BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF MECHANICAL ENGINEERING

FAKULTA STROJNÍHO INŽENÝRSTVÍ

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

DEVELOPMENT OF AN UNSTABLE BALANCING VEHICLE USING MODEL BASED DESIGN

VÝVOJ NESTABILNÍHO BALANCUJÍCÍHO VOZIDLA S VYUŽITÍM NÁSTROJE MODEL BASED DESIGN

MASTER'S THESIS

DIPLOMOVÁ PRÁCE

AUTHOR AUTOR PRÁCE	Bc. Ondřej Mikulenka
SUPERVISOR VEDOUCÍ PRÁCE	doc. Ing. Robert Grepl, Ph.D.

BRNO 2025

Assignment Master's Thesis

Institut:	Institute of Solid Mechanics, Mechatronics and Biomechanics
Student:	**Bc. Ondřej Mikulenka**
Degree programm:	Mechatronics
Branch:	no specialisation
Supervisor:	**doc. Ing. Robert Grepl, Ph.D.**
Academic year:	2024/25

As provided for by the Act No. 111/98 Coll. on higher education institutions and the BUT Study and Examination Regulations, the director of the Institute hereby assigns the following topic of Master's Thesis:

Development of an unstable balancing vehicle using Model Based Design

Brief Description:

The design and implementation of a control system for a balancing unstable vehicle is a rather challenging technical problem, especially with regard to safety and robustness requirements. The standard approach is to use the tools, methods and approaches within Model Based Design. These techniques include modelling, parameter identification, controller design, automatic generation of C code for the controller and component and system level testing.
This thesis will discuss the design of a safe control system for a vehicle available in the Mechatronics Laboratory. The SimScape tool is expected to be used for the design of the electromechanical part model and the dSPACE system for HIL testing.

Master's Thesis goals:

1. Explore the current state of the vehicle, its electronics, and study previous theses dealing with this vehicle.
2. Define the requirements for the resulting control system, including safety and robustness requirements.
3. Create a simulation model of the vehicle and estimate its parameters.
4. Design and validate a control algorithm by simulation.
5. Based on the selected algorithm and requirements, design the architecture of the entire control system (sensors, electronics).
6. Design and manufacture a new control unit.
7. Test the control unit using HIL simulation.
8. Test the resulting control system on a real vehicle.

Recommended bibliography:

[1] Valášek, M.: Mechatronika, Vydavatelství ČVUT 1995

[2] https://www.mathworks.com/products/simscape.html

[3] Noskievič: Modelování a identifikace systémů

[4] Skalický, J. Teorie řízení 1. 1.vyd. Brno, 2002, ISBN 80-214-2112-6

[5] Zouhar, F.: Návrh konstrukce, řízení a elektroniky pro nestabilní balancující vozidlo, mechlab, DP 2011

[6] Štěpánek, J.: Identifikace systému, sensorika a implementace řídicího algoritmu pro nestabilní balancující vozidlo, mechlab, DP 2011

Deadline for submission Master's Thesis is given by the Schedule of the Academic year 2024/25

In Brno,

L. S.

prof. Ing. Jindřich Petruška, CSc.
Director of the Institute

doc. Ing. Jiří Hlinka, Ph.D.
FME dean

Abstrakt

Tato diplomová práce se zabývá návrhem a implementací řídicího systému pro nestabilní dvoukolové vozidlo pomocí metodologie Model Based Design (MBD). Hlavním cílem bylo vytvořit simulační model vozidla, navrhnout stabilizační algoritmus, navrhnout novou řídicí jednotku a otestovat celý systém nejprve pomocí simulací, poté v režimu Hardware-in-the-Loop (HIL) a nakonec i na reálném vozidle. Práce porovnává různé přístupy k modelování systému (white-box, grey-box a black-box) a testuje několik regulačních strategií (PID, SMC, LQR a MPC). Na základě výsledků simulací a experimentů byla pro reálné nasazení vybrána robustní metoda řízení typu Sliding Mode Control. Důraz je kladen na bezpečnost, modularitu systému a opakovatelnost testování. Výsledkem je funkční prototyp řízeného balancujícího vozidla, který slouží jako demonstrátor pokročilých mechatronických technik a metod návrhu řízení.

Summary

This master's thesis focuses on the development and implementation of a control system for an inherently unstable two-wheeled vehicle using the Model Based Design (MBD) approach. The objective was to build a simulation model of the vehicle, design and validate stabilization algorithms, develop a new control unit, and verify the entire system via simulation, Hardware-in-the-Loop (HIL) testing, and real-world deployment. The thesis explores various modeling techniques—white-box, grey-box, and black-box—and evaluates multiple control strategies including PID, Sliding Mode Control (SMC), Linear Quadratic Regulator (LQR), and Model Predictive Control (MPC). Based on the results, SMC was selected for implementation due to its robustness and favorable rider response. Emphasis is placed on safety, system modularity, and reproducibility. The outcome is a functional prototype that demonstrates advanced mechatronic design and control methodologies in practice.

Klíčová slova

Model Based Design, nestabilní balancující vozidlo, Simulink, řízení v reálném čase, Sliding Mode Control, Hardware-in-the-Loop, mechatronika

Keywords

Model Based Design, unstable balancing vehicle, Simulink, real-time control, Sliding Mode Control, Hardware-in-the-Loop, mechatronics

Bibliographic citation

MIKULENKA, O. *Development of an unstable balancing vehicle using Model Based Design.* Brno: Brno University of Technology, Faculty of Mechanical Engineering, 2025. 74 pages, Master's thesis supervisor: Robert Grepl.

I affirm that the presented master's thesis is my genuine work and that it was created with the support of the stated literature, under the supervision of my tutor.

Ondřej Mikulenka

Brno . . . .

I would like to express my sincere gratitude to my supervisor, doc. Ing. Robert Grepl, Ph.D., for his valuable guidance, support, and expertise throughout the development of this thesis. I would also like to thank all members of the Mechatronics Laboratory staff for fostering a collaborative and supportive environment that greatly contributed to the progress of this work.

Ondřej Mikulenka

Contents

1 Introduction

This work embraces the philosophy of **Model-Based Design (MBD)**, aiming to demonstrate its applicability to a real-world engineering problem. By following the **V-diagram development process**, we ensure that every stage—from system specification and modeling to implementation and testing—is traceable and rigorously verified. Ultimately, this thesis should serve as a case study applied to an unstable balancing vehicle showcasing the effectiveness of MBD and V-model methodologies when applied to complex, safety-critical mechatronic systems. If someone wants to improve an unstable vehicle, they will have a clearly defined structure and environment for testing and validation.

The unstable vehicle on which we will demonstrate the model-based design methodology is a two-wheeled unstable Segway-type vehicle. In the Mechatronics Laboratory we have such a vehicle, which was created as part of the students' final theses and has already undergone many modifications. This vehicle represents a highly interesting technical system due to its complex dynamic characteristics—it is underactuated, non-linear, and inherently unstable. These properties make it an ideal candidate for testing advanced control strategies, but at the same time they also demand careful system modeling, simulation, and validation to ensure both safety and performance.

To successfully develop a control system for such a complex vehicle, it is crucial to consider not only the technical feasibility but also the general and safety-related requirements. These requirements influence every aspect of the design and development process—from hardware selection to software implementation. Safety becomes a key driver in the entire project, as real-world testing of unstable systems can pose significant risks to both equipment and personnel if not handled properly.

To achieve a functional and reliable system, the selection of appropriate sensors and actuators becomes one of the initial and most important tasks. These components must be carefully chosen to match the system dynamics and measurement needs, while also providing robustness and fault tolerance. The integration of hardware components directly influences the accuracy of control and the quality of system feedback, which is critical for maintaining stability in real time.

In order to safely test various control algorithms, it is essential to first develop a suitable mathematical model of the vehicle. This model will serve as the foundation for simulations, controller design, and eventual code generation. A well-validated model enables the safe virtual testing of different scenarios and controller configurations before deploying them on the real system, significantly reducing the risk of hardware damage or failure.

Another key aspect of the development process involves selecting an appropriate control unit that can meet the computational and interfacing demands of the system. Since code generated from model-based environments such as MATLAB/Simulink may be used, the control unit must support automatic code deployment, real-time execution, and flexibility in communication protocols. This step ensures that the transition from simulation to real-world application is seamless and efficient.

As the system evolves, functional testing plays a crucial role in verifying new control algorithms and their impact on the system's behavior. Hardware-in-the-loop (HIL) testing emerges as an effective method to validate controller behavior under near-realistic conditions, without endangering the actual hardware. By integrating the controller with a simulated model of the vehicle, HIL allows us to rigorously test fault conditions, edge cases, and safety protocols in a controlled environment.

2 Research - State of the art

2.1 Review of the current literature

Štěpánek Jan - SYSTEM IDENTIFICATION, SENSORY SYSTEM AND IMPLEMENTATION OF CONTROL ALGORITHM FOR UNSTABLE BALANCING VEHICLE

The master's thesis by Štěpánek focuses on the design and construction of a Segway-like vehicle (Measuring data for parameter estimation Fig. 2.1) intended for personal transport, emphasizing the implementation of control algorithms for stability. The research involved an extensive review of existing projects and sensor technologies, leading to the development of a control algorithm utilizing a dsPIC microcontroller, along with safety measures to prevent accidents.

The project culminated in successful simulations and practical tests, demonstrating the vehicle's effective stability and control mechanisms, showcasing the potential for innovative personal transport solutions. [1]

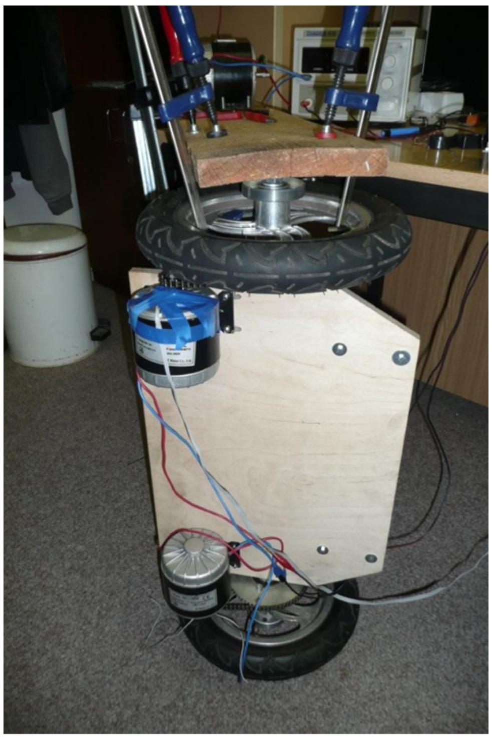


Figure 2.1: Measurement of sample data for the simulation of parameters of the test platform base of the older version of the vehicle from the thesis of Jan Štěpánek

František Zouhar -DESIGN OF CONSTRUCTION, CONTROL AND ELECTRONICS FOR UNSTABLE BALANCING VEHICLE

The thesis by František Zouhar focuses on the design and development of an unstable balancing vehicle, detailing the construction, control systems, and electronics involved.

It includes a comprehensive analysis of requirements, the creation of 3D models (Fig. 2.2), and simulation models using Lagrange equations, along with the design of PID and LQR regulators to optimize vehicle performance. The work culminates in the successful creation of an autonomous vehicle capable of transporting a person, with user-friendly control and verified performance through independent testing. [2]

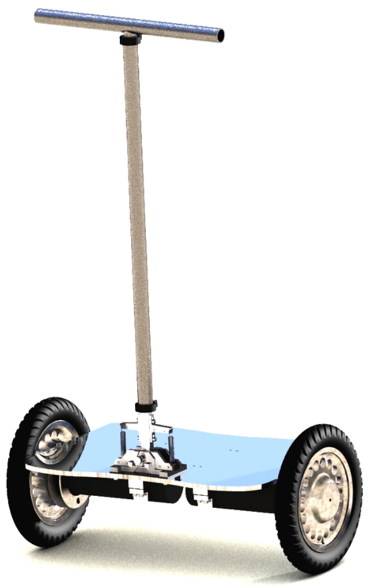


Figure 2.2: A complete 3D model of the carriage's final construction from the thesis of František Zouhar. From [2]

Petr Horák - CONTROL OF LABORATORY MODEL OF UNSTABLE BALANCING VEHICLE

This thesis is part of the HUMMER project, focusing on the development of a two-wheeled unstable vehicle similar to a Segway and its smaller laboratory model. The work includes a comprehensive review of existing models, the design and realization of necessary electronics, and the identification of system parameters, emphasizing the control mechanisms and sensor integration for effective operation. The project aims to facilitate the testing and design of control algorithms through the laboratory model, providing insights into the principles of operation and control strategies such as PID and LQR. [3]

Barnabas Dobossy - SEGWAY DRIVER PARAMETER ESTIMATION AND ITS USE FOR OPTIMIZING THE CONTROL ALGORITHM

The master's thesis by Barnabás Dobossy focuses on the development and implementation of an adaptive controller for a two-wheeled self-balancing vehicle, addressing the challenges posed by varying driver parameters such as body mass and height. The research explores various techniques for parameter estimation and controller synthesis, including the use of Lagrange multipliers and sequential quadratic programming, while emphasizing the importance of adapting the controller to maintain stability and user safety.

The thesis also details the software-in-the-loop (SIL) testing of different algorithms to identify the most effective combination for the adaptive controller. Additionally, it includes practical applications of the developed algorithms, such as improvements in the vehicle's stability and performance, and discusses future development suggestions to enhance the system further. [4]

Michal Matejasko - DESIGN OF A FAULT-TOLERANT CONTROL SYSTEM

FOR A SELF-BALANCING TWOWHEEL VEHICLE

The thesis discusses the development of a safe control system for unconventional personal transport vehicles, specifically the Segway, which is a complex mechatronic system comprising various components such as motors, sensors, and control electronics. It emphasizes the importance of passenger safety and reliability, highlighting the need for a robust design that addresses potential failures through methods like Failure Mode and Effects Analysis (FMEA). The proposed solution involves a control system with two redundant electronic control units (ECUs) and a voter to ensure safe operation, capable of responding to sensor malfunctions and ECU failures while maintaining system simplicity. [5]

Ondřej Richter - DESIGN AND IMPLEMENTATION OF AN INNOVATIVE CONTROL SYSTEM FOR AN UNSTABLE BALANCING VEHICLE

This thesis focuses on innovating the electronics system of a Segway-like vehicle, emphasizing improvements in reliability, safety, and wireless communication. It begins with an evaluation of past works and the current state of the system, identifying areas for enhancement. The research includes the assessment and replacement of outdated electronics, the realization of software for wireless data transfer between the main board, communication board, and PC, and the development of a graphical user interface (GUI) for visualizing sensor data.

The project aims to create a more effective platform for demonstrating mechatronics principles through a functional balancing vehicle, ultimately contributing to advancements in power and control electronics, as well as wireless communication technologies. [6]

Vojtěch Solnický - AUXILIARY SENSORICS FOR UNSTABLE BALANCING VEHICLE

This thesis focuses on enhancing an unstable balancing vehicle by integrating a display device to monitor (Fig. 2.3) driving conditions, which are gathered from various sensors. Key improvements include the implementation of a built-in presence sensor for the driver, data logging capabilities on an SD card, and addressing issues related to electronic damage caused by induced voltage in DC motors. The project aims to improve the vehicle's safety and functionality through the addition of a Multimedia Expansion Board, strain gauge, electromagnetic relay, encoder, and a battery system. [7]

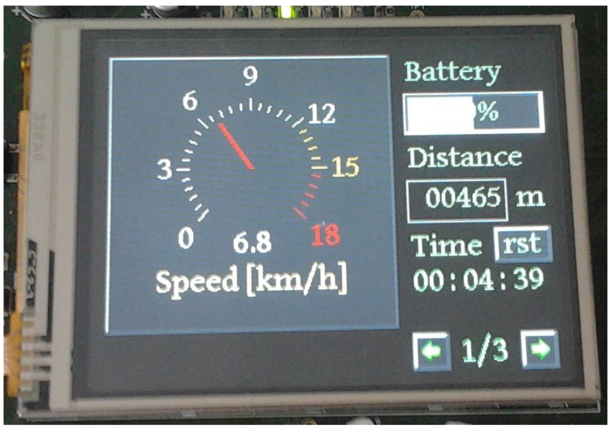


Figure 2.3: Screen display of driving characteristics from Vojtěch Solnicky's thesis [7]

Michal Bastl - DESIGN OF CONTROL UNIT FOR TWO-WHEELED SELF-BALANCING VEHICLE

The project focuses on designing safer electronics for the unstable balancing vehicle HUMMER, emphasizing advanced diagnostics and fault detection. It includes an analysis of the original vehicle using FMEA, leading to a new hardware concept that encompasses power electronics, control units, and supplies, resulting in prototypes for testing the new design.

Key components of the project involve the development of various circuit schematics and PCB designs, including CPLD and dsPIC modules, as well as power supply systems utilizing LM2576HVT regulators. The overall goal is to improve system reliability and performance through innovative electronic solutions tailored for the vehicle's unique challenges. [8]

2.2 Model Based Design

Computer modelling and simulation are increasingly used in the modern design of mechatronic products. We assume that the cost of building a real prototype would be significantly higher than the cost of developing an adequate computer simulation (virtual prototype). The motivation is of course to reduce development costs and shorten innovation cycles. Therefore, we choose a strategy for the entire product design process so that maximum design, verification and implementation activities are performed on the computer. This whole area is developing very dynamically, including the technical terminology and the precise definition of the individual design steps.

The design methodology or philosophy called **Model Based Design** (MBD) can be implemented as follows

- The key is to use the model (set of models) throughout the development process. The same model is used by the whole development team.
- The requirements for the resulting system behavior are defined in the model (Executable specifications).
- The model is refined and modified during development, e.g. based on experimental verification of the properties of (some) components.
- Model is used at a certain stage of development to generate code for an embedded device (control microcontroller).
- Testing takes place continuously throughout the development.

An important part of MBD is the "seamless" transition between computer simulation and reality. This includes Rapid Control Prototyping (RCP), Hardware In the Loop (HIL), C code generation and more.

Rapid Control Prototyping

When designing a control system for an electromechanical or other physical process, the traditional workflow often involves selecting appropriate control hardware (such as a microcontroller or PLC) and manually implementing the control algorithm in a low-level programming language like C. This implementation is usually based on a control model created and validated in a high-level environment such as Simulink. However, transferring a functional Simulink model into executable code for specific hardware introduces several challenges—timing issues, fixed-point arithmetic, limited computational resources, and other hardware constraints must be carefully considered. This process can be time-consuming, error-prone, and requires deep knowledge of both the model and the target hardware.

Moreover, the control strategy is typically designed using a simplified mathematical model of the real system. Since this model rarely matches the physical system perfectly, the implemented control algorithm might not perform as expected once deployed, potentially requiring numerous iterations and re-implementations.

Rapid Control Prototyping (RCP) offers an alternative that significantly streamlines this process. Instead of manually rewriting the model into code, RCP enables direct execution of the control algorithm on a real-time capable platform (e.g., a PC with an I/O card or a dedicated prototyping unit), which interfaces directly with the physical system (Fig. 2.4). The control logic remains in the Simulink environment, and the entire system runs in real time, allowing for immediate testing and tuning of control strategies—without writing a single line of low-level code.

This approach accelerates development, facilitates experimentation, and allows evaluating controller performance on the real system at a much earlier stage. If needed, the Simulink model can later be automatically translated into optimized C code for deployment on the final embedded hardware, using tools such as Simulink Coder or Embedded Coder.

RCP is thus a key element of **model-based design** methodologies. It shortens development cycles, reduces the risk of implementation errors, and provides a highly flexible environment for iterative controller design and validation. [9, 10]

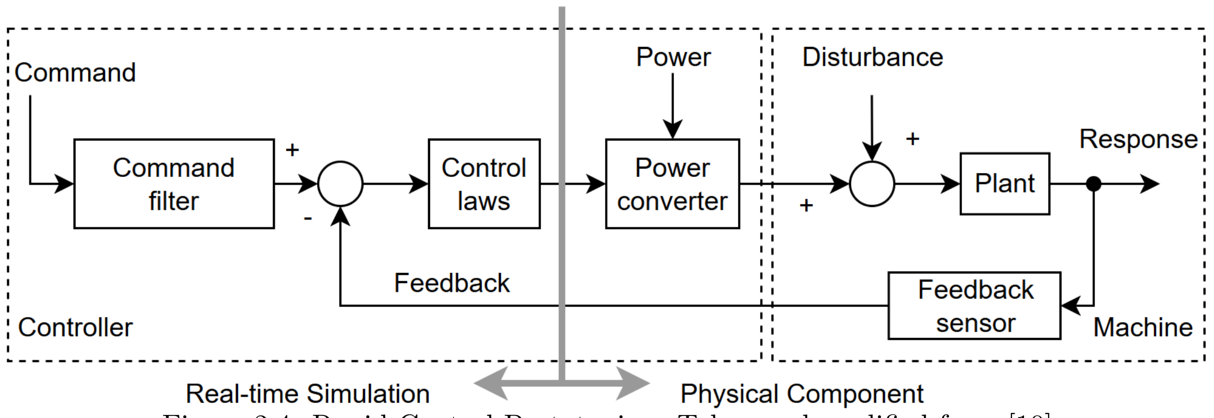


Figure 2.4: Rapid Control Prototyping: Taken and modified from [10]

Hardware In the Loop (HIL)

In traditional mechatronic system development, the typical process begins with designing and building the electromechanical part of the system, followed by implementing the control logic on the target hardware. Only then can the entire system be tested. While this method is straightforward, it presents several limitations: the real hardware must already be available for testing, some operational states may be dangerous or impossible to test, and test conditions are often difficult to reproduce consistently.

The Hardware-in-the-Loop (HIL) approach addresses these limitations by introducing real-time simulation into the testing process. In HIL testing, the real control hardware (e.g., a microcontroller or embedded computer) is connected to a simulated model of the physical system it is meant to control. This model runs on a dedicated computer in real time and interacts with the control hardware through appropriate input/output (I/O) interfaces.

HIL testing provides a safe and efficient environment for verifying embedded control algorithms under realistic conditions (Fig. 2.5). It enables testing of edge cases and failure scenarios, such as sensor malfunctions or unexpected disturbances, without risking damage to real hardware. Moreover, it allows for repeatable experiments and continuous integration of new software versions throughout the development cycle.

