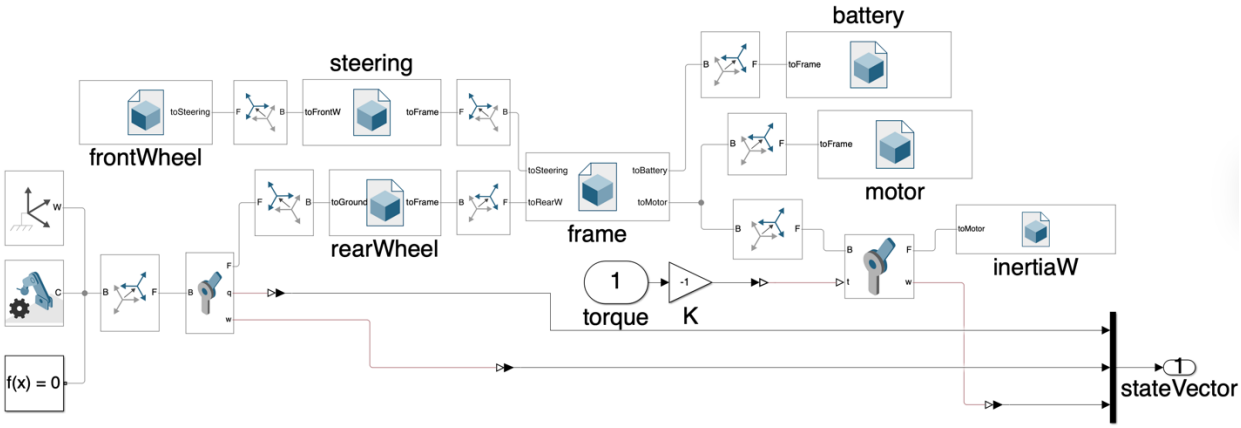

Figure 5. Physical Plant Model of the Virtual Prototype

The feedback controller is implemented with a virtual plant model, which enables monitoring of system response and supports iterative tuning of the PD controller, as shown in Figure 6. By reading the simulated tilt angles and applying torque to the simulated motor shaft, the controller's ability to balance the motorcycle can be evaluated purely in software and in various initial states, eliminating the risk of hardware damage during the initial development phase.

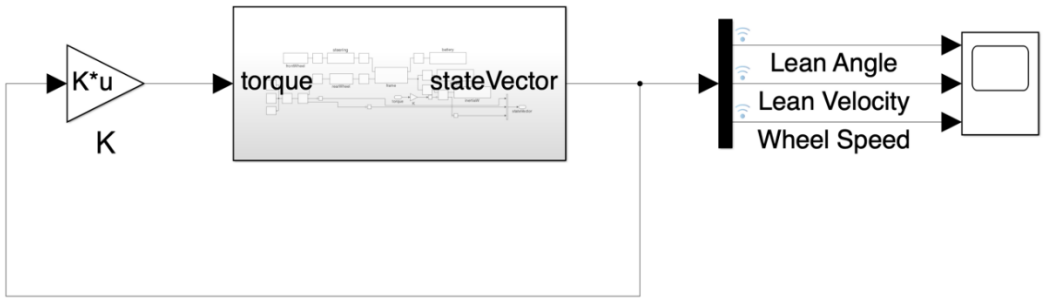

Figure 6. Virtual Plant of the Visual Model

3.4 Feedback Control Design

The control logic is developed using a PD design, as shown in Figure 7. To keep the motorcycle in an upright position, the proportional gain K_p is set much higher than in the visual model to react aggressively to lean-angle errors. However, the first physical tests reveal that the IMU data is heavily contaminated by vibrations from the spinning inertia wheel. To filter out high-frequency mechanical vibrations from polluting the sensitive derivative gain K_d, a filtered-derivative block is implemented in the sensor preprocessing subsystem. This acts as a low-pass filter, cleaning the $\dot{\theta}$ signal before it reaches the main controller loop, as shown in Figure 8.

Furthermore, a specific K_{pw} gain is implemented to prevent the DC motor from reaching saturation during constant physical disturbance. Without this parameter, steady-state errors would force the motor to accelerate until the motorcycle collapses. By feeding the inertia wheel's velocity $\dot{\phi}$ back into the system, K_{pw} acts as an active dampening factor without overrunning the commands of the K_p and K_d gains.

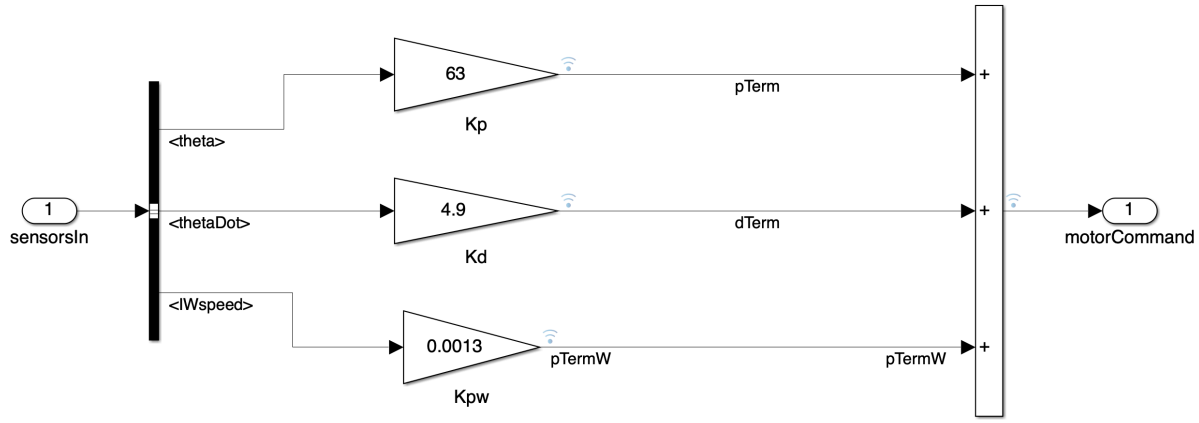

Figure 7. The PD control design

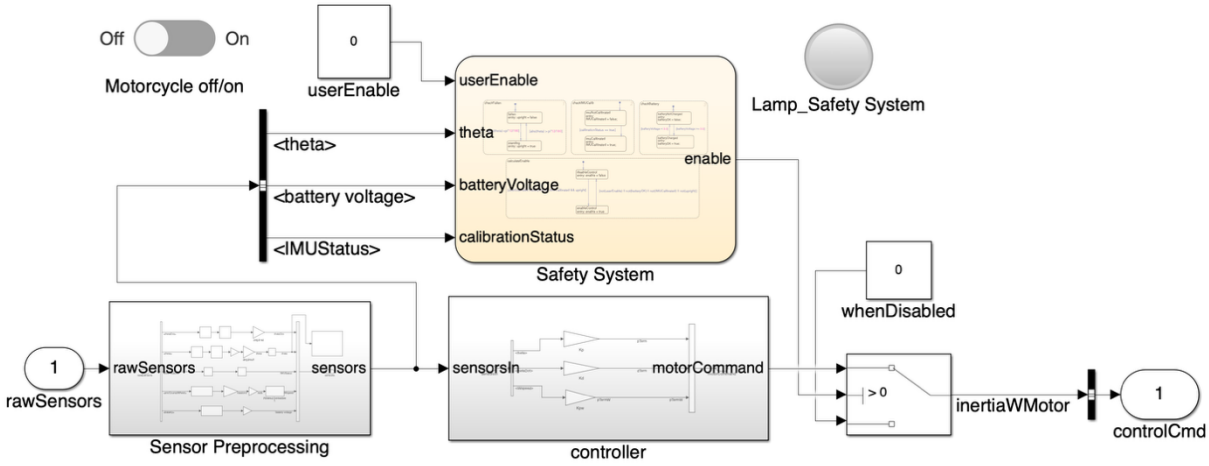

Figure 8. The overall proposed control architecture

4 Results

4.1 Virtual Prototype: Simulation Performance

Before deploying code to the physical hardware, the feedback control logic is first tested and tuned using the digital twin. This phase serves as the primary validation layer for observing the system's behavior and identifying structural discrepancies in the 3D geometry in a safe environment.