The main benefits of HIL include reduced development time and cost, improved test coverage, early validation of control logic, and increased safety. As such, HIL plays a critical role in modern **model-based design** workflows and serves as a key step in the transition from simulation to real-world deployment.[9, 10]

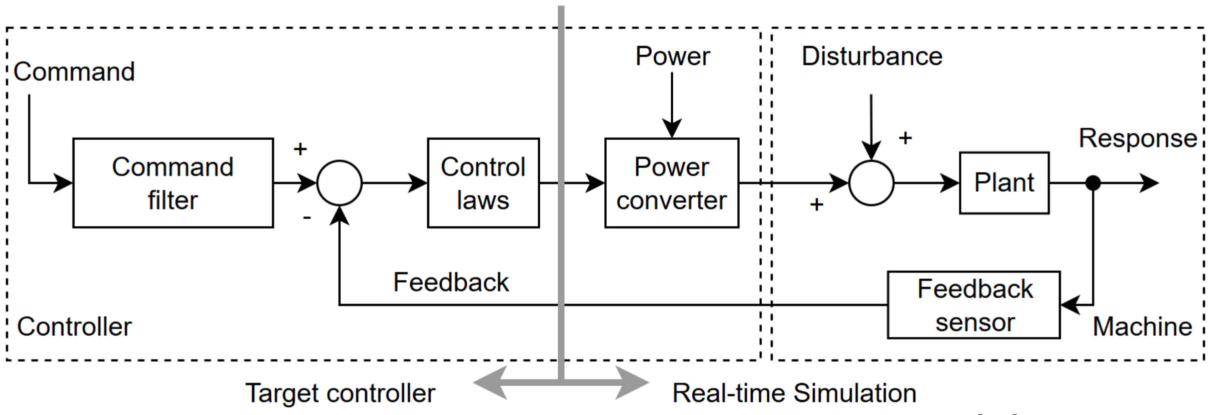


Figure 2.5: Hardware In the Loop: Taken and modified from [10]

Table 2.1 provides a summary of key differences between RCP and HIL, highlighting how each technique fits into different stages of the Model-Based Design workflow.

dSPACE

dSPACE is a proprietary hardware platform designed for real-time simulation and control applications. It combines a high-performance CPU with a wide range of I/O peripherals, making it a powerful tool for Hardware-in-the-Loop (**HIL**) testing, Rapid Control Prototyping (**RCP**), and other real-time control tasks.

The simplest configuration of dSPACE hardware is a single PCI or PCIe card equipped with a dedicated processor and numerous standard I/O channels, including analog inputs/outputs, digital I/O, encoder inputs, and communication interfaces such as CAN. More advanced setups are modular, consisting of a central processor board connected to specialized I/O modules. These can be housed either in a laboratory Expansion Box (Scalexio Fig. 2.7) or in a more rugged AutoBox, designed for in-vehicle testing under real-world conditions.

The development workflow is based on **Model-Based Design**. Control algorithms

Table 2.1: Comparison of Rapid Control Prototyping and Hardware-in-the-Loop

Feature	**Rapid Control Prototyping (RCP)**	**Hardware-in-the-Loop (HIL)**
Purpose	Validation of control algorithms on the real system	Testing of control hardware without requiring the physical system
Model usage	Control algorithm model runs on a powerful PC or prototyping platform	Model of the controlled system runs in real-time on a PC or RT system
Physical components	The physical plant is present	The physical plant is replaced by a simulation model
Controller hardware	Control is executed via a PC or general-purpose HW (e.g., dSPACE)	Control is executed by the target embedded system intended for deployment
Programming	No manual coding required (Simulink model runs directly)	Embedded system uses auto-generated or hand-written code
Main benefits	Fast validation, easy tuning, no need for programming	Safe testing, repeatability, failure and fault simulation
Development phase	Early stage – validating the algorithm on real hardware	Later stage – testing the embedded system before deployment

are developed in Simulink, then automatically converted into real-time executable code using Real-Time Workshop (RTW) and the Real-Time Interface (RTI). The model is uploaded and executed on the dSPACE hardware. Experiment control—such as real-time monitoring, signal visualization, and data logging—is handled via the ControlDesk environment, which provides an intuitive and powerful user interface.

Due to its reliability, precision, and flexibility, dSPACE has become a standard tool in the automotive industry and other high-demand sectors. However, this high-end solution also comes with a relatively high cost. [9]

Summary:

- Software: MATLAB, Simulink, RTW, RTI, ControlDesk
- Hardware: Modular or single-board system with real-time CPU and I/O
- Sampling rates: Up to $100kHz$ (depending on configuration)
- Ease of use: Very user-friendly; most interactions are done directly in Simulink and ControlDesk
- Application area: Industry-standard platform for real-time HIL/RCP testing

V-model

The development of mechatronic systems involves several steps, the essential ones of which Isserman summarized in the V-diagram for mechatronic systems (Fig.2.6). Progressing along the V-model results in an increasing degree of maturity of the mechatronic systems[13]. The V-model should be seen as a guide to the structure of the work, which systematically defines a given phase of development and helps to achieve a successful out-

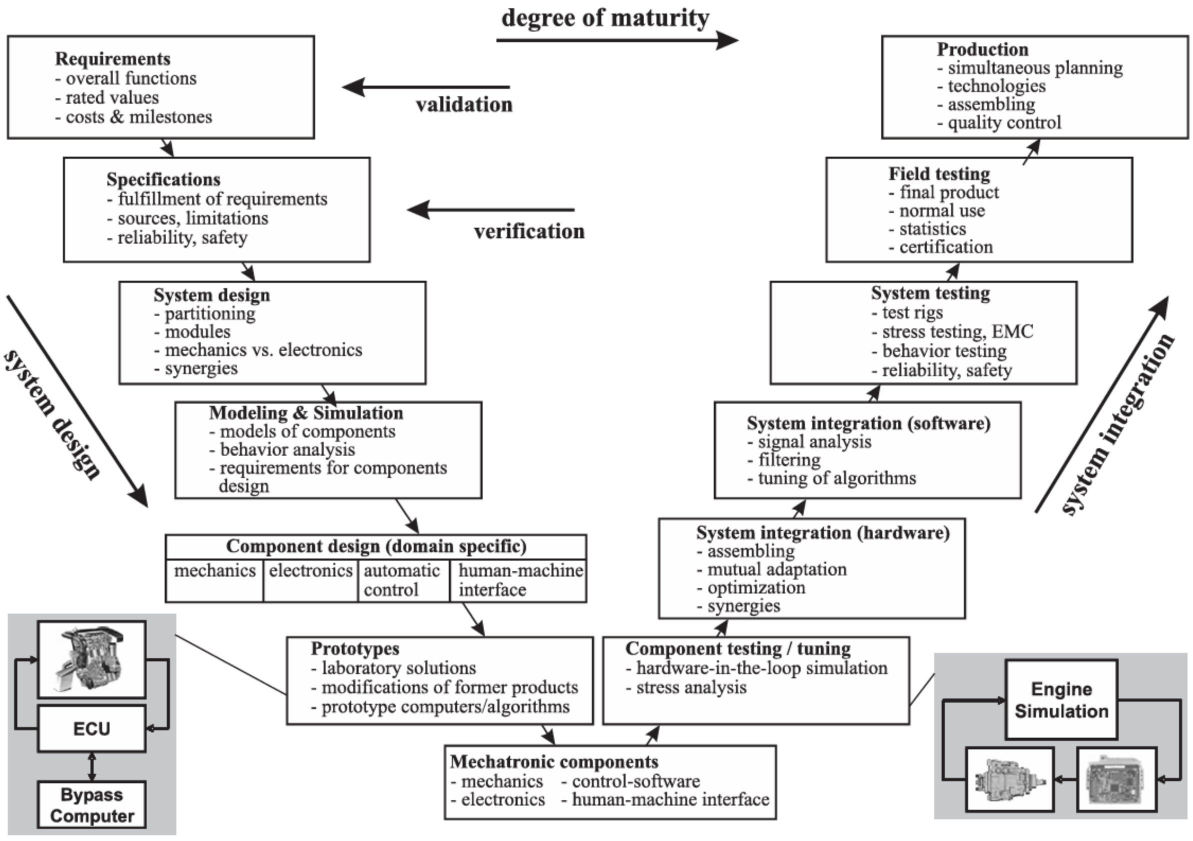


Figure 2.6: "V" development scheme for mechatronic systems. From [11]

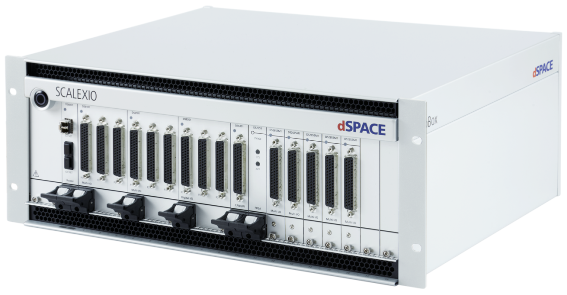


Figure 2.7: dSpace SCALEXIO LabBox Modular real-time system for control and test applications [12]

come. It is an iterative process and during the development we go back to previous steps and refine or change requirements that, for example, could not be met or needed to be developed.

Thus, in this thesis, we will attempt to walk through the entire development cycle within MBD using the V-diagram structure.

2.3 Current state of the vehicle

Current state of the mechanical parts of the vehicle

The mechanical design of the unstable balancing vehicle (Fig.2.8 and Fig.2.9) has remained almost unchanged since Zouhar's final thesis [2]. The user stands on an aluminum platform on which two DC motors are mounted on the underside using aluminum couplers. On the top, there is a handlebar mount with springs that help return the handlebars to their original position when no external force is applied. There are two user buttons on the handlebars, one of which is locking and the other not. There is also a master switch which serves as the main power supply to the entire system. The buttons are not soldered to anything and are connected to the system via WAGO pins on the underside, so their purpose can be slightly altered.

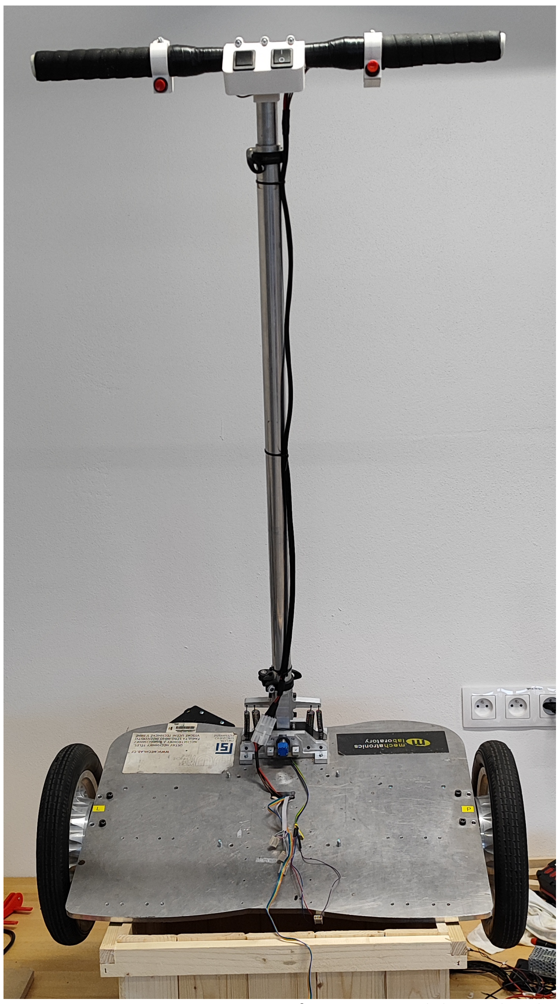


Figure 2.8: Top view of an unstable vehicle

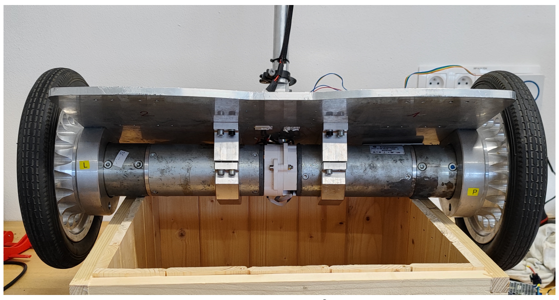


Figure 2.9: Bottom view of an unstable vehicle

The actuators on the unstable vehicle are two **Dunkermotoren GR80x80** (Fig. 2.10) DC motors with a planetary PLG60, which has a gear ratio of 9:1. The basic motor parameters are written in the table [2.2]. These motors have been on the unstable vehicle for several iterations and therefore it can be assumed that they are powerful enough to stabilize the vehicle with or without a rider.

Figure 2.10: Motor GR80x80 [14]

Nominal voltage	[V]	48
Nominal current	[A]	5.6
No-load current	[A]	0.35
Starting current	[A]	72
Nominal torque	[Nm]	0.67
Maximal torque	[Nm]	6
Nominal speed	[rpm]	3200
No load speed	[rpm]	3300
Torque constant	[NmA^{-1}]	0.132

Table 2.2: Motor parameters

The **MPU 6050** (Fig. 2.12) is a key sensor on this unstable vehicle as it contains an accelerometer and gyroscope which are used to calculate the vehicle's roll. Communication with this sensor is via I2C with a baud rate of up to 400 kHz. One of the features of the sensor is a programmable range, which in the case of the accelerometer is from $\pm 2g$ to $\pm 16g$ and in the case of the gyroscope is from $\pm 250°s^{-1}$ to $\pm 2000°s^{-1}$.

The **potentiometer** (Fig. 2.11) is mounted on the bottom of the handlebars and is used to determine the absolute rotation of the handlebars and therefore the rider's ability to let the system know that he wants to turn.

Figure 2.11: Potentiometer [15]

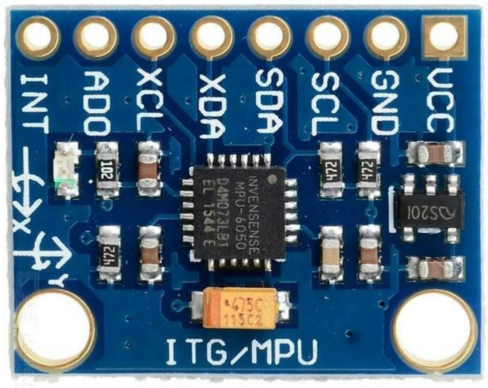


Figure 2.12: MPU 6050 [16]

The **ADuM1410** (Fig. 2.13) integrated circuit is used to separate the signals between the microcontroller and the motor drivers. The integrated circuit works on the principle of induction and is capable of high data rates of up to 20 Mbps.

DC-DC converter **MEAN WELL SD-15C-12** is used to reduce the supply voltage from batteries to the voltage level of 12V. The advantage of this converter is that it has separate grounds, thus protecting the signal part of the electronics from the power part.

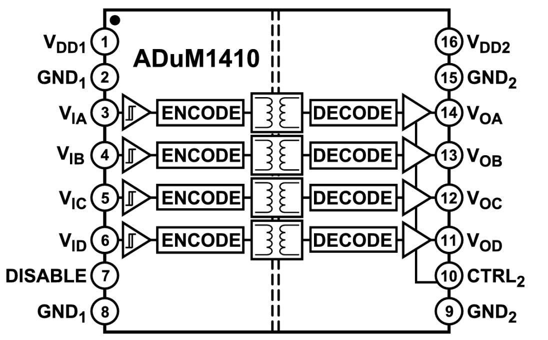


Figure 2.13: Quad-Channel Digital Izolator ADuM1410 [17]

Figure 2.14: DC-DC converter MEAN WELL SD-15C-12 [18]

For sensing the rotation of the shaft by the motors, an optical encoder **HEDM5500** (Fig. 2.15) with 1000 CPR is used here and since it is located on the rotor (before the gearbox), due to the gear ratio, one revolution of the wheel is reflected as 9000 CPR. So for our purposes this is too accurate an encoder, considering that even the backlash in the gearbox itself is much greater than the resolution itself. So replacing this encoder will not be necessary.

ESCON 50/5 motor driver (Fig. 2.16) was used for the motor control. It is a four-quadrant bridge with 250W output. The continuous load of the driver is 5A and the maximum current is 15A. According to the motor parameters, this is a weaker driver than would be appropriate, but for control purposes, it will only be determined from simulation whether the use of this driver is sufficient, or whether it will be necessary to replace this driver with a stronger one (probably a replacement for the ESCON 70/10, as this would be a simple replacement without any other necessary HW or SW modifications).

Figure 2.15: Optical incremental encoder HEDM5500 [19]

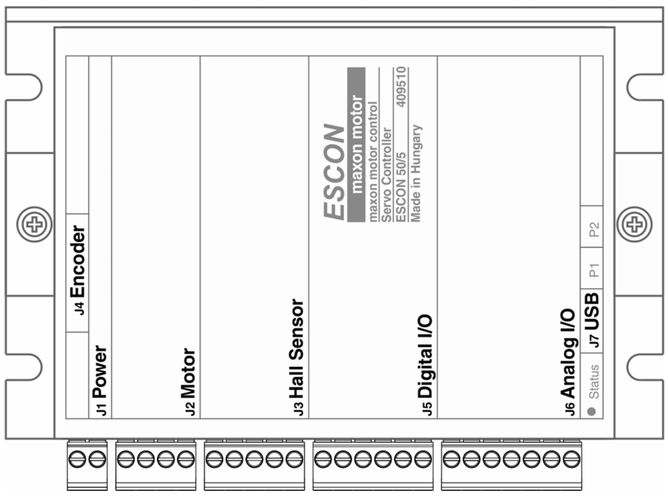


Figure 2.16: Motor driver ESCON 50/5 [20]

The vehicle is powered by three lead acid batteries. The advantage of these batteries is their safety and high current output. The disadvantage is that to charge these batteries,

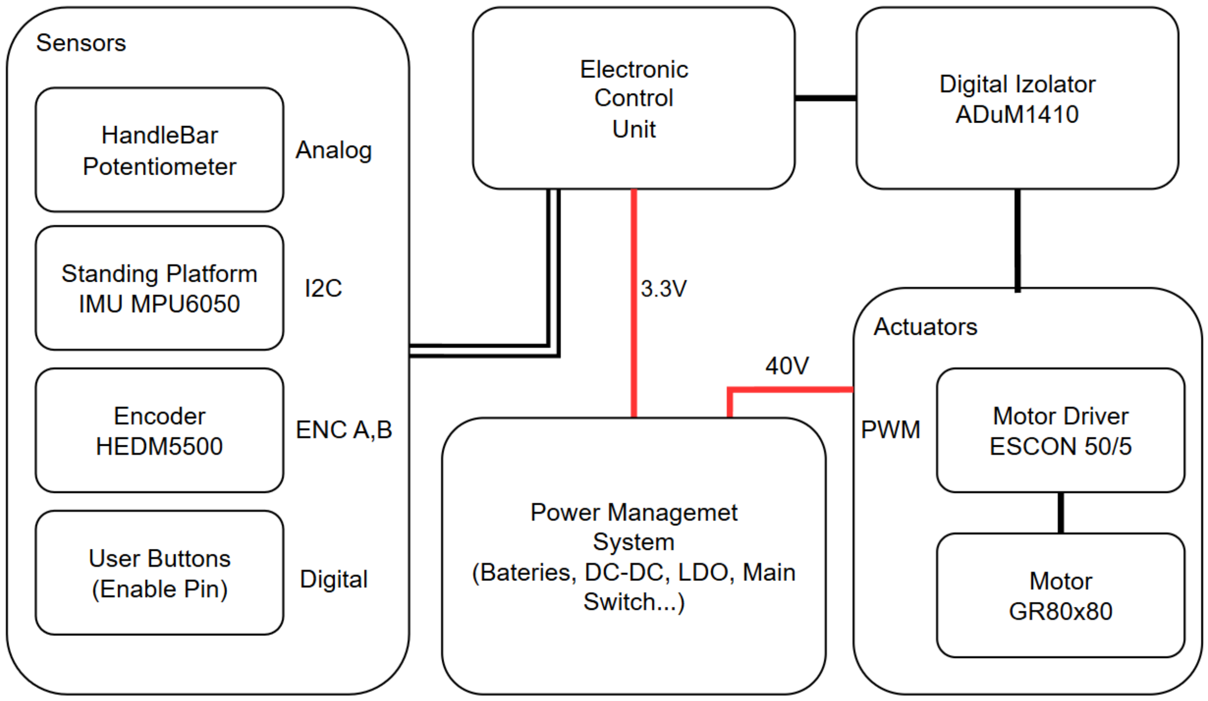


Figure 2.17: Wiring architecture of electronic components on an unstable vehicle

they need to be removed from the vehicle, which is both inconvenient and time-consuming. Therefore, as part of this work, consideration will be given to replacing these batteries.

The control unit that is currently on the vehicle is a learning platform from the mechatronics lab “Mechlab PicKit Board”, which is built on a 16-bit MicroChip dsPIC33FJ128MC804. Since the given control unit with the program was not part of any of the previous final works and its connection to the sensors and power electronics was done using dupont cables without any documentation, the current design cannot be related to the current one and it will be necessary to disassemble the electronic part of the vehicle and start from scratch with the new control unit as part of this work.

The wiring diagram in Fig.2.17 shows the connection of the individual components of the control system of an unstable balancing vehicle. The power management system provides power to the entire system, with the low voltage required for the control unit being generated separately, while the ESCON 50/5 motor driver is powered directly from the batteries. The control signal towards the motor driver is digitally isolated by the ADuM1410 circuitry, ensuring galvanic isolation and protecting the control unit from any faults or spikes on the power side. The schematic of Fig.2.18 shows the structure of the power system, including the transition from the battery supply through the individual inverters to the control electronics and the motor power amplifier.

There are four key sensors on the vehicle. A potentiometer is used to sense the rotation of the handlebars, whose analogue output is processed by the control unit. The main sensor for determining roll is the MPU6050, combining accelerometer and gyroscope, which communicates with the control unit via the I2C bus. Dual-channel HEDM5500 incremental encoders with a resolution of 1000 pulses per revolution (CPR) are installed on the motors to accurately measure the speed and direction of motion. The last element is the user buttons, which in the current configuration are not actively used during operation, but may be used in the future for activation or mode selection, for example.

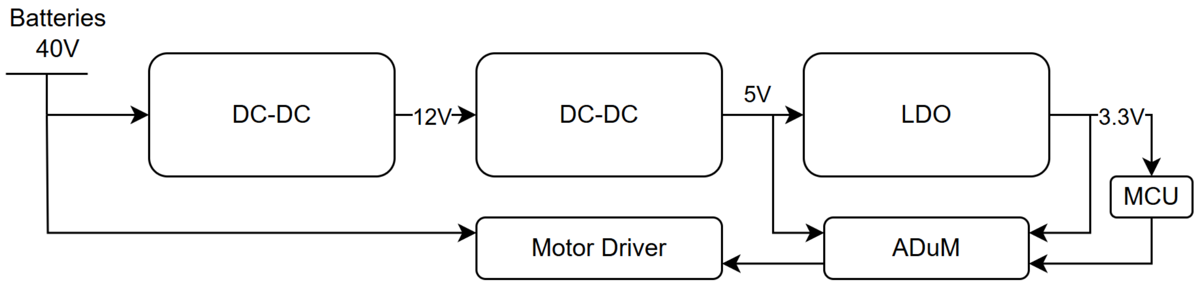


Figure 2.18: Electrical connection of power supply and galvanic isolation

2.4 Evaluation of the research study

The research study shows that the HUMMER project, or unstable balancing vehicle, has already been used several times in the mechatronics laboratory as a topic for the final thesis and has undergone many modifications. At the same time, however, after assessing the current state of the vehicle, it was found that modifications had been made to the vehicle outside of the final thesis, and since the changes were not documented anywhere, this resulted in a vehicle that is currently not functional. However, the mechanical part of the vehicle remained almost unchanged and so for the parameter estimation in this thesis we will partly rely on Dobossy's thesis [4]. At the same time, the requirements for the resulting control system and safety requirements have also remained unchanged over the years, and so we will base this on the thesis of Zouhar[2]. Since according to previous work the unstable vehicle was able to operate with and without a rider using the sensors and actuators that are currently on the vehicle, their replacement is not foreseen and the system architecture (electro-mechanical) is therefore likely to remain the same.