During the initial simulation runs, the virtual motorcycle fails to stay upright. Instead of balancing, the digital twin immediately rotates and hangs upside down beneath the virtual floor. This reveals a direct mismatch in the controller's coordinate frame. The mathematics behind the generated code assumes a specific rotational polarity for the reaction torque, whereas the CAD model's joint connecting the inertia wheel to the motor is oriented in the opposite direction. Consequently, when the controller attempts to actuate the inertia wheel to prevent a fall, it unintentionally applies a positive feedback force. This accelerates the digital twin away from its upright equilibrium and pushes it directly into a downward equilibrium. This coordinate mismatch is resolved by inverting the polarity of the PD gain matrix.

Once the polarity is corrected, achieving equilibrium introduces the concept of a "Reality Gap". Initial attempts to utilize PD gain values intended for real-time cause the system to crash immediately when the program is carried out. In an ideal 3D simulation, these aggressive gains command massive spikes in signals to the inertia wheel. The breakthrough to achieving simulation stability is the implementation of a "soft suspension" tuning mindset. By drastically reducing the gain values to fractional levels of the original ones, the controller is able to correct the virtual frame without breaking the physics engine. With these low-magnitude values, the virtual twin successfully achieves a perfect equilibrium when starting the simulation from a 10° offset, as shown in Figure 9.

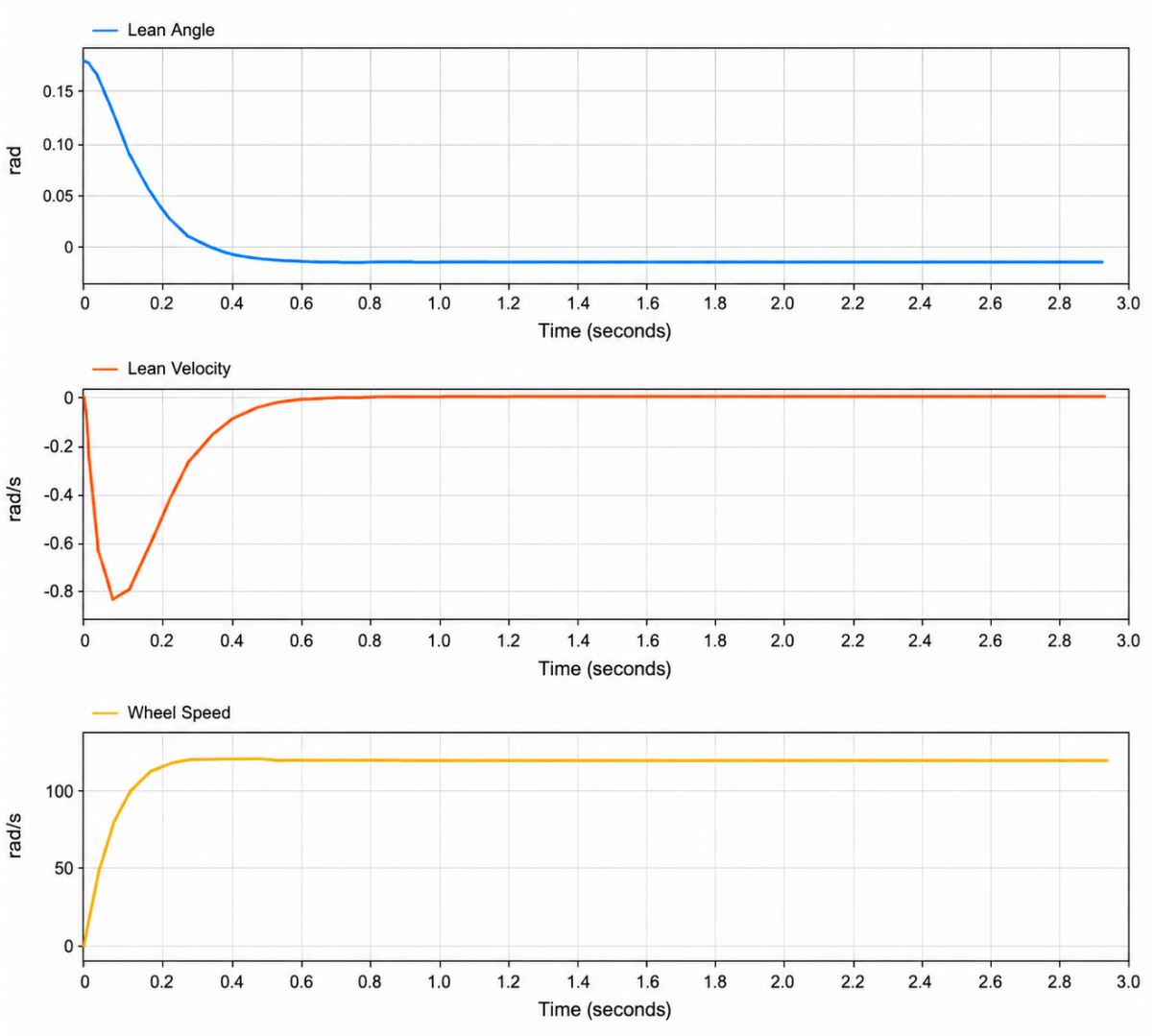

Figure 9. Simulation results of the Virtual Prototype

4.2 Real-Time Testing: Physical Performance

The control logic is automatically compiled and deployed to the Arduino board. Physical testing introduces the complexity of real-time implementation, as the idealized simulation gains from the digital twin are entirely insufficient to beat actual gravitational forces. Stabilization requires scaling the proportional gain to 63, as shown in Figure 7.

4.2.1 Balancing Performance

The dynamic performance of the PD controller in a real-time implementation is shown in Figure 10. The telemetry data in question shows a period of active stabilization in which the system maintains an upright posture for nearly 3 minutes on its own.

As seen in the top graph, where "*pTermH*" means the Lean Angle, the controller maintains the chassis near the equilibrium point ($\theta = 0$). In the middle graph, where "*dTerm*" means the Lean Velocity, the controller prevents sensor noise from distorting the motor command. The bottom graph, where "*pTermW*" means Inertia Wheel Speed, shows the active work of the inertia wheel.