Furthermore, the basic idea of MBD and Isserman's V-model was described in the research part. This thesis will be oriented accordingly. Thus, the outcome of this work will be an attempt to incorporate development on an unstable vehicle within the MBD framework so that when further modifications are made, the development will have a clearly defined structure to help achieve a successful implementation. Thus, development will always start from specification, testing the model in an offline environment to testing the HIL on dSPACE, which if successful will minimise error in the final implementation.

3 Formulation of the thesis goal

As stated in the introduction, the primary motivation of this work is to enable the development and, in particular, the testing of a control unit for an unstable vehicle within the framework of Model-Based Design.

For the safety and reliability of the unstable vehicle, we need to define the requirements of the resulting control system. From the research part of the thesis, we know that security requirements have already been addressed by Zouhar in his thesis and hence we will use them in this thesis as well.

- Define control system requirements for driverless vehicle operation
- Define control system requirements for vehicle with a rider
- Define requirements to ensure basic safety conditions

We want to approach the vehicle as a big project. In order not to simply load the control algorithms onto the vehicle at the risk of harming people or objects in the vicinity, the control algorithms will need to be tested beforehand. For testing, we will need a model that plausibly describes the dynamics of the vehicle itself. We will therefore show some ways to obtain a model of the system and compare its behaviour with a real vehicle. Many previous theses have dealt with the creation of a mathematical model, but none of them dealt with the creation of a physical model in the Simscape multibody environment or the black-box method. Therefore, in this thesis we will also look at these modelling methods and compare their outputs against a real vehicle to determine their suitability for use within MBD.

- Outline the creation of a mathematical model using Lagrange's equations of the second kind
- Create a model of an unstable vehicle in the Simscape multibody environment
- Identify a system based on actual measured data of a real vehicle
- Compare different approaches to modelling an unstable vehicle in terms of applicability within MBD and in comparison to the actual dynamics of the vehicle itself.

The vehicle itself is unstable in its working position. Designing the controller directly on the vehicle is therefore a difficult task. Designing the control algorithm in a simulation environment will allow us to get at least an initial sketch of the controller parameters and an understanding of the system behaviour. For the use of advanced control algorithms, we will need to know all the states of the system and therefore we will also focus on the creation of a state observer. Previous work has primarily focused on the implementation of PID and LQR algorithms, but in this work we want to test other control algorithms and assess their suitability for implementation on a real vehicle.

- Design the PID, LQR, SMC and MPC control algorithms by simulation
- Evaluate their behavior on the simulation model and their suitability for implementation on a real vehicle
- Based on the mathematical model, we will determine which sensors (and how suitable they are) are needed to reconstruct all vehicle states.
- To be able to implement advanced control algorithms, we create a state observer (Kalman filter) and compare its outputs with the simulation model.
- We will create basic safety algorithms.

Based on the results from the simulation design of the control algorithm, we get the requirements for sensors and actuators. At this stage, we will evaluate the existing electromechanical part of the vehicle and make any necessary modifications. At the same time, we will focus on processing the data from the sensors themselves, which will be on the vehicle. Since the unstable vehicle has already been in a state of operation with the existing sensors (or a similar alternative), no drastic intervention in the electromechanical part is foreseen.

- Replacing lead-acid batteries with a more suitable alternative
- Evaluating the usability of the existing sensors on board the vehicle based on the control and safety algorithms used.
- Determine the accuracy of the MPU6050's location (rotation) on the vehicle to correctly recalculate the rotation angle and determine the maximum rotation range for use in safety algorithms.
- Testing different methods of calculating roll from the MPU6050 and comparing their results with each other and evaluating their applicability on a real vehicle.
- Solve the problem related to counter overflow on the control unit when reading encoder values.

The existing control unit currently on the vehicle was added outside of the final thesis and its wiring is not documented. We will therefore design a new control unit. Our requirements will be sufficient computational power, the ability to generate code from Simulink, and the ability to test function blocks from Simulink using Software In the Loop and Processor In the Loop. The choice of a new control unit was last addressed by Richter in his thesis, so we will use his work to evaluate whether the chosen control unit meets our requirements.

- Verification of peripherals and performance for correct functionality of components on the vehicle.
- Verify the ability to generate code from Simulink.
- Verify the ability to test function blocks in Simulink using the Processor In the Loop method

- Create an interconnect board to connect the new control unit to the vehicle. This will mainly be a connection to the existing connectors of the individual sensors.

To verify the functionality of the control unit and the algorithm generated on it, we will have to test the control unit beforehand. This will minimize errors that may have occurred before the unit is actually implemented on the vehicle. For testing, we will use dSPACE, which will run the mathematical model of the vehicle. The output from dSPACE will be the emulated outputs of the individual sensors that are on the vehicle. So in this part we will prepare the test environment and program the behavior of each sensor.

- Program individual function blocks for emulation of individual sensors
- Due to the absence of I2C peripherals on the dSPACE platform located in the lab, create an I2C bridge between dSPCAE and the controller to properly emulate the MPU6050
- Compare the behaviour of the real unstable vehicle with the HIL test to verify that the prepared controller test station is indeed a plausible representation of the real system and can be further used for testing and developing new control algorithms.

4 Solution process and results

4.1 Control system requirements

The general requirements for the control system have already been described in the final thesis of Zouhar [2] and since there was no need to change these requirements in the process of other works, we will consider them here as well. The basic requirements can be divided into two main groups. Stabilization in place without a rider and stabilization/riding with a rider. For both situations, the following criteria have been established.

Control system requirements for driverless vehicle operation:

- The vehicle must be able to stand on its own without oscillating around the equilibrium position after the external influences have disappeared.
- The vehicle shall not drive away spontaneously.
- The vehicle shall be allowed to move when an external force is applied, but still only in the upright position.

Requirements for the control system when operating a vehicle with a rider:

- The vehicle must be able to maintain stability after the rider has ”boarded”.
- It must react to the transfer of a person’s centre of gravity by driving forward or backwards.
- Tilting must be firm enough to prevent the rider from falling forward or backward.

Safety requirements:

- Preventing motors from reaching maximum speed.
- Necessity to monitor the state of charge of the batteries.
- Detection of the presence of a rider on the vehicle.
- Limit the maximum tilt angle of the platform.

4.2 Simulation model and parameter estimation

In order to design and test the control algorithm, it is necessary to create a model of the unstable vehicle that faithfully represents its dynamic behaviour. As can be seen from the research section, modelling of this type of system has been addressed in some previous thesis. However, none of them have focused on model building using the black-box method, nor on modeling using physical simulation. Therefore, this part of the thesis focuses on different approaches to modelling - first, a white-box model based on known physical principles will be briefly outlined, followed by the presentation of alternative approaches including black-box identification and physical simulation. Finally, we will compare the models in terms of their practical applicability within Model-Based Design and their accuracy in simulating real system behaviour.

4.2.1 Use of white-box modeling approach

White-box modelling can be approached from the very beginning of the development process, as there is no need for real measured data. This approach will give us an idea within MBD of what physical values our system will be operating at and will give us an initial indication of, for example, the motor power that will need to be used to keep the system operating within the chosen limits. In this work, as part of the white-box modelling, we derive the differential equations of the simplified system shown in Fig. 4.1 and model the dynamic behaviour of the vehicle using physical simulation in Simscape.

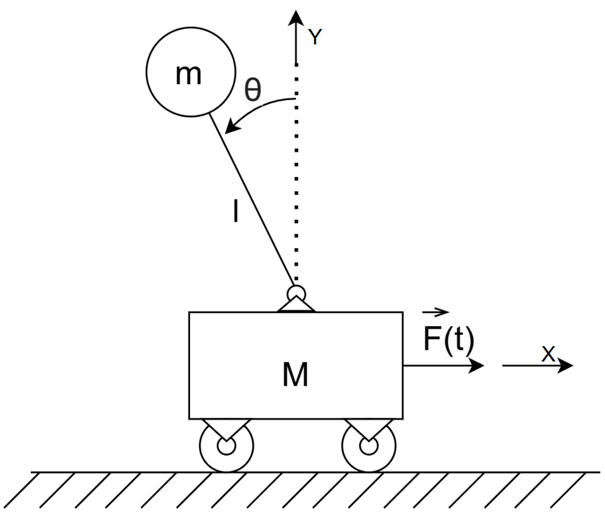


Figure 4.1: Schematic model of an unstable balancing vehicle

In the Fig. (4.1) we can see a schematic drawing of an unstable balancing vehicle, where m is the mass of the upper body or rider on the vehicle, M is the mass of the base, $\overrightarrow{F}$ is the force input acting on the lower body, l is the arm on which the mass m is located, and θ is the rotation of the body m with respect to a plane perpendicular to the base. In this simplified scheme, we do not consider any frictional losses. At the same time, only horizontal motion is assumed here.

The following derivation uses the second order Lagrangian formulation, based on kinetic and potential energy, to obtain the equations of motion. The resulting dynamic model is expressed both in scalar form and later transformed into matrix notation, which is more convenient for simulation and control design.

Kinematic Equation

$$x_m(t) = x(t) - l\sin\theta(t) \tag{4.1}$$

$$y_m(t) = l\cos\theta(t) \tag{4.2}$$

Where x_m and y_m are the positions of the solid m and l is the length of the arm. The position of the carriage is x.

Next, we calculate the potential energy of the system. Since we consider motion only along the horizontal plane, we can declare the potential energy of the cart to be zero.

Potential Energy

$$E_p(t) = mgy_m(t) = mgl\cos\theta(t) \tag{4.3}$$

Next, calculate the kinetic energy of the system.

Kinetic Energy

$$E_k(t) = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}m(\dot{x}_m^2 + \dot{y}_m^2) \tag{4.4}$$

After breaking down and inserting the derivatives from the kinematic equations, we get the following relation.

$$E_k(t) = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}m[\dot{x}^2 - 2l\dot{\theta}\dot{x}\cos\theta + l^2\dot{\theta}^2(\cos^2\theta + \sin^2\theta)] \tag{4.5}$$

The resulting relation for calculating the kinetic energy of the system can be written as follows.

$$E_k(t) = \frac{1}{2}(M+m)\dot{x}^2 + \frac{1}{2}ml^2\dot{\theta}^2 - ml\dot{\theta}\dot{x}\cos\theta \tag{4.6}$$

Lagrange equation

To derive the equations of motion, we start from the basic formulation of the Lagrange equation of the second kind.

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{q}_i}) - \frac{\partial L}{\partial q_i} = Q_i \tag{4.7}$$

Where the Lagrangian is the difference between the Kinetic and Potential energy of our system. After substituting the equations (4.3) and (4.6) into the equation (4.8) we get the Lagrangian prescription for our system, see Equation (4.9).

$$L = E_k - E_p \tag{4.8}$$

$$L = \frac{1}{2}(M+m)\dot{x}^2 + \frac{1}{2}ml^2\dot{\theta}^2 - ml\dot{\theta}\dot{x}\cos\theta - mgl\cos\theta \tag{4.9}$$

Equations of motion

Since we have a system with two degrees of freedom, we also have two generalized coordinates, where the first is $q_1 = x$ and the second is $q_2 = \theta$. After substituting the first generalized coordinate into (4.7), we write the following equation.

$$(M+m)\ddot{x} - ml\ddot{\theta}\cos\theta + ml\dot{\theta}^2\sin\theta = F(t) \tag{4.10}$$

After substituting the second generalized coordinate into the equation (4.7) we get the following equation.

$$l\ddot{\theta} - \ddot{x}\cos\theta - g\sin\theta = 0 \tag{4.11}$$

For clarity, the equations of the dynamic system can be written in matrix notation.

$$\mathbf{M}(\mathbf{q}(\mathbf{t}))\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}(\mathbf{t}), \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{K}(\mathbf{q}(\mathbf{t})) = \mathbf{H}u(t) \tag{4.12}$$

Where $q = (x, \theta)^T$. $u(t) = F$[N] represents external control forces applied to the system.

The matrix **M** can be called the mass/inertia matrix.

$$M = \begin{bmatrix} (M+m) & -mL\cos\theta \\ -\cos\theta & L \end{bmatrix} \tag{4.13}$$

C is also known as the Coriolis and centrifugal matrix.

$$C = \begin{bmatrix} 0 & mL\dot{\theta}\sin\theta \\ 0 & 0 \end{bmatrix} \tag{4.14}$$

K is a vector of generalized forces derived from potential energy.

$$K = \begin{bmatrix} 0 \\ -g\sin\theta \end{bmatrix} \tag{4.15}$$

H is an input matrix that maps the control input $u(t)$ into generalized forces.

$$H = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{4.16}$$

$$z_1 = q \tag{4.17}$$
$$z_2 = \dot{q} \tag{4.18}$$
$$\dot{z}_1 = z_2 \tag{4.19}$$
$$\dot{z}_2 = M^{-1}Cz_2 - M^{-1}K + M^{-1}Hu \tag{4.20}$$

$$\frac{d}{dt}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 0 & I \\ 0 & -M^{-1}C \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -M^{-1}K \end{bmatrix} + \begin{bmatrix} 0 \\ M^{-1}H \end{bmatrix} u \tag{4.21}$$

Alternative approach to this:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{K}(\mathbf{q}) = \mathbf{H}u \tag{4.22}$$

Which leads to little changes in a few matrices.

$$C = \begin{bmatrix} mL\dot{\theta}^2 \sin\theta \\ 0 \end{bmatrix} \tag{4.23}$$

$$\frac{d}{dt}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -M^{-1}K - M^{-1}C \end{bmatrix} + \begin{bmatrix} 0 \\ M^{-1}H \end{bmatrix} u \tag{4.24}$$

Linearization

Most control design techniques are based on linear model of the controlled system. If the mathematical model of the system is nonlinear, it has to be linearized around some appropriate working point.[21]

The linearization is performed around the nominal working point (z_n, u_n) by the first terms of the Taylor series:

$$\dot{z} = \dot{z}_n - \Delta\dot{\theta} = F(z_n, u_n) + \frac{\partial F(z_n, u_n)}{\partial z}\Delta z = \frac{\partial F(z_n, u_n)}{\partial u}\Delta u \tag{4.25}$$

Where the matrices of partial derivatives by all states and input are.

$$A = \frac{\partial F(z_n, u_n)}{\partial z} = \begin{bmatrix} \frac{\partial F_1(z_n,u_n)}{\partial z_1} & \cdots & \frac{\partial F_1(z_n,u_n)}{\partial \dot{z}_2} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_4(z_n,u_n)}{\partial z_1} & \cdots & \frac{\partial F_4(z_n,u_n)}{\partial \dot{z}_2} \end{bmatrix} \tag{4.26}$$

Physical modelling using Simscape multibody

For the physical modeling, I used ready-made 3D models from the thesis of Zouhar mentioned in the research section [2].

Another possible approach to model a dynamic system using the white-box method is to use a physical simulator. In contrast to the previously mentioned method, where we have used a system of differential equations (ODEs), physical software “creates” a system of differential algebraic equations (DAEs), which is more computationally demanding, but the construction of such a model does not require knowledge of differential equations. To build this model, the Simscape multibody tool was used, into which an existing 3D model (Fig. 4.2) in step format can be inserted. The subsequent connection of the individual parts of the model to each other is done by using “rigid transform” blocks and the possibility of mutual movements between each other, for example, by using “revolute joint” blocks and others.

The advantage of using already created 3D parts of the unstable vehicle model is that we are able to specify the weight or density of the individual parts in Simscape and thus get as close as possible to the behaviour of the real vehicle. Within MBD, using such simulated dynamic behaviour is simple and fast and can be worked with early in the project, before the actual prototype is created.

Some simplifications were taken into account when modelling the unstable vehicle in Simscape. The torque to revolute joint was chosen as the inputs to the motors that drive

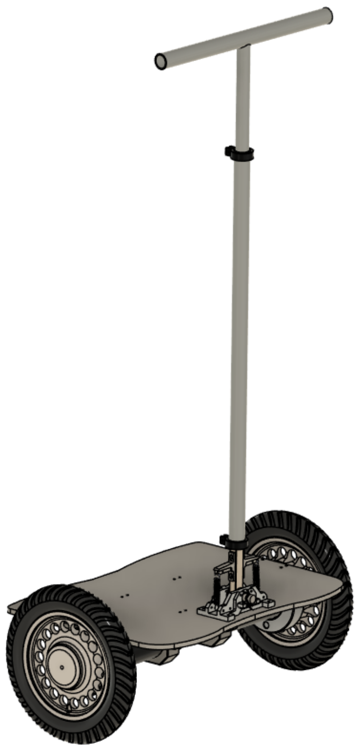


Figure 4.2: 3D model of an unstable balancing vehicle in Fusion360

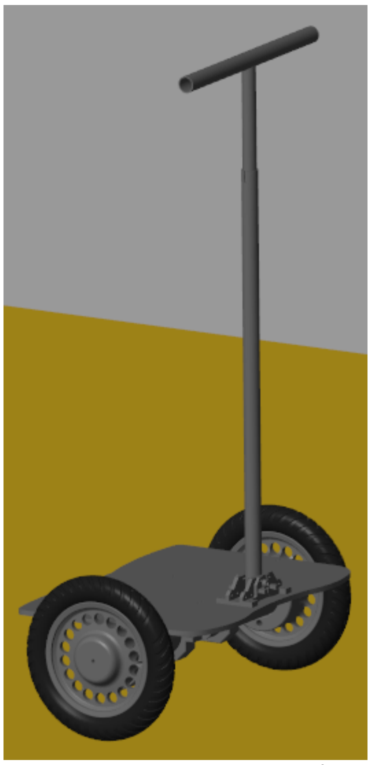


Figure 4.3: 3D model of an unstable balancing vehicle in Simulation environment in Simscape Multibody

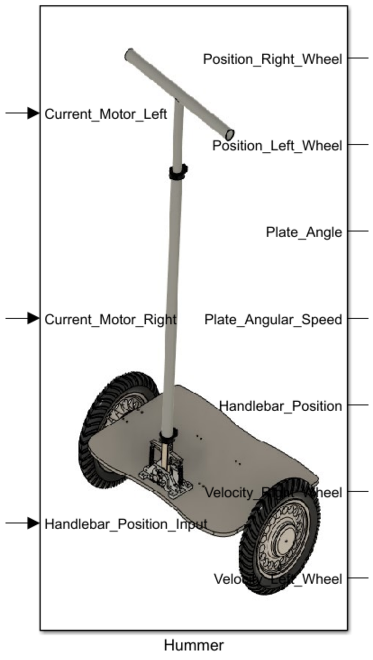


Figure 4.4: Function block of unstable balancing vehicle in Simscape Multibody

the vehicle itself, neglecting the dynamics of the current in the motor, as the mechanical time constant was assumed to be an order of magnitude larger than the electrical time constant. At the same time, the real vehicle is current controlled from the ESCON 50/5 drivers, so the input to the function block created in Simscape is the current flowing into the motor, which is then converted via the motor constant and gear ratio to the torque at the wheel.

Finally, a function block was created in the Simscape Multibody program (Fig.4.4), where its inputs are the currents to the motors and the handlebar tilt and outputs are the individual states of the unstable vehicle itself. When using this function block, the user sees a 3D visualization of the dynamic behavior of the vehicle itself (Fig.4.3).

A 6-DOF function block (Fig.4.5) was used to move the balancing vehicle freely. However, its disadvantage is that it expresses the vehicle rotation in quaternions and therefore it is advisable to convert it to Euler angles.

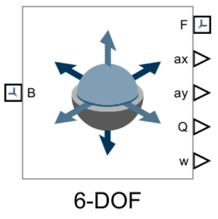


Figure 4.5: 6-DOF joint in Simscape Multibody

Roll (X-axis rotation)

$$\phi = \arctan \frac{2(q_w q_x + q_y q_z)}{1 - 2(q_x^2 + q_y^2)} \tag{4.27}$$

Pitch (Y-axis rotation)

$$\theta = -\frac{\pi}{2} + 2 \arctan \frac{\sqrt{1 + 2(q_w q_y - q_x q_z)}}{\sqrt{1 - 2(q_w q_y - q_x q_z)}} \tag{4.28}$$

Yaw (Z-axis rotation)

$$\psi = \arctan \frac{2(q_w q_z + q_x q_y)}{1 - 2(q_y^2 + q_z^2)} \tag{4.29}$$

4.2.2 Use of grey-box modeling approach

The modelling of a dynamic system by the grey-box method consists in the known behaviour of the dynamic system described by a differential equation and the parameters of the differential equation are obtained from the measured data by means of optimization (minimizing the residual between the measured and simulated data). There are several ways we can approach to obtain the parameters of the unstable vehicle by the grey-box method; for example, we can divide the whole complex unstable system into smaller and stable parts and perform parameter estimation for each part. In our case, we will follow the second way by trying to identify the system as a whole.

We will apply the parameter estimation not to the real vehicle, but to the data obtained from the previous part of the white-box modelling, namely the SimScape model. One reason for this is that if we were still in the MBD phase of development when the vehicle was not yet assembled, we would have no way to measure real data. Another reason is that simulating a system in Simscape multibody takes much more computational time than simulating it via differential equations. In fact, in a later stage of this work, we will also want to look at whether the Simscape model actually represents the dynamics of the system in a plausible way. By obtaining the parameters of the Simscape model, we are then able to create an advanced control system that needs to know the model of the system to function, and this will serve as an initial sketch for controlling a real vehicle.