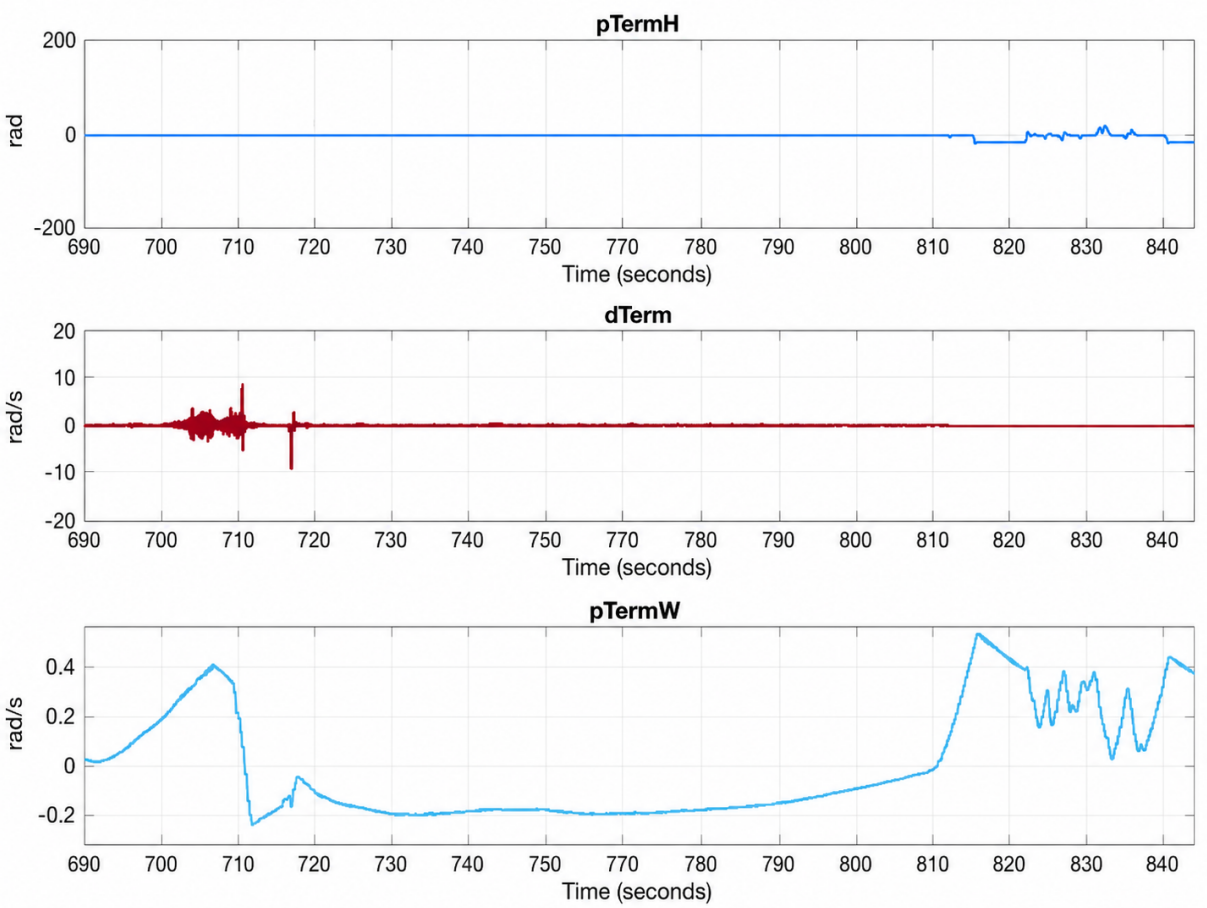

Figure 10. Telemetry data from the physical self-balancing motorcycle

4.2.2 Recurring Calibration and Equilibrium Offset

During real-time testing, the need for recurring calibration is identified as important. To maximize the IMU's likelihood of being in a correct calibration state, a calibration process in which the motorcycle is tilted in every direction is performed between each test run.

While the IMU successfully maintains its internal calibration value, the physical balance point is not aligned with the sensor's vertical axis. Factors such as component wiring,

asymmetric structures, and the pull of the power/data cable cause the true balance point to be offset by almost 4 degrees, which is corrected by using a bias block in the sensor preprocessing subsystem.

4.2.3 Mechanical Slippage and Error Analysis

The most significant problem identified while testing the physical motorcycle is the lack of friction between the inertia wheel and the DC motor axle. During high acceleration and a change in spin direction, the inertia wheel spins independently of the axle. Because the motor encoder tracks the axle's rotational speed rather than the inertia wheel, the slip compromises the integrity of the controller. When the slip occurs, and the IMU detects angular acceleration, the controller applies more power to the motor, whereas in the real-time, it doesn't generate any gyroscopic force.

The slippage results in an increase in error in the inertia wheel speed, which inevitably leads to a system collapse. However, putting pieces of tape around the mounting hardware of the inertia wheel reduces the clearances and finally restores the immediate momentum transfer. This allows the controller to successfully push the lean angle back toward zero and maintain stability.

5 Discussion

5.1 Comparison of Modeling Approaches

This project compares theoretical mathematical modeling, virtual CAD-based simulation, and physical hardware implementation. The Lagrangian equations provide an optimal theoretical starting point but assume a perfectly rigid body operating in a frictionless environment.

The Simscape Multibody model proves to be valuable in closing the gap between ideal mathematics and reality. By extracting properties from 3D geometry, the MBD workflow avoids the theoretical uncertainty inherent in manual parameter estimation. Simulating the system before actual implementation prevents hardware damage by revealing coordinate polarity mismatches in the virtual environment, which would otherwise be transferred to the physical model.

However, MBD also highlights a big difference in control tuning. The “soft suspension” PD gains that balance the virtual simulation successfully are entirely insufficient for the real-time model, which requires raw parameter multiplication to overcome physical constraints and gravity. The digital twin model can’t account for the mass of the motorcycle's external wiring, and axle slippage is a serious issue. Therefore, while the virtual simulation reduces the time required for the actual development process, iterative physical testing remains an absolute necessity.

5.2 Reliability

The reliability of the self-balancing motorcycle is heavily affected by the physical limits of its hardware and the vulnerability of its mechanical assembly. After the center of balance is corrected, the system appears highly reliable when operating near its vertical equilibrium.

However, what remains is a tendency toward motor saturation. When the inertia wheel spins too fast, and the motor changes direction, the axle slips from the inertia wheel. Saturation makes the inertia wheel spin too fast, and slippage makes it spin too slowly. In both situations, the inertia wheel loses its intended purpose.

5.3 Future Work

While the current PD control design seems to maintain balance at steady state, the reliance on the method in question can leave the system vulnerable to steady-state errors. Future development on this subject could focus on upgrading the control logic to a full Proportional-Integral-Derivative (PID) or just a Proportional-Integral (PI) controller. Introducing an integral term K_i would allow the system to automatically calculate constant offset disturbances by actuating the past error over time [17].

Additionally, redesigning the coupling between the inertia wheel and the motor shaft to use a keyed joint would eliminate slipping, improve the system's reaction time, and eliminate the need to overtighten the shaft cap. Ultimately, turning the microcontroller into a fully wireless system would eliminate another mechanical disturbance and enable further development, with the motorcycle intended to travel.

6 Conclusion

This thesis has presented the design, analysis, and implementation of a self-balancing motorcycle using an inertia wheel. The system is modeled as an underactuated inverted pendulum, where stabilization is achieved through reaction torque generated by controlled angular acceleration of the inertia wheel, counteracting gravitational disturbances and maintaining an upright posture.

The development is carried out using an MBD approach, combining CAD-based system modeling, simulation in Simscape Multibody, and automatic code generation for embedded implementation. This methodology enabled efficient system development by allowing early validation in a virtual environment and reducing implementation errors during hardware deployment.

The results address the research questions formulated in Chapter 1. The study demonstrates that a stationary motorcycle can be stabilized using a PD-based feedback controller with a low-pass filter. It also shows that the MBD workflow is an effective development approach for embedded control systems, particularly in improving design iteration speed and system verification. However, significant differences were observed between the simulation and physical implementation. While the virtual model achieved stable balancing under relatively low tuning effort, the real system required more aggressive controller gains due to sensor noise, mechanical imperfections, calibration drift, and unmodeled dynamics. These findings highlight that although MBD provides a powerful framework for system development, real-time performance is ultimately constrained by physical effects that are not fully captured in simulation models. This emphasises the importance of validating control systems beyond simulation when dealing with inherently unstable mechanical systems.

Future work may focus on extending the controller to a full PID structure to improve disturbance rejection and steady-state performance. Additionally, improvements in sensing, calibration, and mechanical design could further reduce the discrepancy between simulation and real-time behavior.

References

[1] I. Siradjuddin, E. R. K. Pradani, E. Rohadi, S. Adhisuwignjo, M. Kusumawardani, ja I. M. Fitriani, ”Designing, implementing and analysing optimal controllers on a non-linear reaction wheel pendulum”, *J. Phys. Conf. Ser.*, vol. 1402, nro 4, s. 044025, joulu 2019, doi: 10.1088/1742-6596/1402/4/044025.

[2] I. M. Mehedi, U. Ansari, A. H. Bajodah, U. M. AL-Saggaf, B. Kada, ja M. J. Rawa, ”Underactuated rotary inverted pendulum control using robust generalized dynamic inversion”, *J. Vib. Control*, vol. 26, nro 23–24, s. 2210–2220, 2020, doi: 10.1177/1077546320916022.

[3] C. Huygens ja H. Oscillatorium, ”The pendulum clock”, *Trans RJ Blackwell Iowa State Univ. Press Ames*, 1986.

[4] B. Garabedian, M. Benoit, ja S. Krut, ”A Futuristic Monorail Tramway Stabilized by an Inertia Wheel”, heinä 2007, s. 1581–1586. doi: 10.1109/ICCA.2007.4376626.