From the research part of the thesis, we know that the estimation of unstable vehicle parameters has already been dealt with by Dobossy [4], and therefore I will use the initial estimation of his results and the equations associated with them for a start.

$$\ddot{\beta}(m_w R^2 + m_p R^2 + I_w) + \ddot{\varphi}(m_p L R \cos\varphi) = \dot{\varphi}^2 L m_p \sin\varphi - M \tag{4.30}$$

$$\ddot{\beta}(m_p L R \cos\varphi) + \ddot{\varphi}(L^2 m_p + I_{p,y}) = m_p g L \sin\varphi + M \tag{4.31}$$

The given differential equation is further linearized according to the following criteria.

$$\sin\varphi \approx \varphi \tag{4.32}$$

$$\cos\varphi \approx 1 \tag{4.33}$$

$$\dot{\varphi}^2 \approx 0 \tag{4.34}$$

The linearized differential equation looks like this.

$$\ddot{\beta}(m_w R^2 + m_p R^2 + I_w) + \ddot{\varphi}(m_p L R) = -M \tag{4.35}$$

$$\ddot{\beta}(m_p L R) + \ddot{\varphi}(L^2 m_p + I_{p,y}) = m_p g L \varphi + M \tag{4.36}$$

We then convert the linearized differential equations into the form of a state model. Since the relation would gain volume when recalculated, let us introduce new constant parameters:

$$\begin{aligned} a &= m_w R^2 + m_p R^2 + I_w \\ b &= m_p L R \\ c &= L^2 m_p + I_{p,y} \\ d &= m_p g L \end{aligned}$$

After a few adjustments, we get the following equations:

$$\ddot{\varphi} = \frac{M(1+\frac{b}{a}) + \varphi d}{-\frac{b^2}{a} + c} \tag{4.37}$$

$$\ddot{\beta} = \varphi \frac{db}{(-\frac{b^2}{a} + c)a} + M\left(\frac{(1+\frac{b}{a}b)}{(-\frac{b^2}{a}+c)a} - \frac{1}{a}\right) \tag{4.38}$$

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}$$

Where:

$$x = \begin{bmatrix} \varphi \\ \beta \\ \dot{\varphi} \\ \dot{\beta} \end{bmatrix}, A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{d}{-\frac{b^2}{a}+c} & 0 & 0 & 0 \\ \frac{db}{-b^2+ca} & 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ \frac{1+\frac{b}{a}}{-\frac{b^2}{a}+c} \\ \frac{b+\frac{b^2}{a}}{-b^2+ca} - \frac{1}{a} \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, D = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

As we can see from the state (system) matrix A, the dynamics of the system do not depend on the rotation of the wheels of the vehicle (β) and at the same time if we want to design a controller for tilt, we do not have to take into account either the position or the speed of the wheels and our system is simplified to a system with one degree of freedom. Of course, the dynamics itself is still tied to the parameters of both bodies. This assumption can also be used for system identification (discussed later in this work).

$$x = \begin{bmatrix} \varphi \\ \dot{\varphi} \end{bmatrix}, A = \begin{bmatrix} 0 & 1 \\ \frac{d}{-\frac{b^2}{a}+c} & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ \frac{1+\frac{b}{a}}{-\frac{b^2}{a}+c} \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, D = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

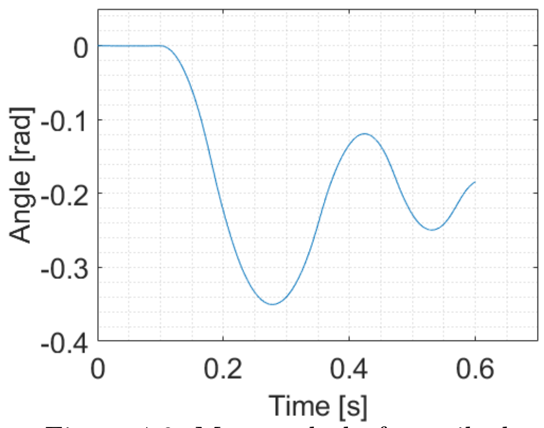


Figure 4.6: Measured platform tilt data

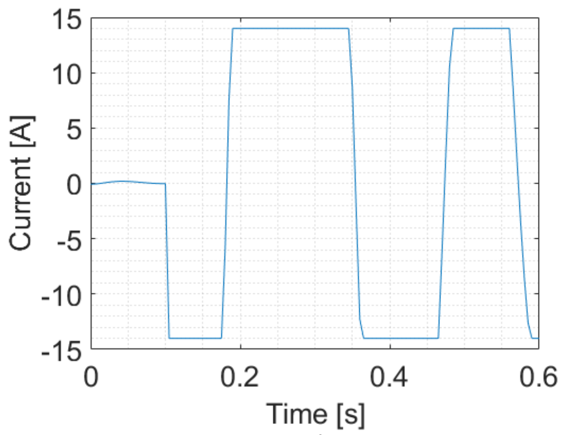


Figure 4.7: Input data for parameter estimation

In the graphs (Fig.4.7 and Fig.4.6) we can see the measured data from the simulation of the unstable vehicle in Simscape. In the graph (Fig.4.8) we can see the result of parameter estimation. The discrepancies between the individual signals can be explained by the simplified model, as we have neglected friction. Furthermore, the vehicle in the Simscape simulation may slip a wheel, etc.

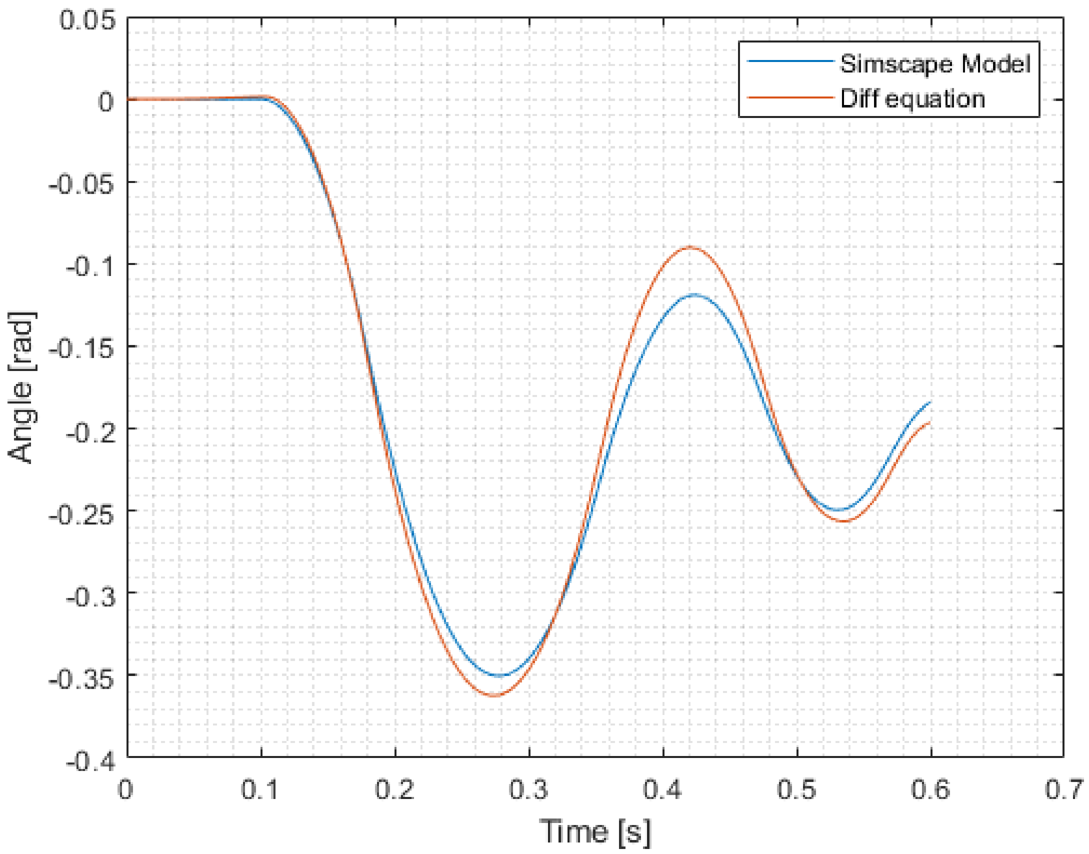


Figure 4.8: Estimated parameters of the simscape model

Estimating the parameters of an unstable vehicle as a whole has some disadvantages.

The system is unstable, which means that during parameter estimation the roll will drift away several times, even if the parameters are found ”correctly enough”. Next, a sensitivity analysis of each parameter was performed and with expectations, the arm length parameter L came out as sensitive. The explanation is its squared term in the notation of the differential equation. Another hurdle with parameter estimation was the correlated parameters.

4.2.3 Use of black-box modeling approach

Another way of modelling the dynamic behaviour of an unstable vehicle is to use a black-box modelling approach. However, in the MBD framework, this is the stage where the vehicle is already manufactured. The data to determine the dynamic behaviour can be obtained by using an existing controller and programming the required inputs to the system or by using RCP and controlling the unstable vehicle using, for example, dSPACE. The disadvantage of this approach in our case is the instability of the vehicle itself, as we cannot just choose an input signal that would be ideal for black-box identification, as the vehicle would immediately crash. Therefore, we need to have at least a small part of the controller already working in order to measure enough data to identify the system as a whole.

To identify the system using the black-box method, the following data were measured.

- Sine wave (Fig. 4.9)
- Repeating stair sequence (Fig. 4.10)

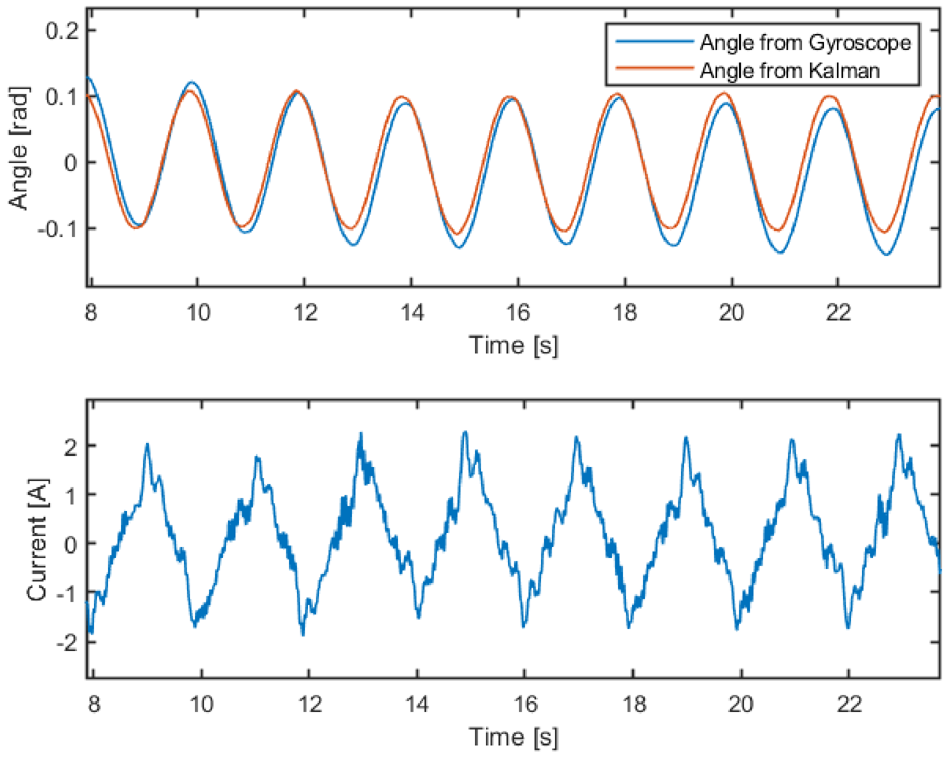


Figure 4.9: Measured data for system identification - Sine wave

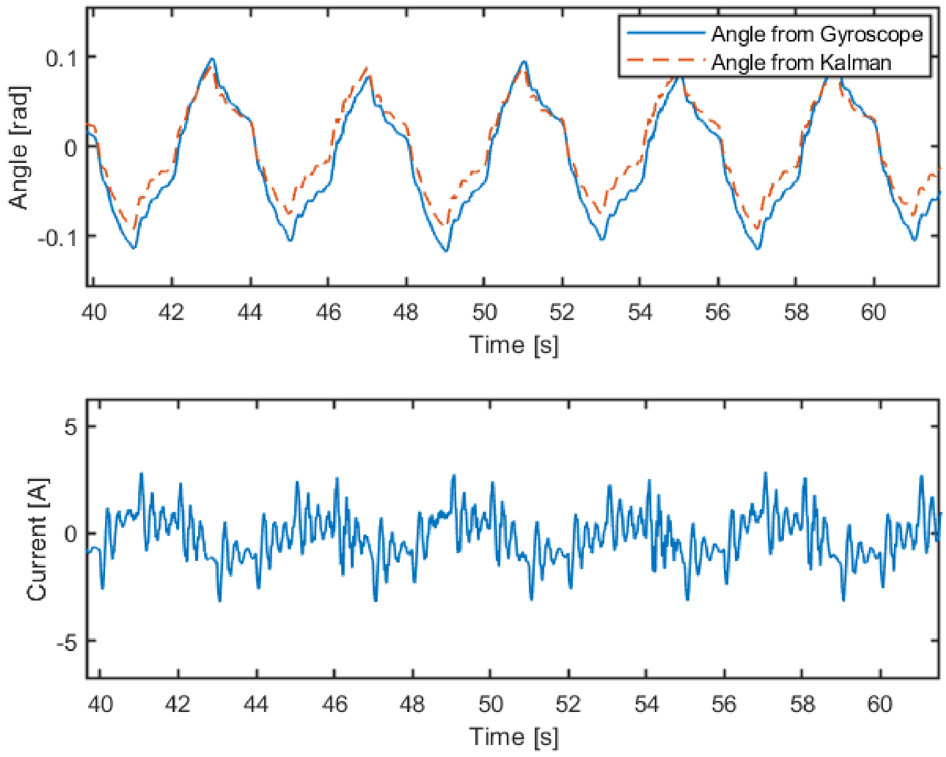


Figure 4.10: Measured data for system identification - Repeating stair sequence

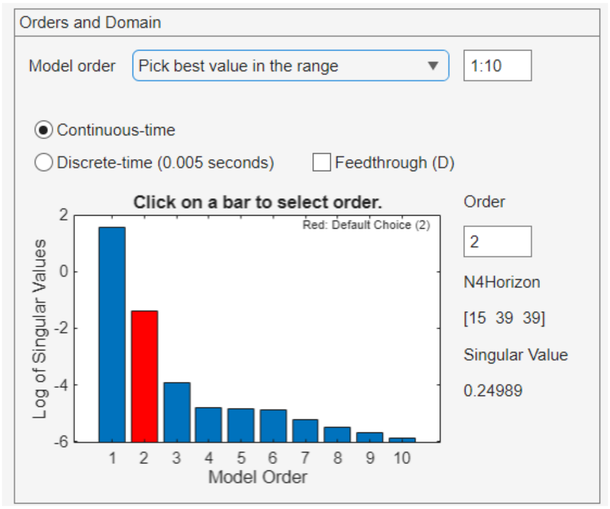
Selecting the state space system order for system identification

When identifying linear systems, the system can be described by various forms, for example, Polynomial model, StateSpace, Transfer function. For our purpose, a state space model in canonical form was chosen, since by writing it with knowledge of our system we are able to say exactly what $x_1, \dot{x}_1, x_2, \dot{x}_2$ are. After choosing to identify the system using the state space model, we need to determine the order of the system, which we are able to determine from our knowledge of the system, or we can use the tools in the System Identification toolbox from MathWorks. As we can see in (Fig.4.11) the second order system sufficiently describes the dynamic behavior of the unstable vehicle, which is what we arrived at for the derived equations for parameter estimation in grey-box modeling.

As we can see from the graphs (Fig.4.12) and (Fig.4.13), the identified system credibly describes the dynamics of the real system, and its model can be used in the design of control algorithms and in the validation of control algorithms in offline simulations.

Comparison of simscape estimation and identification on a real device

We have currently shown 3 ways to obtain a system model, but there is still the important question of how much the systems differ from each other. In the white-box model approach, we currently omit the model we obtained by computing a Lagrange of the second kind, since obtaining the parameters of the systems so that the outputs faithfully represent reality would involve a significant amount of work. We therefore focus on comparing the Simscape multibody physical model (white-box) and system identification (black-box).



Orders and Domain

Model order: Pick best value in the range — 1:10

● Continuous-time

○ Discrete-time (0.005 seconds)

☐ Feedthrough (D)

Order: 2

N4Horizon: [15 39 39]

Singular Value: 0.24989

Figure 4.11: Selecting the state space system order for system identification

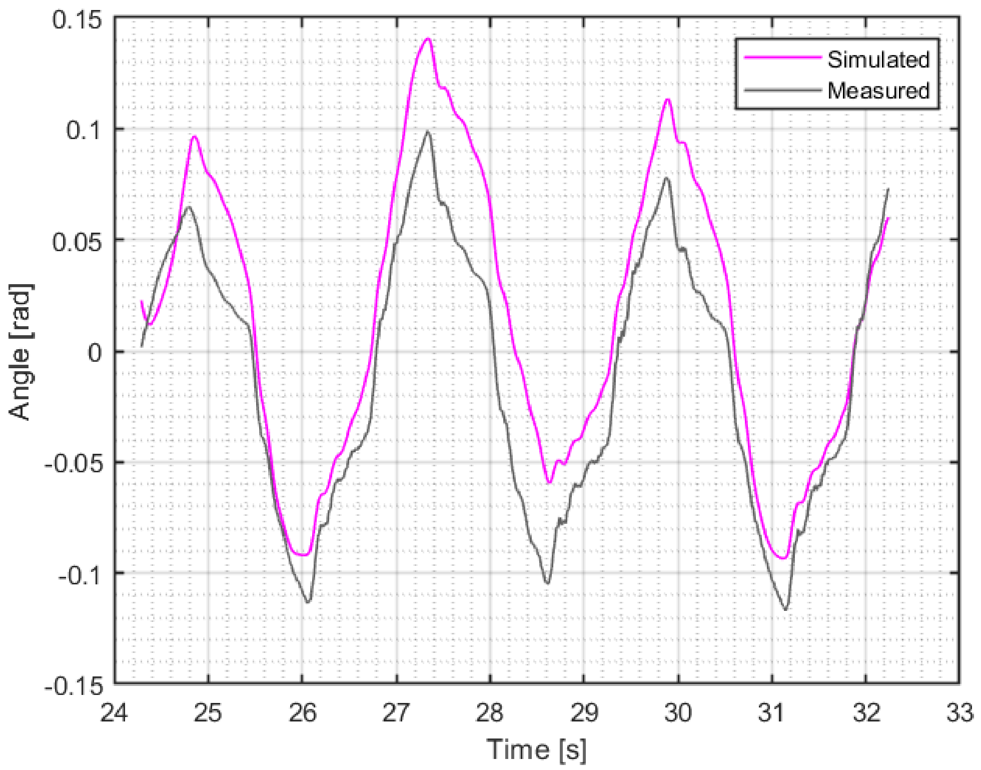


Figure 4.12: Indentified system using second order State Space model - Simulation

From the graph (Fig. 4.14) we can see a comparison of the behavior of the system obtained by the black-box modeling approach and the system from the Simscape multibody environment. Looking at the plot, we can conclude that the system responses are almost identical, which implies that we can use either of the models created for controller design or testing of control algorithms.

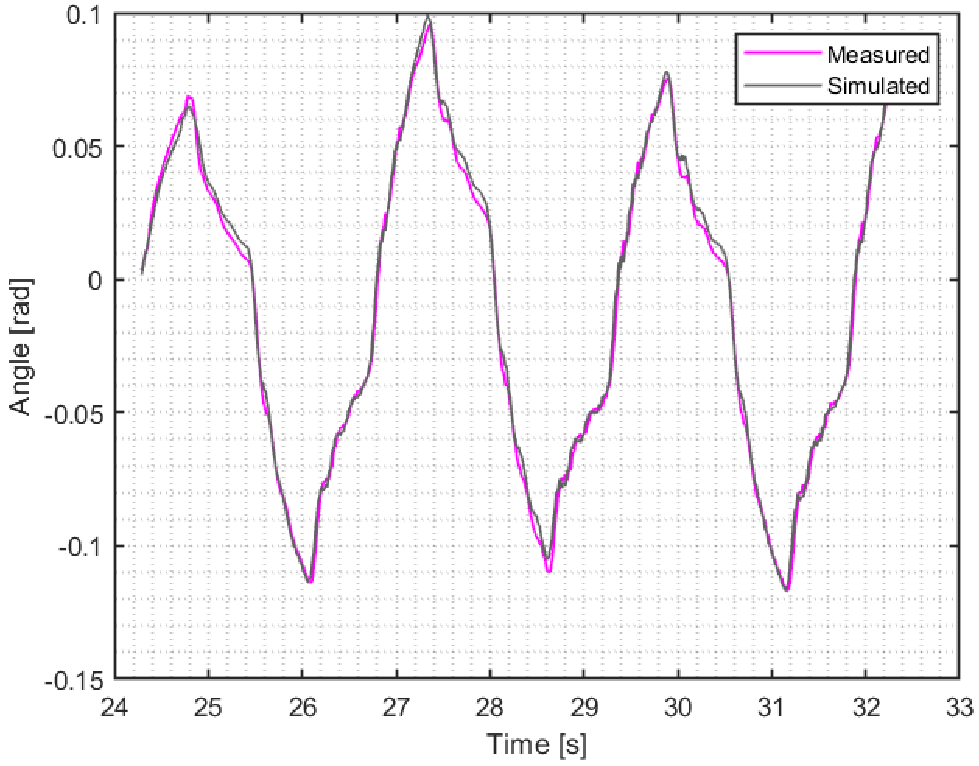


Figure 4.13: Indentified system using second order State Space model - Ten step prediction

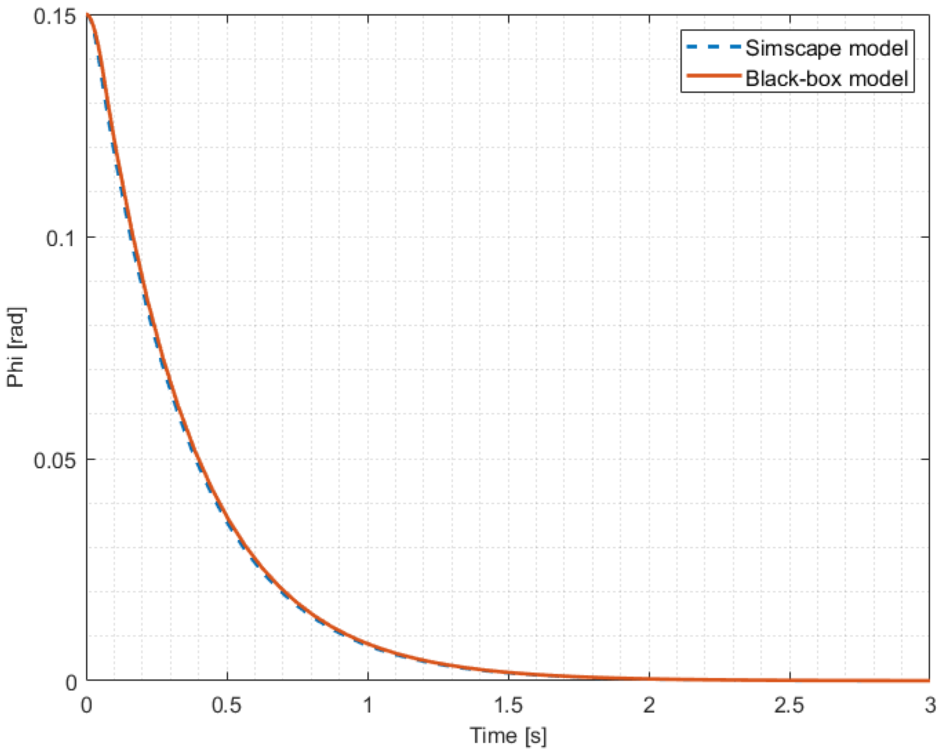


Figure 4.14: Comparison of Simscape model and Black-box model

4.3 Design and validation of control algorithms

In the design phase of the control system for the unstable balancing vehicle, several widely recognized control strategies were considered and implemented, including PID control[22, 23], Sliding Mode Control (SMC)[24], Model Predictive Control (MPC)[25], and Linear Quadratic Regulator (LQR)[26]. These algorithms were selected based on their frequent occurrence in scientific literature addressing similar underactuated and nonlinear systems, confirming their relevance and suitability for this application.