[5] L. D. Taylor, ”The monorail ‘revolution’ of the 1950s and 1960s and its legacy”, *J. Transp. Hist.*, vol. 37, nro 2, s. 236–257, 2016, doi: 10.1177/0022526616667955.

[6] K. H. Quang ja B. P. Cong, ”Study on Inertia Wheel Pendulum Applied to Self-Balancing Electric Motorcycle”, teoksessa *Proceedings of 2018 4th International Conference on Green Technology and Sustainable Development (gtsd)*, New York: IEEE, 2018, s. 687–692.

[7] Arduino Education, ”CONTENTPREVIEW AEKR2”. [Verkossa]. Saatavissa: https://edu-content-preview.arduino.cc/content-preview/university/project/CONTENTPREVIEW+AEKR2

[8] H. Gritli ja S. Belghith, ”Robust feedback control of the underactuated Inertia Wheel Inverted Pendulum under parametric uncertainties and subject to external disturbances: LMI formulation”, *J. Frankl. Inst.*, vol. 355, nro 18, s. 9150–9191, 2018, doi: https://doi.org/10.1016/j.jfranklin.2017.01.035.

[9] M. Olivares ja P. Albertos, ”Linear control of the flywheel inverted pendulum”, *ISA Trans.*, vol. 53, nro 5, s. 1396–1403, 2014, doi: https://doi.org/10.1016/j.isatra.2013.12.030.

[10] K. Kanjanawanishkul, ”LQR and MPC controller design and comparison for a stationary self-balancing bicycle robot with a reaction wheel”, *Kybernetika*, s. 173–191, maalis 2015, doi: 10.14736/kyb-2015-1-0173.

[11] S. Vadlamudi, K. V. Lakshmi, A. Yaramala, C. Uppalapati, ja P. K. Kolluri, ”Self Balancing Motorcycle Using Reinforcement Learning”, teoksessa *2024 International Conference on Emerging Systems and Intelligent Computing (ESIC)*, 2024, s. 691–696. doi: 10.1109/ESIC60604.2024.10481556.

[12] D. Zaborniak, K. Patan, ja M. Witczak, ”Design, Implementation, and Control of a Wheel-Based Inverted Pendulum”, *Electronics*, vol. 13, nro 3, 2024, doi: 10.3390/electronics13030514.

[13] A. Abdelgawad, T. Shohdy, ja A. Nada, ”Model- and Data-Based Control of Self-Balancing Robots: Practical Educational Approach with LabVIEW and Arduino”, *IFAC-Pap.*, vol. 58, nro 9, s. 217–222, tammi 2024, doi: 10.1016/j.ifacol.2024.07.399.

[14] I. Balan, A.-I. Timofte, S.-N. Plăcintă, V. Horga, ja A. Sălceanu, ”Using Model-Based Design in the Digital Control of Electrical Drives Laboratory”, teoksessa *2025 International Aegean Conference on Electrical Machines and Power Electronics (ACEMP) & 2025 International Conference on Optimization of Electrical and Electronic Equipment (OPTIM)*, 2025, s. 1–9. doi: 10.1109/OPTIM-ACEMP62776.2025.11075289.

[15] K. Kshirsagar, P. Shah, ja R. Sekhar, ”Model Based Design in Industrial Automation”, teoksessa *2022 6th International Conference On Computing, Communication, Control And Automation (ICCUBEA*, 2022, s. 1–6. doi: 10.1109/ICCUBEA54992.2022.10010895.

[16] T.-J. Shen ja C.-L. Chen, ”Model-in-the-Loop Design and Flight Test Validation of Flight Control Laws for a Small Fixed-Wing UAV”, *Drones*, vol. 9, nro 9, 2025, doi: 10.3390/drones9090624.

[17] J. Sun *ym.*, ”Proportional–Integral Controller Modified Landweber Iterative Method for Image Reconstruction in Electrical Capacitance Tomography”, *IEEE Sens. J.*, vol. 19, nro 19, s. 8790–8802, 2019, doi: 10.1109/JSEN.2019.2919923.