There are many possibilities of controlling an unstable balancing vehicle and many papers deal with this problem, but most of them apply the control algorithms to a scaled-down version of the unstable vehicle, which is not supposed to transport people. On the other hand, some of them deal more with the non-holonomy of the vehicle itself and the algorithms for controlling the trajectory of the vehicle itself, which in our case we do not have to deal with[27, 28, 29, 30].

For each of the control strategies, the theoretical background is briefly introduced, followed by the derived control equations, the corresponding control scheme, and the results obtained through simulation. The simulations were conducted using a linearized model of the system under the assumption of an ideal linear time-invariant (LTI) behavior without external disturbances or parameter variations. Under these conditions, all implemented control algorithms demonstrated very similar performance, successfully stabilizing the vehicle around the desired upright position. This validates the theoretical expectations and further supports the choice of the selected controllers for detailed analysis and future hardware-in-the-loop (HIL) testing.

4.3.1 Control algorithms

PID

The PID controller (Fig.4.15) was selected as the initial control strategy due to its simplicity and wide use in practice. It was tuned based on the linearized model of the system. Using PID control, the balancing vehicle was successfully stabilized around the upright equilibrium (Fig.4.16). Although nonlinearities and constraints are not directly handled by this method, very good results were achieved in the LTI simulation environment.

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt} \tag{4.39}$$

where:

- $e(t) = \theta_{\text{ref}} - \theta(t)$ is the control error
- K_p, K_i, K_d are the gain constants of the PID controller
- $u(t)$ is the control input (motor current)

Usage:

- Can be used for balance and speed control.
- Often multiple PID controllers are used - e.g. one for pitch stabilization and one for attitude control.

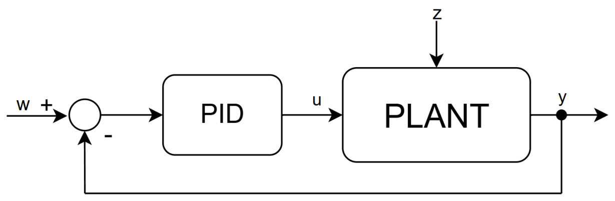


Figure 4.15: Block diagram of PID controller

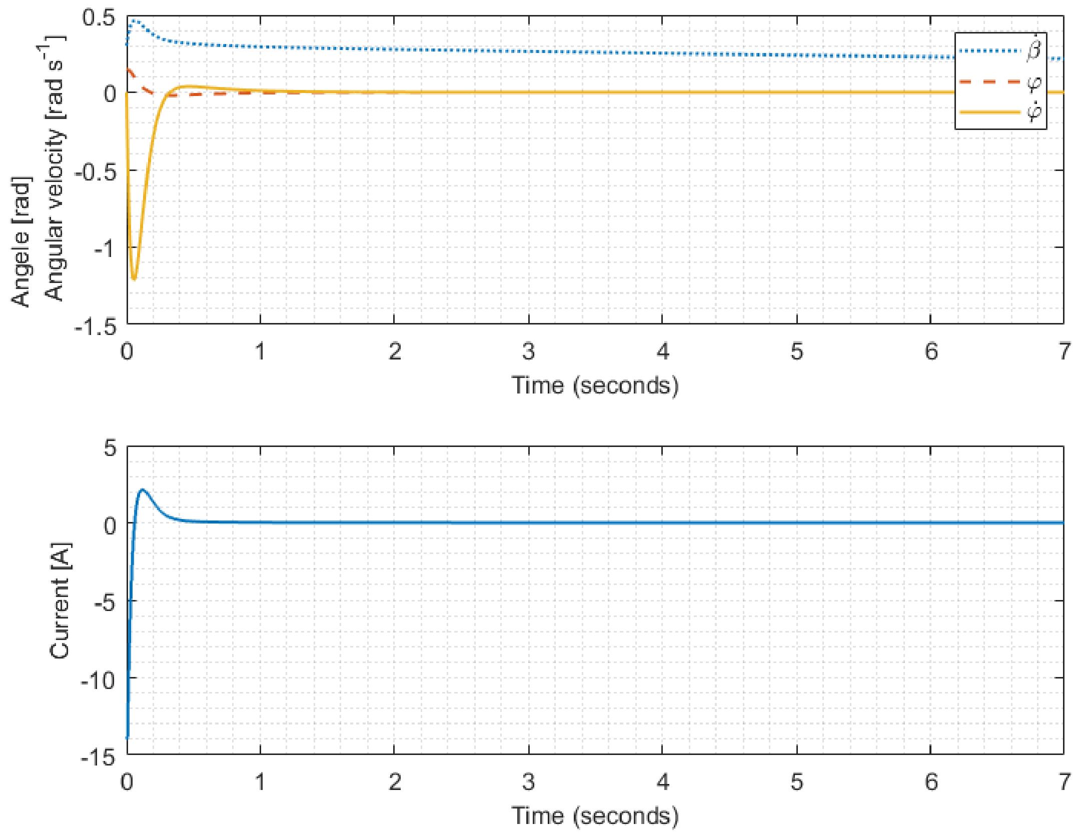


Figure 4.16: Stabilization of the balancing vehicle: system response and control signal generated by the PID controller.

Disadvantages:

- Debugging is difficult due to system instability.
- Does not take into account the dynamics of the whole system, which can lead to poorer performance during rapid changes.
- Can be sensitive to interference and wheel slips.

SMC (Sliding Mode Control)

Sliding Mode Control (Fig.4.17) was implemented due to its strong robustness against disturbances and model uncertainties, as often cited in the literature [31]. Using SMC, stabilization of the balancing vehicle was successfully achieved (Fig.4.18). However, the phenomenon of "chattering" was observed, which could affect the actuators if not properly mitigated.

Define the sliding surface:

$$s(x) = \dot{\theta} + \lambda\theta \tag{4.40}$$

Control law:

$$u = -K\,\mathrm{sgn}(s(x)) \tag{4.41}$$

A smoother approximation can be used to reduce chatter:

$$u = -K\frac{s(x)}{|s(x)| + \delta} \tag{4.42}$$

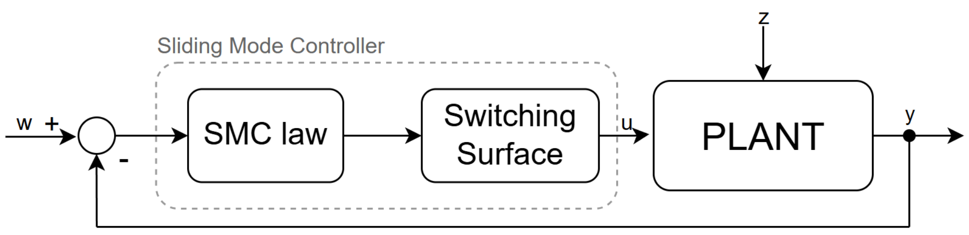


Figure 4.17: Block diagram of Sliding mode controller

Usage:

- Suitable for robust Segway tilt control.
- Can compensate well for model uncertainties (e.g. rider weight change).
- Effective for fast changes and disturbances.

Disadvantages:

- Chattering (rapid tilt oscillations) can be a problem and must be suppressed by filtering.
- More computationally intensive than PID.

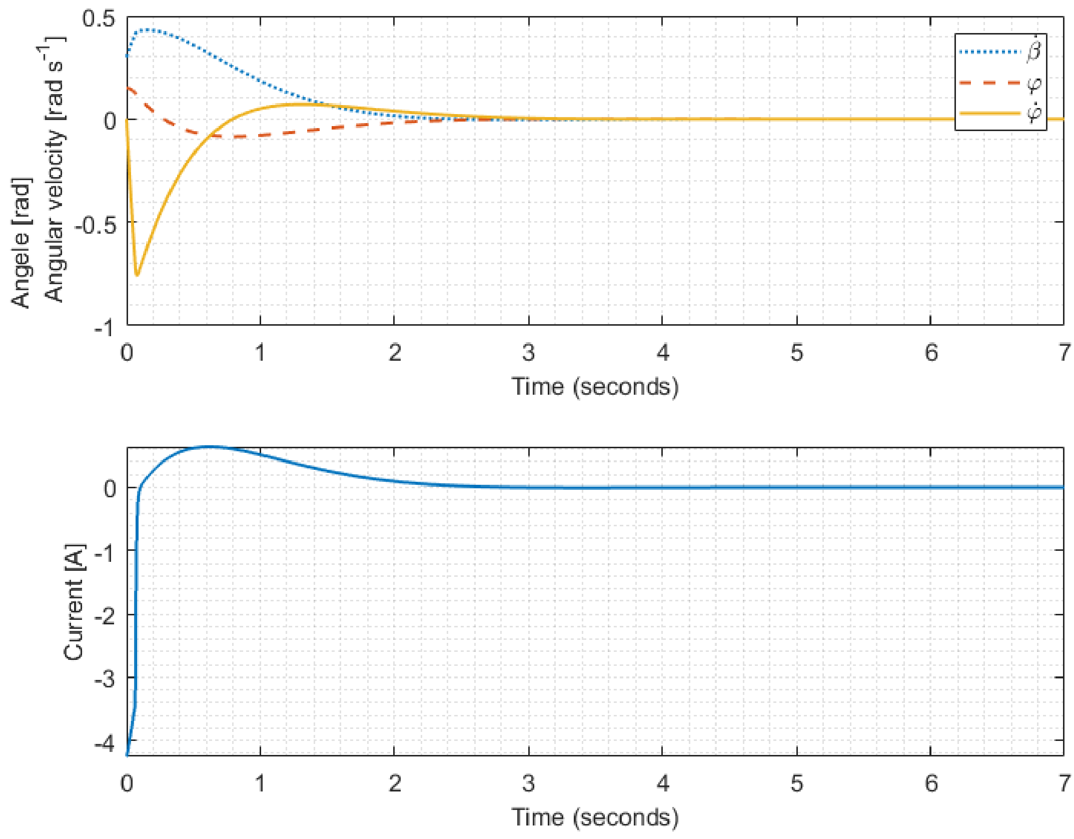


Figure 4.18: Stabilization of the balancing vehicle: system response and control signal generated by the SMC controller.

LQR (Linear-Quadratic Regulator)

The LQR controller was designed (Fig.4.19) based on the linearized model of the balancing vehicle. By optimizing a quadratic cost function, very smooth control was achieved (Fig.4.20). Since the system was well approximated by a linear model, LQR provided suitable stabilization results with minimal computational effort, making it highly suitable for real-time applications.

System state model:

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

Optimal control law:

$$u = -Kx \tag{4.43}$$

The feedback matrix K is given by:

$$K = R^{-1}B^{T}P \tag{4.44}$$

where P is the solution of the algebraic Riccati equation:

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{4.45}$$

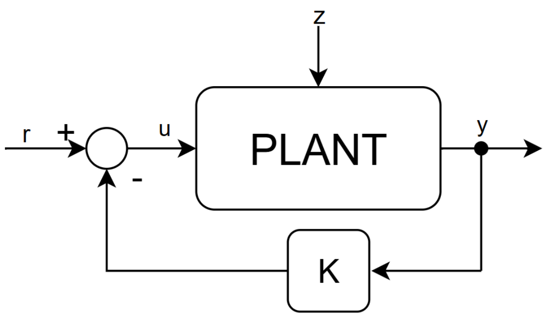


Figure 4.19: Block diagram of Linear Quadratic Regulator

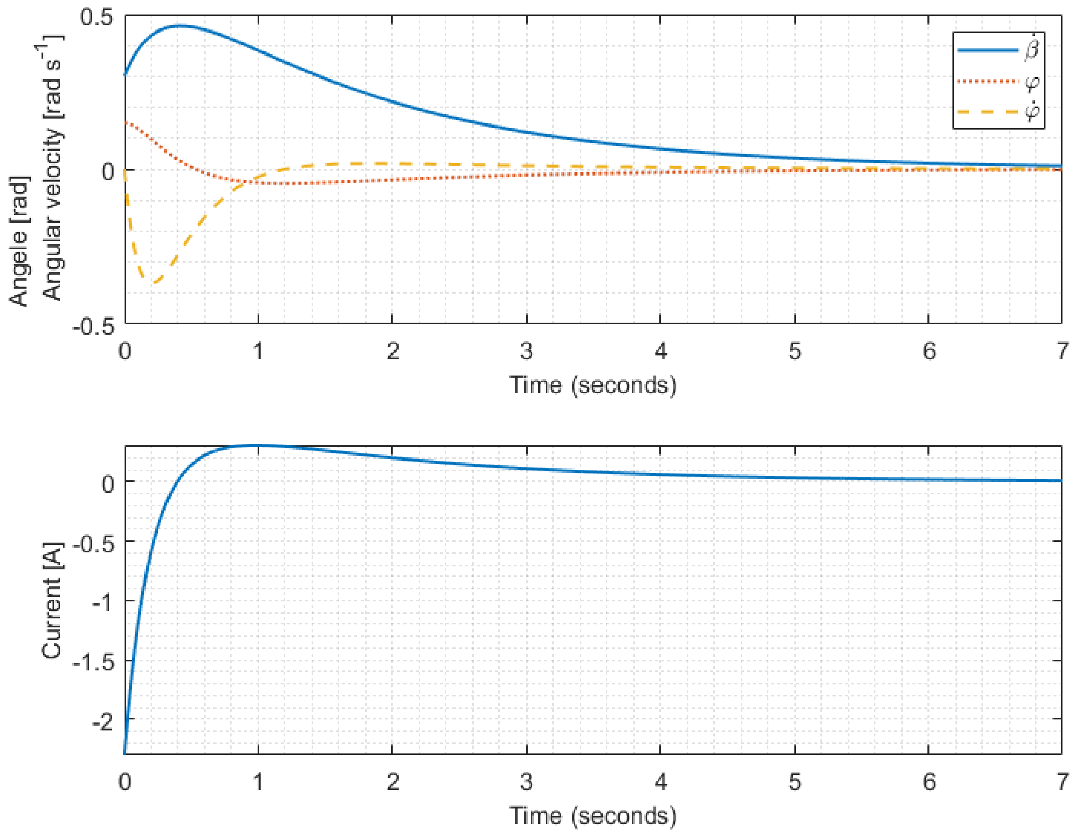


Figure 4.20: Stabilization of the balancing vehicle: system response and control signal generated by the LQR controller.

Usage:

- Very suitable for stabilizing the balance of the Segway.
- Takes into account the whole system dynamics and minimizes the error in optimal

control.

- Works well with small deviations from the equilibrium position.

Disadvantages:

- Requires an accurate model of Segway dynamics.
- Does not account for constraints on action variables (e.g. maximum motor power).
- May have problems with non-linearities at high inclinations.

MPC (Model Predictive Control)

Model Predictive Control (Fig.4.21) was chosen because of its ability to explicitly handle system constraints and predict future behavior. The balancing vehicle was controlled using MPC by solving an optimization problem at each simulation step. While very good results were obtained (Fig.4.22), it was noted that the computational requirements are higher compared to simpler controllers, which may limit practical implementation on embedded hardware.

Optimization problem for MPC:

$$\min_{u_k,\dots,u_{k+N}} \sum_{i=k}^{k+N} (x_i^T Q x_i + u_i^T R u_i) \tag{4.46}$$

System dynamics:

$$x_{k+1} = Ax_k + Bu_k \tag{4.47}$$

Constraints on input:

$$u_{\min} \leq u_k \leq u_{\max} \tag{4.48}$$

Calculation steps:

- Prediction of future states using a dynamic model.
- Control optimization at each step to minimize error.
- Application of first input and horizon shift.

Usage:

- Ideal for Segway control, especially if we want both stabilization and trajectory planning.
- Allows you to predict future behaviour and include constraints (e.g. maximum speed or motor acceleration).

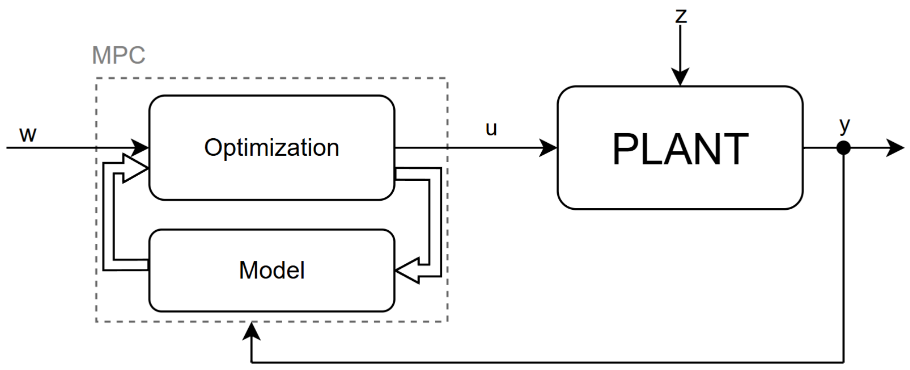


Figure 4.21: Block diagram of Model Predictive Controller

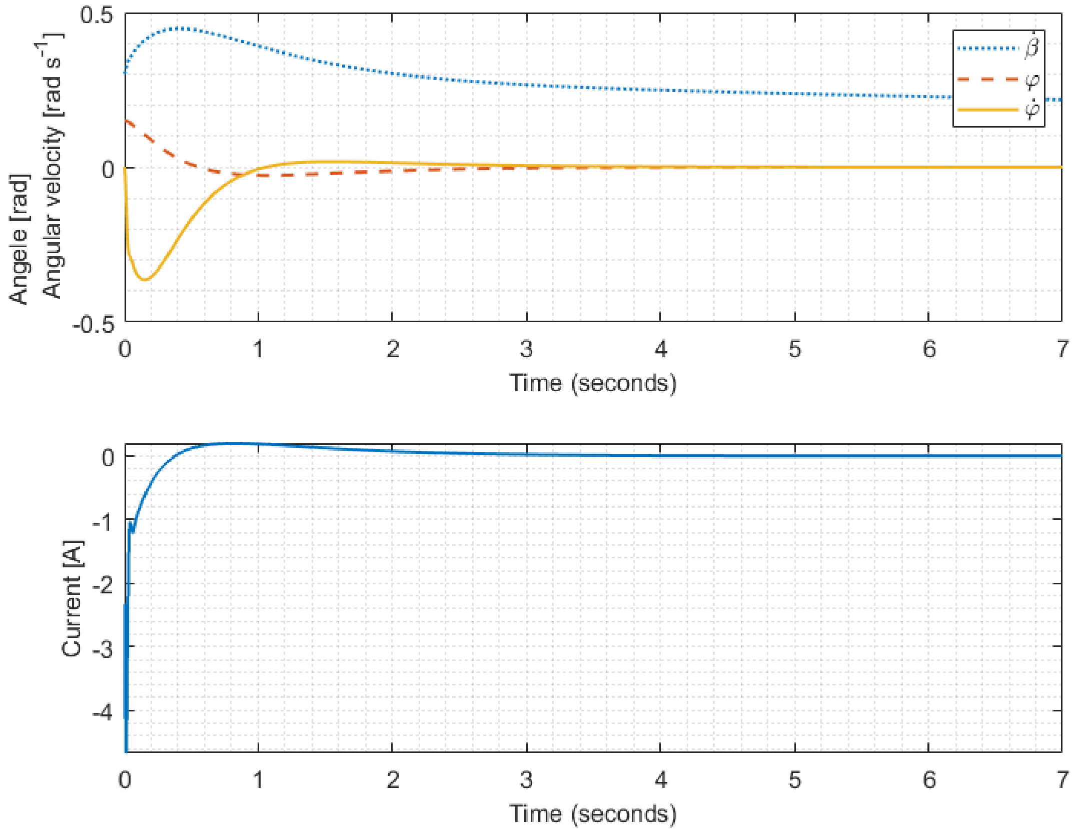


Figure 4.22: Stabilization of the balancing vehicle: system response and control signal generated by the MPC controller.

- Can adapt to changes in conditions (e.g. surface slope, rider weight).

Disadvantages:

- Highly computationally intensive - normally requires more powerful hardware (e.g. fast processors,Optimizes control inputs at every step.).

- Needs an accurate system model.

Summary

- PID is the simplest and can work, but is not ideal.
- The SMC is good for robust stabilization, but may have chattering issues.
- LQR is an optimized solution, but requires an accurate model and does not account for constraints.
- The MPC good choice for complex Segway control, but is more computationally intensive.

As far as the simulation on the LTI model with predefined deflection and zero control is concerned, we are able to obtain similar results for all tested controller types. So it depends on how the controllers behave at different system parameters. In addition, another criterion used to determine the suitability of a given controller is ride comfort and stability with the rider. A too hard or too soft regulator can be very uncomfortable for the rider. Therefore, the Sliding mode controller was chosen for the real device because it was assumed that the rider would respond best to a first-order, exponential type response and this is what the controller is trying to achieve.

4.3.2 State observer

The theoretical knowledge written in this part of the thesis is primarily obtained from *"Linear system theory and design"* [32]. This is a book that serves as a study aid in the field of linear systems and multivariable system design.

As part of the design of a state control for an unstable balancing vehicle, it is important to verify that the internal state of the system can be reconstructed based on the output measurements alone. This property, known as observability, is crucial in the design of state observers such as the Luenberger observer or the Kalman filter.

The Gram observability matrix is used to analyze observability, which allows one to evaluate whether a system is fully observable - that is, whether all internal states affect the output of the system in a measurable way. If this matrix is regular (positive definite and with a non-zero determinant), the system is considered observable. If it is close to a singular matrix, observability can be ensured in theory, but in practice reconstruction of some states is very difficult.

This information is important not only from a theoretical point of view, but also in practical control implementation, especially state control as an LQR, where accurate state estimation is crucial. The Gram matrix can also serve as a tool in optimizing the placement of sensors - helping to determine where additional measurements would contribute most to improving the reconstruction of the system state.

Controllability and observability Gramians

Given the continuous-time state-space model

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t)\\ y(t) &= Cx(t) + Du(t)\end{aligned}$$

The controllability Gramian is defined by:

Continuous-time:$W_c = \int_0^\infty e^{At} BB^T e^{A^T t} dt$ Discrete-time:$W_c = \sum_{k=0}^\infty A^k BB^T (A^T)^k$

The observability Gramian is defined by:

Continuous-time: $W_o = \int_0^\infty e^{A^T t} C^T C e^{At} \, dt$ Discrete-time: $W_o = \sum_{k=0}^{\infty} (A^T)^k C^T C A^k$ [33]

	$\det(W_o)$
Measuring φ	0.1133
Measuring $\dot{\varphi}$	0.0604
Measuring $\dot{\beta}$	191.6
Measuring $\dot{\beta}$ & φ	$4.2 \cdot 10^4$
Measuring $\dot{\beta}$ & $\dot{\varphi}$	$1.6 \cdot 10^7$

Table 4.1: A Gramian determinant for determining how well a given system is observable using certain sensors

The Table 4.1 shows that the system is in theory observable even with only one sensor, and it does not matter whether we measure the base rotation, its angular velocity, or the wheel rotation speed. However, we can say that when only one sensor is needed, it would be more appropriate to measure the wheel speed. Since this is still a simulation model at the moment, we will consider that we are measuring the angular velocity of the base and the rotational velocity of the wheels, which also came out best from the determinant of the Gramian. Furthermore, measuring only these values has a clear justification since we are able to directly obtain the angular velocity from, for example, the IMU (Inertial Measurement Unit), and the angular velocity of the wheels again by simple differentiation from the encoder.

We do not consider the position of the vehicle itself in this case, since it is crucial for us to balance the platform ($\varphi = 0$) and also to keep the vehicle from moving ($\dot{\beta} = 0$), but we are not interested in where the vehicle was turned on and in getting to the original turn-on location.

In the graph (Fig. 4.23) we can see the measured data of the simulation model of the unstable vehicle. As already mentioned, we chose the wheel rotation speed and the angular velocity of the platform as the measured signals. Since this is a simulation, we artificially affected the outputs with noise with a normal distribution and zero mean and added an input disturbance to the system as well. These signals were then used as input to the Kalman filter. The output of the Kalman filter was then used as feedback to the controller.

Using the simulation model, we are able to compare the output of the Kalman filter against the actual system states. In the graph (Fig. 4.24) we see a comparison of the actual output and the output from the Kalman filter, and it is important to focus on the state $\varphi, \hat{\varphi}$. We can see that the Kalman filter has faithfully reconstructed the actual rotation of the platform and has also filtered out the signals that were affected by noise appropriately.

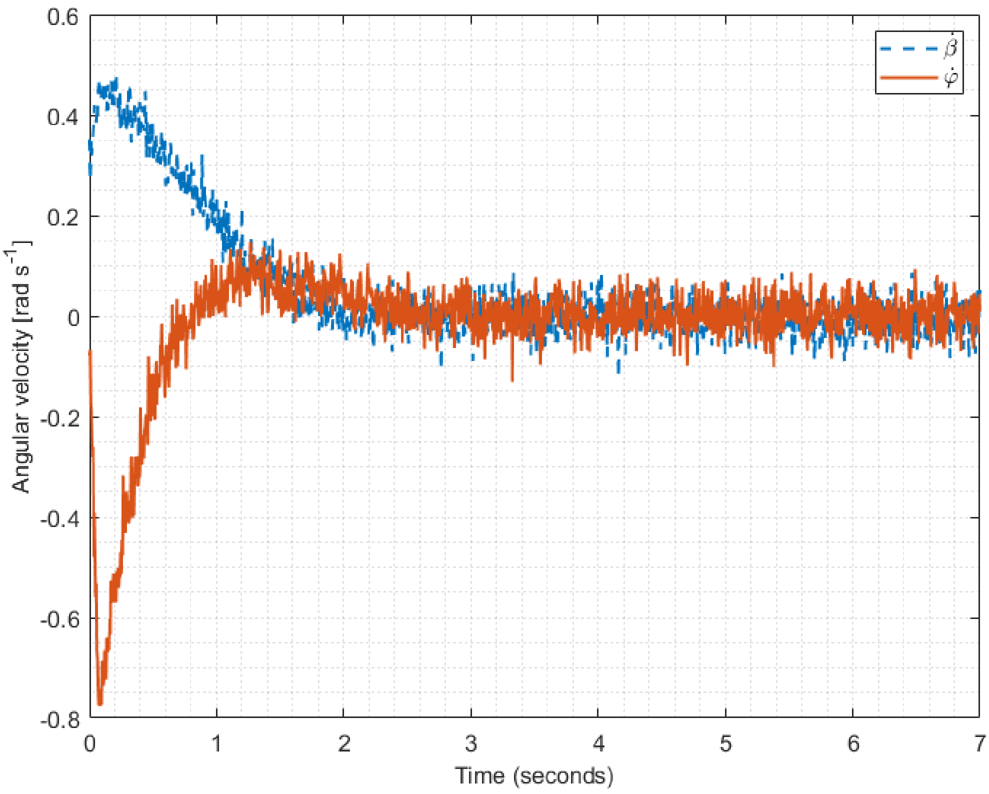


Figure 4.23: Measured data as an input to the Kalman filter

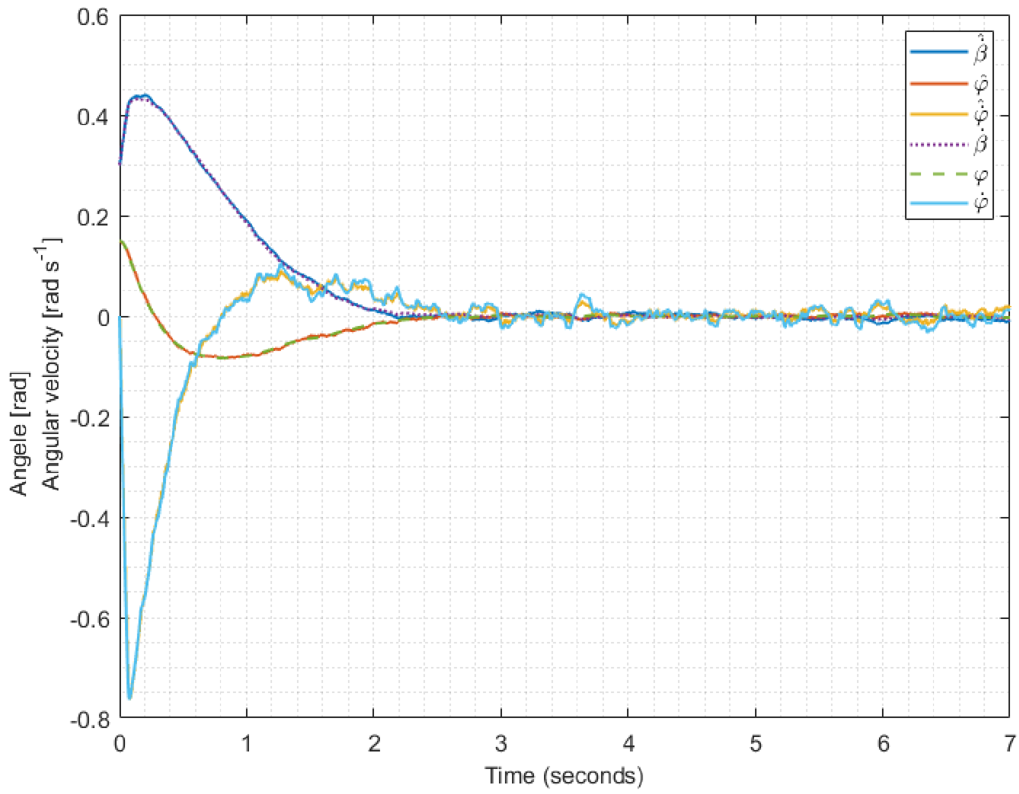


Figure 4.24: Comparison of the outputs from the real system and the Kalman filter, where the system was controlled by SMC with feedback from the Kalman filter

4.3.3 Safety algorithms

Forward speed limit

One of the safety algorithms that was taken into account was the maximum forward speed limit. The speed limit is important for two factors. One is that the speed of the motor is directly proportional to its supply voltage, and since we are limited by the voltage of the "battery" source, when a certain critical speed is reached, the motor might no longer be physically able to get the vehicle upright or start braking, since braking the vehicle actually means accelerating enough to make the vehicle "trip" our feet. The second factor is purely feeling wise, as not everyone is comfortable going too fast and the driver could lose control of the vehicle very quickly. At the same time, cornering at higher speeds is also uncomfortable.

The algorithm for limiting the maximum permitted speed works on the principle that the vehicle speed is monitored from the encoders and when the maximum set speed is reached, the PI controller is activated, which shifts the sliding plane of the Sliding mode controller in the axis of the desired inclination and thus the vehicle begins to smoothly slow down to the desired speed and at the same time turns the rider (platform) to an upright position. The goal of this controller is of course to limit the maximum speed of the vehicle, but at the same time we don't want it to twitch unpleasantly with the rider.

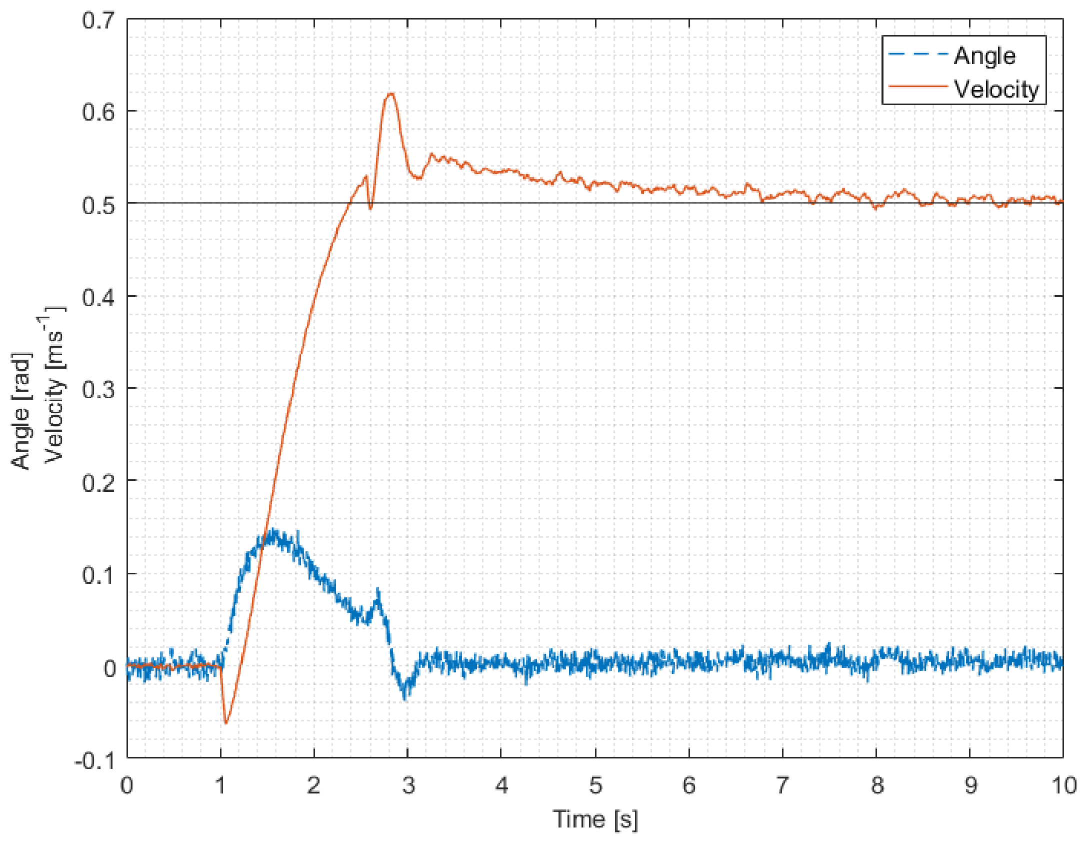


Figure 4.25: Maximum velocity limit

As we can see on the graph (Fig. 4.25) in time approximately 2.5s the maximum speed allowed was reached and the controller was switched on, which moves the sliding mode controller sliding plane. The sharp increase in speed after reaching the maximum allowed can be interpreted as accelerating the vehicle to make the rider's roll lower so that the vehicle can naturally decelerate.

Battery status monitoring

Another algorithm is battery monitoring. If the battery voltage drops below a certain critical level during operation, the algorithm changes the desired tilt from the equilibrium position to a sinusoidal waveform of a "higher" frequency, which significantly reduces the user's driving comfort and prompts the user to exit the vehicle and charge the batteries.

Presence of a rider on the vehicle

No algorithm was created to detect the presence of the rider on the vehicle, but existing buttons on the vehicle handlebars were used. The buttons are directly connected to the ESCON 50/5 motor control unit on the Enable pin and thus directly activate the motor control when they are pressed. There are two of these buttons on the handlebars, one of which is a lock button and the other is not, so if we want the vehicle to run without a rider, the lock button is pressed and the other switch needs to be held down for the entire ride when running with a rider. Of course, there is nothing to prevent the rider from just latching the first button while riding, but that is up to the discretion of each user.

Maximum platform rotation

The reason to address the maximum platform roll is more aimed at the vehicle's crash condition, since during actual operation the controller should be hard enough and at the same time with the maximum forward speed algorithm the user should not get into such tilts. Additionally, when the rider falls off the vehicle, the motors should automatically shut down, as the rider is required to hold the button that turns on the drivers while riding, as already mentioned. However, if the vehicle were to fall without a rider, or if the rider had the arrest button depressed while riding, the motors should be shut off to prevent damage to the vehicle or its surroundings. The roll value was empirically determined to be $\pm 30°$, since the maximum roll range of the vehicle is approximately 85°. When the maximum pitch is reached, the controller sets the value of the duty $s = 0$ and thus the drivers enter the Error state and disconnect the motors from the power supply.

4.4 Architecture of the entire system

In the "State observer" section, we concluded that for good observability of the system, we need to know the wheel speed and angular velocity of the platform. From the search section "Current state of the vehicle" we know that there are HEDM5500 encoders to measure the wheel positions of the vehicle and there is an MPU6050 on the vehicle to get the angular velocity. In the previous thesis, the use of these sensors was sufficient and therefore we will not assume their replacement.

As part of the system improvements, the original lead-acid batteries installed in the unstable balancing vehicle were replaced with lithium-ion (Li-Ion) batteries (Fig. 4.26). The primary motivation for this change was to enhance the practicality and maintenance of the vehicle during frequent laboratory experiments.

Li-Ion batteries offer several advantages over traditional lead-acid technology. They are significantly lighter and more compact, which makes their removal and reinstallation into the vehicle much easier and more efficient. Furthermore, lithium-ion batteries are better suited for cyclic charging and discharging, a behavior typical during repeated testing and development cycles. This characteristic ensures longer battery life and more stable performance over time.

Figure 4.26: Li-Ion battery installed in the balancing vehicle [34]

Additionally, Li-Ion batteries provide a higher energy density and improved charging efficiency, which contribute to more consistent system behavior and shorter downtime between tests. Overall, the battery replacement was an important step towards making the experimental platform more robust, user-friendly, and sustainable for long-term development [35].

Apart from the replacement of the control unit and the battery system, all other components of the vehicle, such as the sensors, motors, and the overall mechanical structure, were retained from the previous thesis project by Richter [6]. The existing components proved to be fully sufficient for the intended experiments and system development, and therefore, no modifications or replacements were necessary. Their performance met all required criteria, allowing the focus of this work to remain primarily on control system design and testing.

4.4.1 Sensor data processing

Reading values from encoder and solving counter overflow

To calculate the wheel speed, the overFlow Safety function block was created (Fig. 4.27), since the output of the encoder counter from the STM blockset is a UINT16 number and the overflow causes the counter to reset its output. The principle is simple; if the differential is greater/less than 8192, the value 65536 will be added/subtracted.

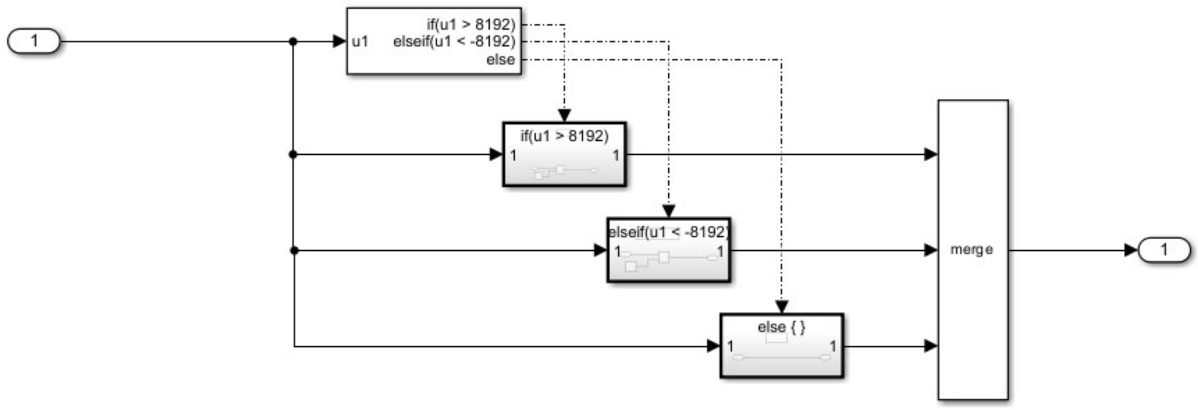


Figure 4.27: OverFlow Safety function block

Tilt calculation from Inertial Measurement Unit

To calculate the rotation of the platform of an unstable balancing vehicle, we must first check the fit of the accelerometer. If the axes of the accelerometer were generally rotated, we could not simply account for them and would have to consider a linear combination of axes. From Richter's thesis [6], we can assume the accelerometer is located in the platform rotation axis or the rotation axis of the motors. Furthermore, it is useful to calculate the maximum range of platform rotation for subsequent use in safety algorithms, where it is assumed that the power supply to the motors will be switched off if a certain value of tilt is exceeded.

The measurements were very easy to make by first turning the vehicle to one side and then to the other and reading the accelerometer. This gave us two directional vectors that point in the direction of gravity, see (4.49 and 4.50).

$$\overrightarrow{g_1} = -0.76\,\overrightarrow{i} + 0.662\,\overrightarrow{j} + 0.076\,\overrightarrow{k} \tag{4.49}$$

$$\overrightarrow{g_2} = -0.705\,\overrightarrow{i} - 0.695\,\overrightarrow{j} + 0.058\,\overrightarrow{k} \tag{4.50}$$

From the scalar product of the vectors (4.51) we then determine the maximum rotation range of the platform of the unstable vehicle.

$$\alpha = \arccos \frac{\overrightarrow{g_1} \cdot \overrightarrow{g_2}}{|\overrightarrow{g_1}||\overrightarrow{g_2}|} = 85°24' \tag{4.51}$$

The maximum rotation range of the platform is therefore approximately 85°. The vector product (4.52) is then used to obtain a vector perpendicular to both direction vectors, which in turn gives the vector around which the unstable vehicle rotates.

$$\vec{o} = det \begin{pmatrix} \vec{i} & \vec{j} & \vec{k} \\ -0.76 & 0.662 & 0.076 \\ -0.705 & -0.695 & 0.058 \end{pmatrix} = 0.0912\,\vec{i} - 0.0095\,\vec{j} + 0.9949\,\vec{k} \tag{4.52}$$

From the calculated axis of rotation we can say that the balancing vehicle tilts almost exactly around the Z axis, and the error caused by not recalculating will be negligible.

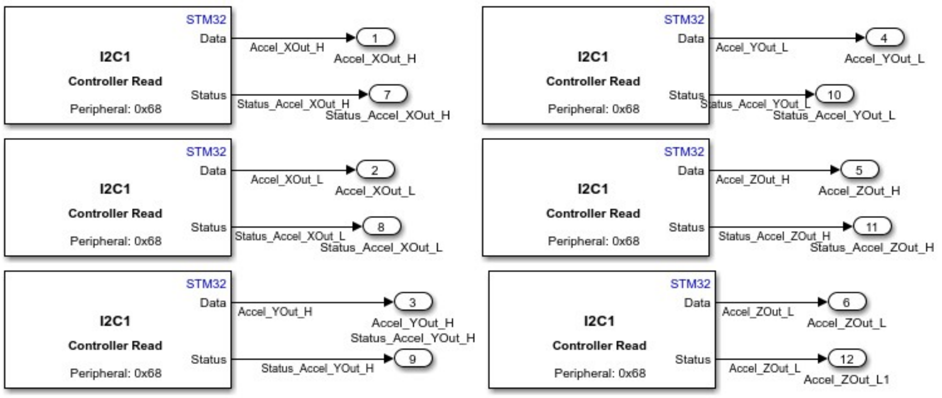


Figure 4.28: Reading I2C data from STM in Simulink environment

Tilt just from Accelerometer

The calculation of the platform rotation for steering an unstable vehicle can be approached in several ways. Using the MPU650, we measure acceleration and angular velocity in three axes. Therefore, we can use only the accelerometer data to calculate the angle.

$$\theta_{accel} = -\arcsin\frac{g_x}{g} \tag{4.53}$$

$$\phi_{accel} = \frac{\pi}{2} - \arccos\frac{g_y}{g\cos\theta_{accel}} \tag{4.54}$$

The advantage of using only the accelerometer to calculate the rotation is long-term temporal stability, since we use the acceleration direction (in steady state only gravitational) written in the accelerometer axes for the calculation. The disadvantage of this method is that the calculated roll value has a lot of noise, basically we do not compensate for the forward acceleration of the vehicle and poor impact readings.

Tilt just from Gyroscope

Another way to obtain the tilt platform is to integrate the angular velocity obtained again from the MPU6050 (Eq.4.55). However, we assume that in steady state the mean value read from the gyro is zero.

$$\Phi_{Gyro} = \int \omega_{gyro} dt \tag{4.55}$$

Due to temperature changes, unstable force effects and noise, the gyroscope has an offset, which we denote as σ. The gyroscope then generates a drift error that increases with time, which can be described as (Eq. 4.56).

$$\Phi_{Gyro} = \int (\omega_{gyro+\sigma}) dt \tag{4.56}$$

The advantage of using only the gyroscope to calculate the platform roll of an unstable vehicle is its insensitivity to vibration (this is based on the fundamental amplitude-frequency characteristic of the integrator and the fact that noise appears as high frequency). However, a major drawback is the aforementioned offset, which even if we calibrate the program at the beginning, after a while the offset value either changes or even the calibration itself is not perfect and after a while the tilt value integrates beyond unacceptable limits.

Complementar filter

A complementary filter was then implemented, combining data from the gyroscope and the accelerometer.

$$\hat{\Phi}_{acc} = \arctan \frac{Accel_z}{Accel_x} \tag{4.57}$$

$$\hat{\Theta}_{acc} = \arcsin Accel_y \tag{4.58}$$

$$\dot{\Theta}_{gyr} = \cos \hat{\Phi} \cdot Gyro_z - \sin \hat{\Phi} \cdot Gyro_x \tag{4.59}$$

$$\dot{\Phi}_{gyr} = [(\cos \hat{\Phi} \cdot Gyro_x - \sin \hat{\Phi} \cdot Gyro_z) \tan \hat{\Theta}] + Gyro_y \tag{4.60}$$

$$\hat{\Theta}_{n+1} = [\hat{\Theta}_n + \dot{\Theta}_{gyr} \cdot T_s] \cdot (1 - \alpha) + \hat{\Theta} \cdot \alpha \tag{4.61}$$

$$\hat{\Phi}_{n+1} = [\hat{\Phi}_n + \dot{\Phi}_{gyr} \cdot T_s] \cdot (1 - \alpha) + \hat{\Phi} \cdot \alpha \tag{4.62}$$

Kalman filter on IMU

A part of the implementation was based on the work presented in [36], which focused on the development of a low-cost IMU using sensor fusion for attitude estimation. The

Kalman filter algorithm described in the article was replicated and adapted for use in this project.

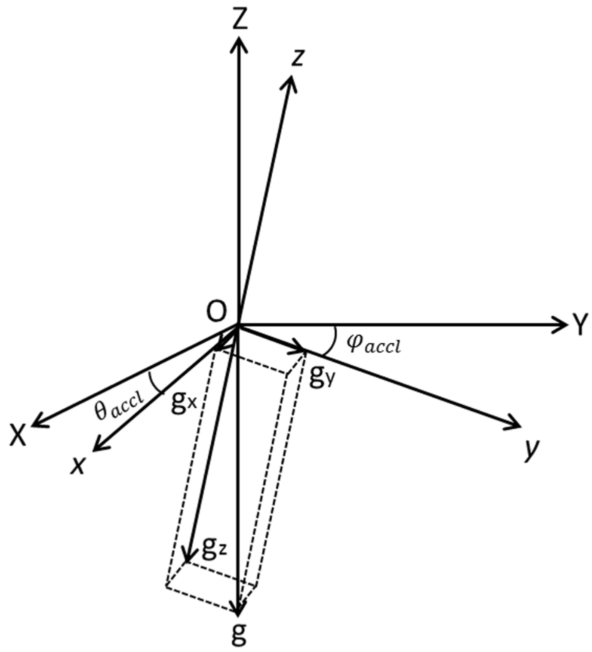


Figure 4.29: Attitude angle relation of the accelerometer in coordinate [36]

The Kalman filter is optimal if process noise and measurement noise can be modeled by white Gaussian noise. As already mentioned, the relationship between tilt and angular velocity is via the derivative. We can use the actual tilt ∂ to develop the equation of state and the measurement equation.

$$\begin{bmatrix}\dot{\partial}\\ \dot{b}\end{bmatrix} = \begin{bmatrix}0 & -1\\ 0 & 0\end{bmatrix}\begin{bmatrix}\partial\\ b\end{bmatrix} + \begin{bmatrix}1\\ 0\end{bmatrix}\omega_{gyro} + \begin{bmatrix}\sigma_{gyro}\\ 0\end{bmatrix} \tag{4.63}$$

$$\partial_{accel} = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}\partial\\ b\end{bmatrix} + \sigma_{accel} \tag{4.64}$$

where ω_{gyro} is the angular velocity with bias, ∂_{accel} is the angle calculated from the accelerometer. σ_{gyro} is the noise from the gyro measurement, σ_{accel} is the noise from the accelerometer measurement, and b is the drift error of the gyro. T_s is the sampling period. We then convert the equation of state into a discrete system.

$$\mathbf{X(k|k-1) = AX(k-1|k-1) + BU(k)} \tag{4.65}$$

Where A is the transition matrix $A = \begin{bmatrix}1 & Ts\\ 0 & 1\end{bmatrix}$, B is System control matrix $B = \begin{bmatrix}Ts\\ 0\end{bmatrix}$

$X(k|k-1)$ is the state of the system at time k estimated from states k-1. $U(k)$ is the exogenous control input at time k. $P(k|k-1)$ is the covariance of the error estimate $X(k|k-1)$

$$\mathbf{P(k|k-1) = AP(k-1|k-1)A^T + Q} \tag{4.66}$$

Where Q is the covariance matrix of the system process noise $Q = \begin{bmatrix} q_{accel} & 0 \\ 0 & q_{gyro} \end{bmatrix}$, where q_{accel} is the accelerometer variance and q_{gyro} is the gyroscope variance. A^T is the transpose of the matrix A.

We obtain the optimal estimate of $X(k|k)$ in state k by using.

$$\mathbf{X(k|k) = X(k|k-1) + k_g(k) \cdot (Z(k) - HK(k|k-1))} \tag{4.67}$$

Where H is observation matrix, $H = \begin{bmatrix} 1 & 0 \end{bmatrix}$. k_g is the Kalman gain derived from minimizing the covariance of the posterior error by

$$\mathbf{k_g = P(k|k-1)H^T/(H \cdot P(k|k-1)H^T + R)} \tag{4.68}$$

Where R is the covariance matrix of the accelerometer measurement noise. In order to update the Kalman filter, we need to update the covariance equation using

$$\mathbf{P(k|k) = (I - k_g(k) \cdot H)P(k|k-1)} \tag{4.69}$$

Where I is the unit matrix $I = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

Tilt calculation comparison Acc, Gyr, Kalman

On the Fig. 4.30 we can see the comparison of the MPU6050 sensor rotation calculation calculated only from accelerometer values using the equation (Eq. 4.53), only integrating the gyroscope value (Eq. 4.55) and applying the Kalman filter. The measurement was run for approximately $20s$ and we can see that the steady state tilt value calculated by gyroscope integration is different from the value obtained from the accelerometer and Kalman filter. Thus, we can say that when using the Kalman filter or the accelerometer, we do not care about the initial rotation of the platform. At the same time, we can see that the Kalman filter behaves the same as the value calculated from the gyroscope in terms of dynamics and effectively filters out the accelerometer noise.

On the Fig. 4.31 we can see the long-term measurement that was run for $20min$ to verify the temporal stability of the tilt calculation. For this measurement, the measurement unit was left at rest without any external influence. The gyro offset was calibrated to zero before the measurement started. As can be seen, the angle calculated from the gyroscope alone drifted by approximately $1rad$ after $20min$ of measurement, while the angle calculated from the accelerometer and Kalman filter remained stationary for a long time. With this measurement, we verified that using only the gyroscope to calculate platform rotation is indeed possible, but only for short-term applications. Thus, the Kalman filter was used in a real device to obtain the platform rotation values.

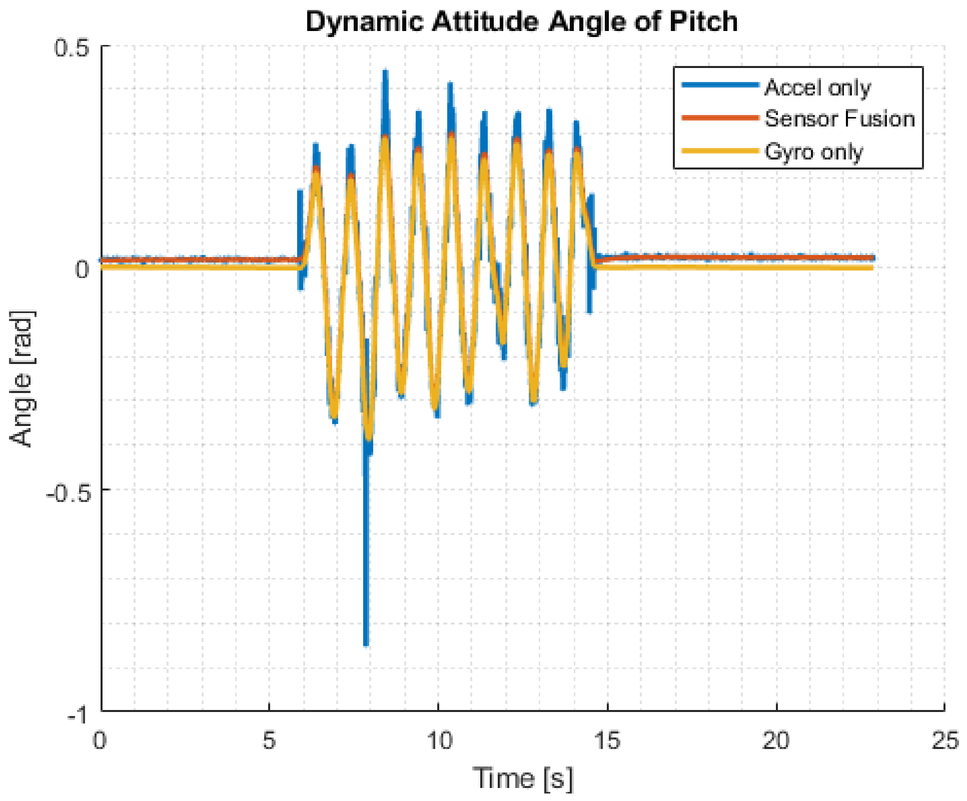


Figure 4.30: Angle of pitch

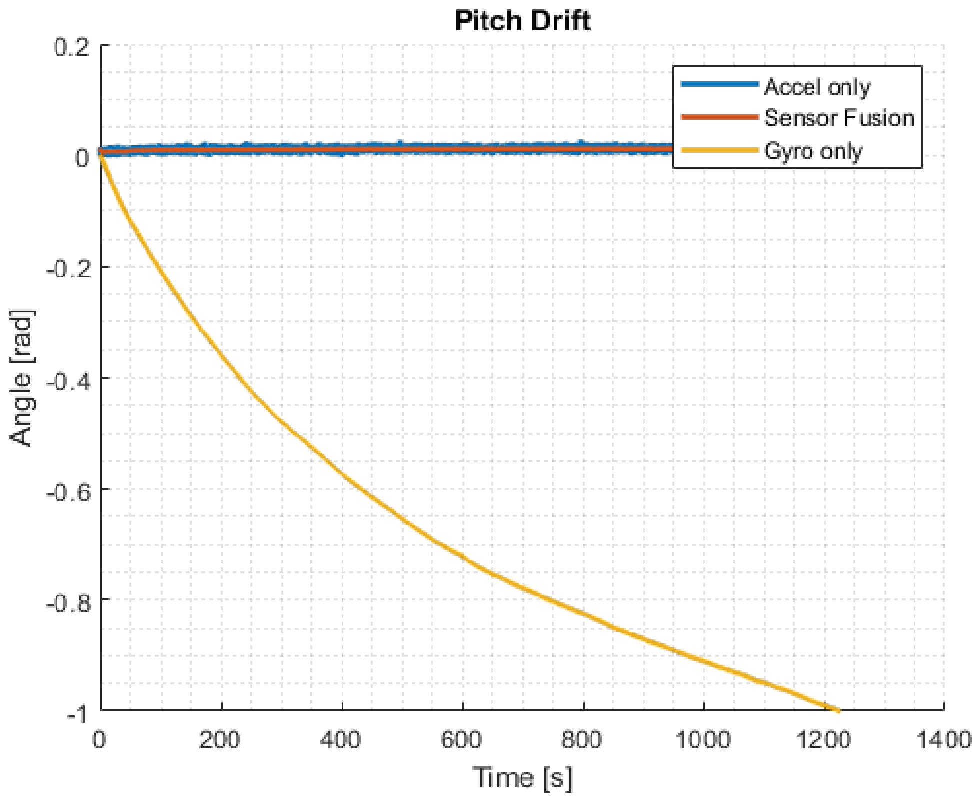


Figure 4.31: Pitch drift

4.5 Design and manufacture of the control unit

When selecting a microcontroller for the new control unit, it was necessary to pay attention to the fact that the microcontroller was powerful enough, had enough peripherals, and it was possible to generate code from the Simulink environment. The selection of a suitable microcontroller has already been part of Richter's thesis [6] and thanks to the availability of the chip itself in the mechatronics laboratory, it was possible to test the functionality directly.

The new controller is therefore based on the Nucleo-F746ZG (Fig. 4.32) and the procedure for working with this controller when generating code from Simulink is suitably described on the MathWorks website [37],[38]. In short, we set up in Simulink which chip we will work with, then in STM32CubeMX we set up the necessary pins and peripherals to use and then we can work in Simulink in the usual way.

To connect the new control unit to the vehicle, a very simple interconnect PCB was created that only had connectors on it, so it was just a matter of connecting the sensors and control signals to the control unit.

During the development of the control software, individual function blocks were tested using both SIL (Software In the Loop) and PIL (Processor In the Loop) . These tests were used to test that the generated C code achieved the required accuracy and also that the control unit itself would be able to achieve the required results in "real" time. The SIL and PIL testing process itself was very straightforward, as it is again well documented on the MathWorks website [39], and all of the results were satisfactory, with the only differences between simulation and SIL/PIL testing being in the accuracy of the last bits of our chosen data type. We could therefore declare that the microcontroller would be suitable for driving the vehicle itself.

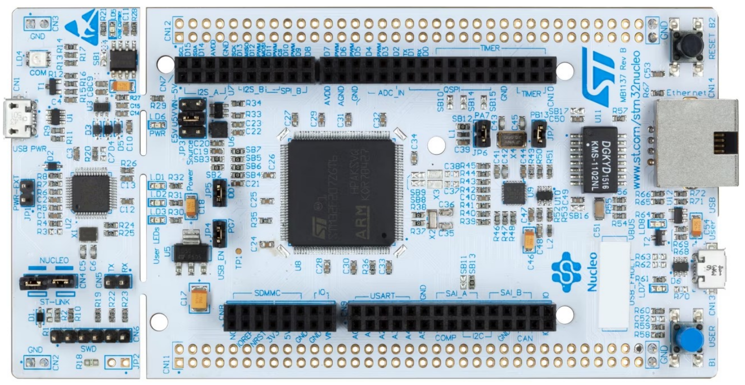


Figure 4.32: New control unit - Nucleo-F746ZG [40]

4.6 Hardware In the Loop testing

Hardware In the Loop is an important part of the development of mechatronic devices within Model Based Design. It involves testing an already developed controller on the simulated behaviour of a dynamic system, in our case an unstable vehicle. The advantage of testing the controller on a simulated environment is the repeatability of the experiments.

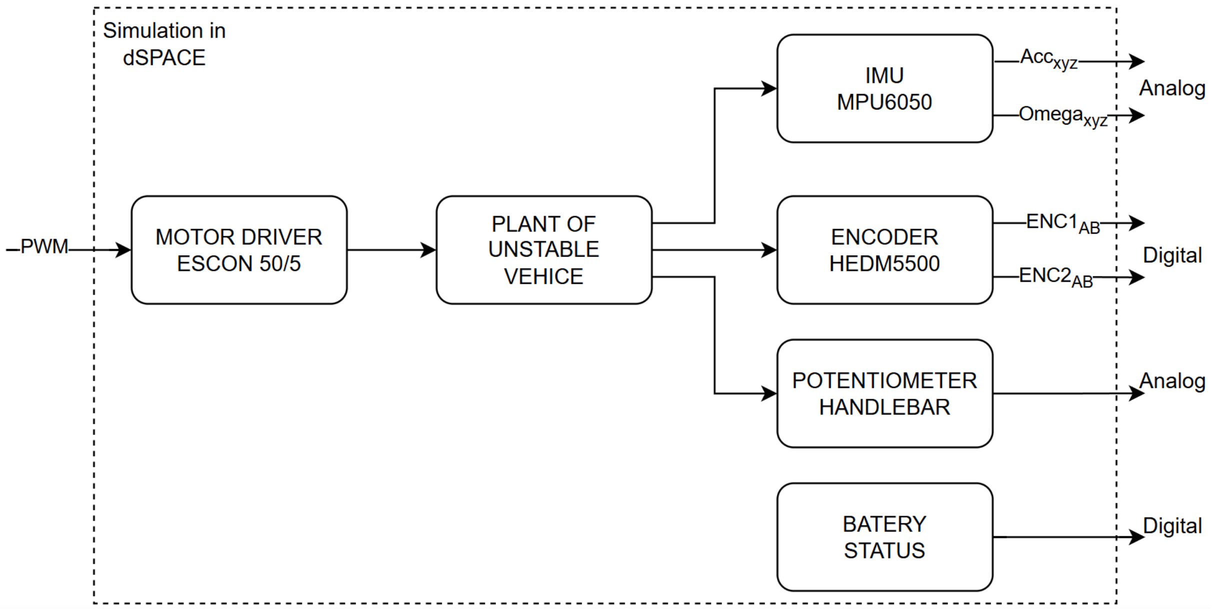


Figure 4.33: Block diagram of HIL implementation in dSPACE

In the previous part of this thesis, we have created a mathematical model of an unstable vehicle and verified its behaviour with a comparison to a real system. Thus, for HIL testing we will use the model created by the black-box method, since we obtained satisfactory results for it. Another reason for not using the unstable vehicle model modeled in Simscape, even though it achieved equally satisfactory results as the black-box model and also has a better visualization of the actual system behavior, is computational complexity. This is because the Simscape model consists of a system of differential algebraic equations, which are more computationally intensive than a system of linear differential equations. For use in HIL testing, our primary concern is also that the system runs in real time and the controller itself does not recognize that it is a simulation.

In different articles, HIL testing means different things. Some include only the environment in the simulation, sometimes known as mHIL (Mechanics hardware in the loop), others include sensors, mechanics of the environment, with the power adaptation and actuators being real, sometimes known as pHIL (power hardware in the loop). Other terms are sHIL (signal hardware in the loop), cHIL (controller hardware in the loop) and others [41],[42], [43]. In our case, we will deal with testing, which is most commonly referred to as sHIL. The (Fig. 4.33) shows what all will be implemented in the simulation. All the power component will be simulated, so for our purposes there will be no need to create artificial loads, either mechanical or electrical, and testing will remain at the signal level only.

Conversion of position signal to encoder pulses

For HIL testing the HEDM5500 encoder (Fig.4.34), we needed to know the motor gear ratio (1:9) and the encoder resolution (1000 pulses per revolution). The output of the simulation is in radians. Emulating the encoder was done in the following steps.
•Convert radians to revolutions
•Conversion via gear ratio to tick count
•Shift one signal by a quarter phase
•Compare against half

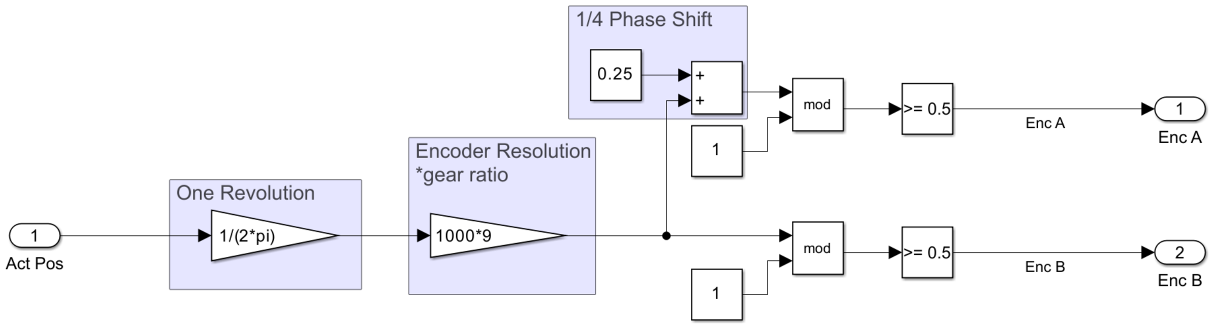


Figure 4.34: Emulation of an encoder signal

Implementation of an I2C Bridge for IMU Simulation

During the HIL testing of the control unit for the self-balancing vehicle, it was necessary to simulate the behavior of an inertial measurement unit (IMU), specifically one that communicates via the I2C protocol. However, the SCALEXIO system from dSPACE, available in the laboratory, does not natively support I2C communication. The official solutions offered by dSPACE, such as the SCALEXIO Serial Interface Solution [44] or the XSG SPI/I2C FPGA library [45], are associated with significant additional costs.

To overcome this limitation, several potential approaches were considered:

- Emulating the I2C protocol directly in the simulation via a state machine.
- Implementing I2C communication in an FPGA using VHDL.
- Replacing the I2C-based IMU with another sensor using a supported communication interface.
- Developing an external I2C bridge using a secondary microcontroller.

The most practical and cost-effective solution was to implement a standalone microcontroller that acts as a communication bridge between the dSPACE system and the control unit, shown at Fig. 4.35. This microcontroller emulates the IMU device and communicates with the control unit using the I2C protocol, while receiving simulated sensor data from dSPACE via a compatible interface (ADC).

The bridge is configured to respond to the same I2C address as the real MPU-6050 sensor. Data transmission is handled in burst read mode Fig.(4.37), which allows the control unit to access all required values—three-axis acceleration, three-axis angular velocity,

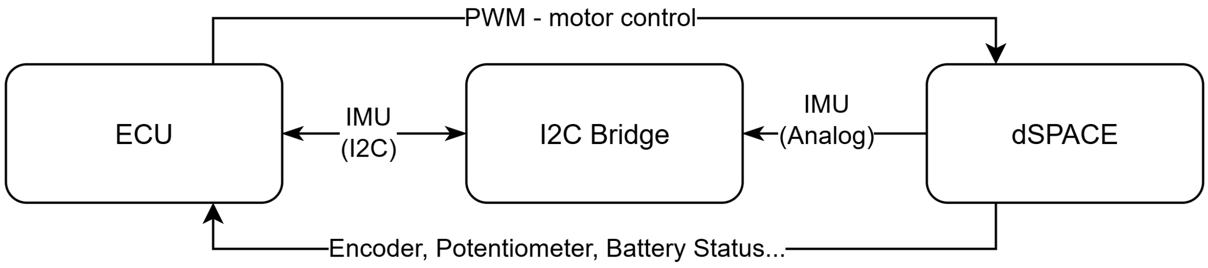


Figure 4.35: I2C bridge between ECU and dSPACE

and internal temperature—in a single, efficient transaction compared to Single-Byte read sequence (Fig. 4.36). This approach preserves the original communication structure and minimizes modifications to the control firmware. Abbreviations in Fig.4.36 and Fig.4.37 are described in Tab. 4.2

Master	S	AD+W		RA		S	AD+R			NACK	P
Slave			ACK		ACK			ACK	DATA		

Figure 4.36: Single-Byte read sequence

Master	S	AD+W		RA		S	AD+R			ACK		NACK	P
Slave			ACK		ACK			ACK	DATA		DATA		

Figure 4.37: Burst read sequence

Signal	Description
S	Start Condition: SDA goes from high to low while SCL is high
AD	Slave I^2C address
W	Write bit (0)
R	Read bit (1)
ACK	Acknowledge: SDA line is low while the SCL line is high at the 9^{th} clock cycle
NACK	Not-Acknowledge: SDA line stays high at the 9^{th} clock cycle
RA	MPU-60X0 internal register address
DATA	Transmit or received data
P	Stop condition: SDA going from low to high while SCL is high

Table 4.2: I^2C Signal Descriptions

Tilt to acceleration conversion

The final thesis of Štěpánek [1] showed that neglecting the effect of forward acceleration on the calculation of the rotational base commits only a minimal error. Therefore, to obtain the x-axis and y-axis acceleration values emulated by the MPU6050, we use only the value of the tilt. Calculation of the static acceleration value.

$$a_x = g \sin \varphi \tag{4.70}$$

$$a_y = -g \cos \varphi \tag{4.71}$$

4.7 Real vehicle testing

As we can see from the graph 4.38 the HIL simulation faithfully represents the dynamic behavior of the real system and can therefore be used to test different driving scenarios.

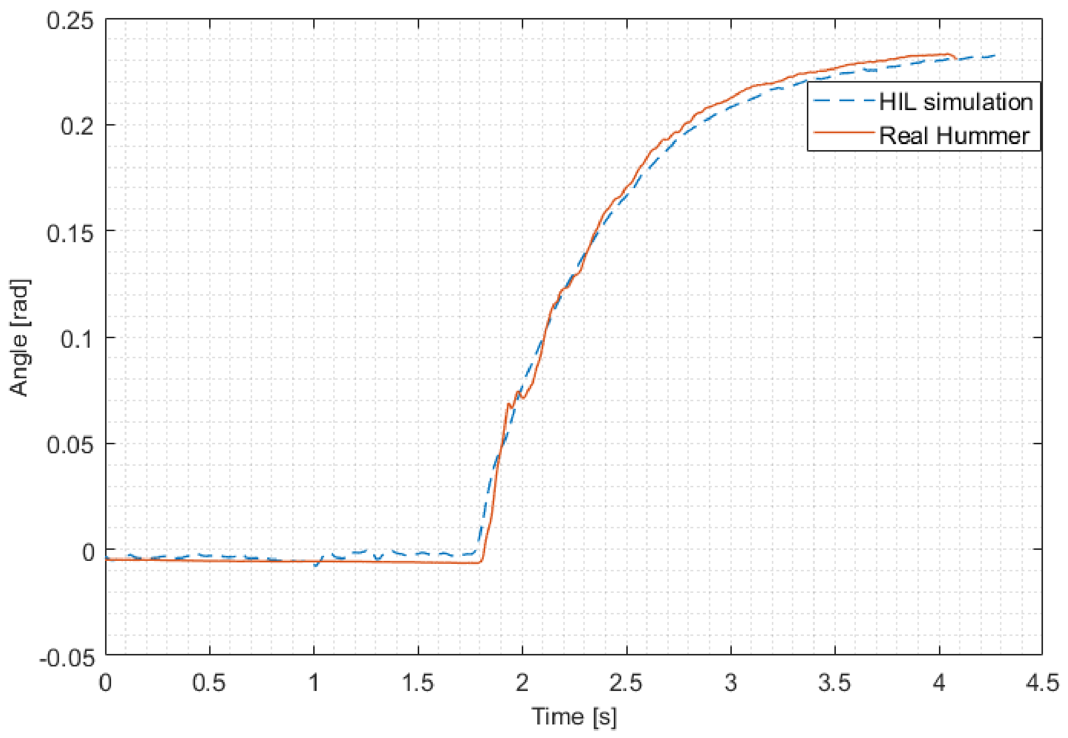


Figure 4.38: Comparison between HIL simulation and real system response

The control unit for the unstable balancing vehicle was programmed (debugged) based on HIL simulation. The real system subsequently behaved almost identically to the simulated system, and therefore its response was predictable.

Driverless vehicle operation:

As far as the driverless vehicle behavior is concerned, all predetermined requirements were met. In steady state and without external forces, the vehicle balances in place without drifting away. When an external force is applied, the vehicle moves in the direction of the applied force but still maintains an upright position. In accomplishing this goal, there was no need to iterate the process many times as the system parameters do not change with the passage of time and, most importantly, there is no human factor as a feel for the controller's hardness.

Operating a vehicle with a rider

All the requirements for the rider control system have been met. The tuning of the rider stabilization controller cannot be so easily set and unsimulated as there is another factor and that is how the user feels when riding. During this process, several types of controllers with different parameters were tested. If the controller is too soft, it may feel to the user as if it is "floating" on the vehicle; however, a too hard controller was also not comfortable for most. The goal was to find a "universal" controller that would be robust enough to accommodate a vehicle with riders of different parameters (height, weight). In the end, SMC was used to try to get the system to behave like a first-order system at the output. At the same time, it is robust enough to disturbances and to model imperfections. The rider parameters are not identified and the same controller is used for both the stabilized with rider and the stabilized without rider.

5 Conclusion

The aim of this master's thesis was to design, implement, and verify a control system for an unstable self-balancing vehicle using the Model-Based Design (MBD) methodology. The entire development process – from evaluating the current condition of the vehicle, through modeling, simulation, and control design, to hardware implementation and real-world testing – was successfully completed in accordance with the V-model development structure.

Three modeling approaches were applied – white-box, grey-box, and black-box – to better understand the vehicle's dynamic behavior and to build an accurate simulation model. A physical model was developed in Simscape Multibody. Black-box identification was implemented in the form of a state-space model. Its dynamic behaviour was compared with the behaviour of the model created in the Simscape Multibody environment. Both approaches provided comparable results and faithfully represented the real system. The main advantage of the Simscape model was the possibility of 3D visualization of the system motions, but its disadvantage was its significant computational complexity. For this reason, the black-box model was preferred in the rest of the work.

Several control strategies were evaluated, including PID, Sliding Mode Control, Linear Quadratic Regulator, and Model Predictive Control. Each algorithm was validated in simulation, and all were capable of stabilizing the vehicle under ideal linear conditions. Based on robustness and ride comfort, the SMC algorithm was selected for real-world implementation due to its exponential convergence behavior and resilience to system uncertainties.

Based on the Gramian analysis, it was verified that the system is observable using sensors already installed on the unstable vehicle. Measuring the wheel speed and the angular velocity of the rotation of the stationary platform was found to be the most suitable to ensure observability. The existing sensors and actuators were thus fully sufficient for the purpose and did not need to be replaced. A state observer (Kalman filter) was designed to estimate internal system states using only measurable signals. The observer's performance was validated in simulation, showing accurate state reconstruction even in the presence of noise and disturbances.

The orientation of the MPU6050 sensor mounted on an unstable vehicle was determined in order to correctly assess the tilt of the platform. It turned out that the IMU rotates almost exclusively around the Z-axis, and thus neglecting to recalculate accurately does not cause a significant error. Subsequently, different methods of calculating the roll were presented - based on accelerometer data, gyroscope data and a combination of both using a Kalman filter. The tilt calculation from the accelerometer showed long-term stability, but was highly sensitive to noise and vibration. In contrast, the gyroscope, although accurate after initial calibration, generated cumulative error beyond acceptable limits during long-term operation. The Kalman filter combined the advantages of both approaches - providing long-term stability of the calculation while being robust to vibration. Therefore, it was ultimately chosen as the method for determining roll on an actual vehicle.

To ensure operational safety, dedicated algorithms were implemented to monitor maximum speed, battery voltage, tilt angles, and other critical conditions. As part of this work, the original lead-acid batteries were replaced with lithium-ion batteries. The main reasons for this replacement were easier handling during battery removal and charging, as well as the better suitability of Li-Ion technology for the regular cyclic operation required in the application. A new control unit was developed with sufficient computational performance and support for automatic code generation from Simulink. Due to the lack of an I2C peripheral on the dSPACE platform used in the mechatronics laboratory, a secondary microcontroller was designed and programmed to serve as a communication bridge between the dSPACE and the controller. This microcontroller was responsible for reading analogue signals from the dSPACE and converting them into I2C communication towards the control unit. In this way, the problem of missing peripherals was solved efficiently, simply, and at low cost. The system's functionality was validated through Hardware-in-the-Loop (HIL) testing using the dSPACE platform, significantly reducing the risk associated with testing on real hardware.

The thesis demonstrates the practical benefits of applying Model-Based Design to the development of complex, safety-critical mechatronic systems. The structured use of the V-model and continuous validation at each development stage contributed to the creation of a robust and adaptable solution, laying the foundation for future upgrades of the vehicle and serving as a reference methodology for similar engineering projects.

References

[1] ŠTĚPÁNEK, J. *Identifikace systému, senzorika a implementace řídicího algoritmu pro nestabilní balancující vozidlo*. Brno, 2011. Available also from: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=42315. MA thesis. Brno University of Technology, Faculty of Mechanical Engineering.

[2] ZOUHAR, F. *Návrh konstrukce, řízení a elektroniky pro nestabilní balancující vozidlo*. Brno, 2011. Available also from: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=40310. MA thesis. Brno University of Technology.

[3] HORÁK, P. *Řízení laboratorního modelu nestabilního balancujícího vozidla*. Brno, 2011. Available also from: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=42317. MA thesis. Brno University of Technology, Faculty of Mechanical Engineering.

[4] DOBOSSY, B. *Segway driver parameter estimation and its use for optimizing the control algorithm*. Brno, 2019. Available also from: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=193560. MA thesis. Brno University of Technology, Faculty of Mechanical Engineering.

[5] MATĚJÁSKO, M. *Design of a Fault-Tolerant Control System for a Self-Balancing Two-Wheel Vehicle*. Brno, 2015. Available also from: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=106374. MA thesis. Brno University of Technology, Faculty of Mechanical Engineering.

[6] RICHTER, O. *Návrh a realizace inovovaného systému řízení pro nestabilní balancující vozidlo*. Brno, 2023. Available also from: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=258471. Bachelor's thesis. Brno University of Technology, Faculty of Mechanical Engineering.

[7] SOLNICKÝ, V. *Doplňující sensorika pro nestabilní balancující vozidlo*. Brno, 2012. Available also from: https://www.vut.cz/www_base/zav_prace_soubor_

verejne.php?file_id=54089. Bachelor's thesis. Brno University of Technology, Faculty of Mechanical Engineering.

[8] BASTL, M. *Návrh elektroniky pro řízení dvoukolového nestabilního vozidla*. Brno, 2015. Available also from: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=105154. MA thesis. Brno University of Technology, Faculty of Mechanical Engineering.

[9] BEZDÍČEK, Milan; GREPL, Robert. *Mechatronika - vybrané problémy*. Vyd. 1. Brno: Tribun, 2008. ISBN 978-80-214-3804-0. OCLC: 403636476.

[10] ELLIS, G. *Control System Design Guide - Using Your Computer to Understand and Diagnose Feedback Controllers (4th Edition)*. Elsevier, 2012. ISBN 978-0-12-385920-4. Available also from: https://app.knovel.com/hotlink/toc/id:kpCSDGUYC7/control-system-design/control-system-design.

[11] ISERMANN, Rolf. Mechatronic systems—Innovative products with embedded control. *Control Engineering Practice* [online]. 2008, vol. 16, no. 1, pp. 14–29 [visited on 2025-04-09]. ISSN 09670661. Available from DOI: 10.1016/j.conengprac.2007.03.010.

[12] *SCALEXIO LabBox — dspace.com* [https://www.dspace.com/en/pub/home/products/hw/simulator_hardware/scalexio/scalexio_labbox.cfm]. [N.d.]. [Accessed 21-04-2025].

[13] ISERMANN, Rolf. *Mechatronics systems: fundamentals*. Londres: Springer, 2005. ISBN 978-1-85233-930-2.

[14] *Brushed DC motor GR80x80* [https://ametekcdn.azureedge.net/mediafiles/project/oneweb/oneweb/dunkermotoren/downloads/pdf/products/002_brushed%20dc%20motors/011_gr_80x80_en_1.pdf?revision:8aac8394-8755-4687-bdcd-3fa7273509e4]. [N.d.]. [Accessed 29-04-2025].

[15] *Potencjometr do sterownika UNO (25k) niebieski — sklep.eur-med.pl* [https://sklep.eur-med.pl/glowna/687-potencjometr-do-sterownika-uno-25k-niebieski]. [N.d.]. [Accessed 29-04-2025].

[16] LASKAKIT. *3-osý gyroskop a akcelerometr GY-521, MPU6050 — LaskaKit — laskakit.cz* [https://www.laskakit.cz/arduino-gyroskop-a-akcelerometr-gy-521--mpu6050/]. [N.d.]. [Accessed 29-04-2025].

[17] *analog.com* [https://www.analog.com/media/en/technical-documentation/data-sheets/ADuM1400_1401_1402.pdf]. [N.d.]. [Accessed 29-04-2025].

[18] *meanwell.com* [https://www.meanwell.com/Upload/PDF/SD-15/SD-15-SPEC.PDF]. [N.d.]. [Accessed 29-04-2025].

[19] *Avago: Optical Kit Encoder (HEDS-5500/5540, HEDS-5600/5640, and HEDM-5500/5600) — e-motionsupply.com* [https://www.e-motionsupply.com/product_p/heds-hedm.htm]. [N.d.]. [Accessed 29-04-2025].

[20] *maxongroup.com* [https://www.maxongroup.com/medias/sys_master/root/8834332262430/409510-ESCON-50-5-Hardware-Reference-En.pdf]. [N.d.]. [Accessed 29-04-2025].

[21] ANTSAKLIS, Panos J.; MICHEL, Anthony N. *A Linear Systems Primer*. Boston, MA: Birkhäuser Boston, 2007. A linear systems primer. ISBN 978-0-8176-4661-5.

[22] DE SOUZA, Marcus Romano Salles Bernardes; MUROFUSHI, Rodrigo Hiroshi; TAVARES, José Jean-Paul Zanlucchi De Souza; RIBEIRO, José Francisco. Comparison Among Experimental PID Auto Tuning Methods for a Self-balancing Robot. In: SANTOS OSÓRIO, Fernando; SALES GONÇALVES, Rogério (eds.). *Robotics* [online]. Cham: Springer International Publishing, 2016, vol. 619, pp. 72–86 [visited on 2025-04-09]. ISBN 978-3-319-47247-8. Available from DOI: 10.1007/978-3-319-47247-8_5. Series Title: Communications in Computer and Information Science.

[23] KNOLL, Carsten; ROBENACK, Klaus; DARIIMAA, Bolorkhuu. Holonomic modeling and hierarchic tracking control of an unstable underactuated nonholonomic system. In: *2016 13th International Multi-Conference on Systems, Signals & Devices (SSD)* [online]. Leipzig: IEEE, 2016, pp. 102–107 [visited on 2025-04-09]. ISBN 978-1-5090-1291-6. Available from DOI: 10.1109/SSD.2016.7473730.

[24] LE, Hoai Nam; PHAM, Minh-Khoi; PHAM, Dinh-Hieu; NGUYEN, Thi-Van-Anh. Trajectory tracking and stabilization of two-wheeled balancing mobile robot with hierarchical and sliding mode control. *International Journal of Dynamics and Con-*

trol [online]. 2025, vol. 13, no. 1, p. 10 [visited on 2025-04-09]. ISSN 2195-268X, ISSN 2195-2698. Available from DOI: `10.1007/s40435-024-01518-0`.

[25] PRABHAKAR, G.; SELVAPERUMAL, S.; PUGAZHENTHI, P. Nedumal; UMAMAHESWARI, K.; ELAMURUGAN, P. Online optimization based model predictive control on two wheel Segway system. *Materials Today: Proceedings* [online]. 2020, vol. 33, pp. 3846–3853 [visited on 2025-04-09]. ISSN 22147853. Available from DOI: `10.1016/j.matpr.2020.06.227`.

[26] KILIAN, F. Johannes; GATTRINGER, Hubert; BREMER, Hartmut. Modeling and Quasi-Static Trajectory Control of a Self-Balancing Two-Wheeled Vehicle. *PAMM* [online]. 2012, vol. 12, no. 1, pp. 75–76 [visited on 2025-04-09]. ISSN 1617-7061, ISSN 1617-7061. Available from DOI: `10.1002/pamm.201210029`.

[27] YUE, Ming; WANG, Shuang; SUN, Jian-Zhong. Simultaneous balancing and trajectory tracking control for two-wheeled inverted pendulum vehicles: A composite control approach. *Neurocomputing* [online]. 2016, vol. 191, pp. 44–54 [visited on 2025-04-09]. ISSN 09252312. Available from DOI: `10.1016/j.neucom.2016.01.008`.

[28] LIU, Yunping; HUANG, Xijie; WANG, Tianmiao; ZHANG, Yonghong; LI, Xianying. Nonlinear dynamics modeling and simulation of two-wheeled self-balancing vehicle. *International Journal of Advanced Robotic Systems* [online]. 2016, vol. 13, no. 6, p. 1729881416673725 [visited on 2025-04-09]. ISSN 1729-8806, ISSN 1729-8814. Available from DOI: `10.1177/1729881416673725`.

[29] PUROHIT, Palak; MODI, Poojan; VYAS, Udit. *Kinematic Control of 2-wheeled Segway* [online]. arXiv, 2021 [visited on 2025-04-09]. Available from DOI: `10.48550/ARXIV.2109.11919`. Version Number: 1.

[30] SEUL JUNG; SUNG SU KIM. Control Experiment of a Wheel-Driven Mobile Inverted Pendulum Using Neural Network. *IEEE Transactions on Control Systems Technology* [online]. 2008, vol. 16, no. 2, pp. 297–303 [visited on 2025-04-09]. ISSN 1063-6536, ISSN 1558-0865. Available from DOI: `10.1109/TCST.2007.903396`.

[31] HOSSAIN, Md. Arafat; ISLAM, Md. Rashidul; HOSSAIN, Md. Alamgir; HOSSAIN, M.J. Control strategy review for hydrogen-renewable energy power system. *Journal*

of Energy Storage [online]. 2023, vol. 72, p. 108170 [visited on 2025-04-28]. ISSN 2352152X. Available from DOI: 10.1016/j.est.2023.108170.

[32] CHEN, Chi-Tsong (ed.). *Linear system theory and design*. 3rd ed. New York: Oxford University Press, 1999. The Oxford series in electrical and computer engineering. ISBN 978-0-19-511777-6.

[33] GAWRONSKI, Wodek; JUANG, Jer-Nan. Model reduction in limited time and frequency intervals. *International Journal of Systems Science* [online]. 1990, vol. 21, no. 2, pp. 349–376 [visited on 2025-04-25]. ISSN 0020-7721, ISSN 1464-5319. Available from DOI: 10.1080/00207729008910366.

[34] *PARKSIDE® Akumulátor 4 Ah PAP 20 B3 — parkside-diy.com* [https://parkside-diy.com/cz/p/100370678/parkside-akumulator-4-ah-pap-20-b3]. [N.d.]. [Accessed 28-04-2025].

[35] KESHAN, H.; THORNBURG, J.; USTUN, T.S. Comparison of lead-acid and lithium ion batteries for stationary storage in off-grid energy systems. In: *4th IET Clean Energy and Technology Conference (CEAT 2016)* [online]. Kuala Lumpur, Malaysia: Institution of Engineering and Technology, 2016, 30 (7 .)–30 (7 .) [Visited on 2025-04-29]. ISBN 978-1-78561-238-1. Available from DOI: 10.1049/cp.2016.1287.

[36] LIU, Yufei; NOGUCHI, Noboru; ISHII, Kazunobu. Development of a Low-cost IMU by Using Sensor Fusion for Attitude Angle Estimation. *IFAC Proceedings Volumes* [online]. 2014, vol. 47, no. 3, pp. 4435–4440 [visited on 2025-04-09]. ISSN 14746670. Available from DOI: 10.3182/20140824-6-ZA-1003.00610.

[37] *Getting Started with STMicroelectronics STM32 Processor Based Boards* [https://www.mathworks.com/help/ecoder/stmicroelectronicsstm32f4discovery/ug/Getting-started-stm32cubemx.html]. [N.d.]. [Accessed 04-05-2025].

[38] *Choose a SIL or PIL Approach* [https://www.mathworks.com/help/ecoder/ug/choosing-a-sil-or-pil-approach.html]. [N.d.]. [Accessed 04-05-2025].

[39] *SIL and PIL Simulations* [https://www.mathworks.com/help/ecoder/ug/about-sil-and-pil-simulations.html]. [N.d.]. [Accessed 04-05-2025].

[40] *NUCLEO-F746ZG* [https://www.st.com/en/evaluation-tools/nucleo-f746zg.html]. [N.d.]. [Accessed 04-05-2025].

[41] ALEKSANDROV, Bulgaria; ACAD, Chavdar; RUMENIN, Bulgaria; MAGELE, Christian; STOYANOV; SOTIROVA, Bulgaria; RITCHIE; TOEPFER; BRAUER, Hartmut; HRISTOV, Marin; REPETTO; ANTCHEV, Bulgaria; MIHAILOV, Bulgaria; ROMANSKY, Bulgaria; VASILEV, Bulgaria; TANAKA, Japan; VALCHEV, Ventsislav; SHELYAGIN, Vladimir; ACAD, Ukraine; STOYNOVA, Anna. Review of hardware-in-the-loop -a hundred years progress in the pseudo-real testing. 2019, vol. 54, pp. 70–84.

[42] MIHALIČ, Franc; TRUNTIČ, Mitja; HREN, Alenka. Hardware-in-the-Loop Simulations: A Historical Overview of Engineering Challenges. *Electronics* [online]. 2022, vol. 11, no. 15, p. 2462 [visited on 2025-04-30]. ISSN 2079-9292. Available from DOI: 10.3390/electronics11152462.

[43] BOUSCAYROL, A. Different types of Hardware-In-the-Loop simulation for electric drives. In: *2008 IEEE International Symposium on Industrial Electronics* [online]. Cambridge: IEEE, 2008, pp. 2146–2151 [visited on 2025-04-30]. ISBN 978-1-4244-1665-3. Available from DOI: 10.1109/ISIE.2008.4677304.

[44] *SCALEXIO Serial Interface Solution — dspace.com* [https://www.dspace.com/en/pub/home/products/hw/simulator_hardware/scalexio/scalexio-board-overview/scalexio_serial-interface.cfm]. [N.d.]. [Accessed 04-05-2025].

[45] *XSG SPI/I2C Solution Patches - dSPACE — dspace.com* [https://www.dspace.com/en/pub/home/support/patches/solpat/xsgspii2csp.cfm]. [N.d.]. [Accessed 04-05-2025].

List of Figures

List of Tables