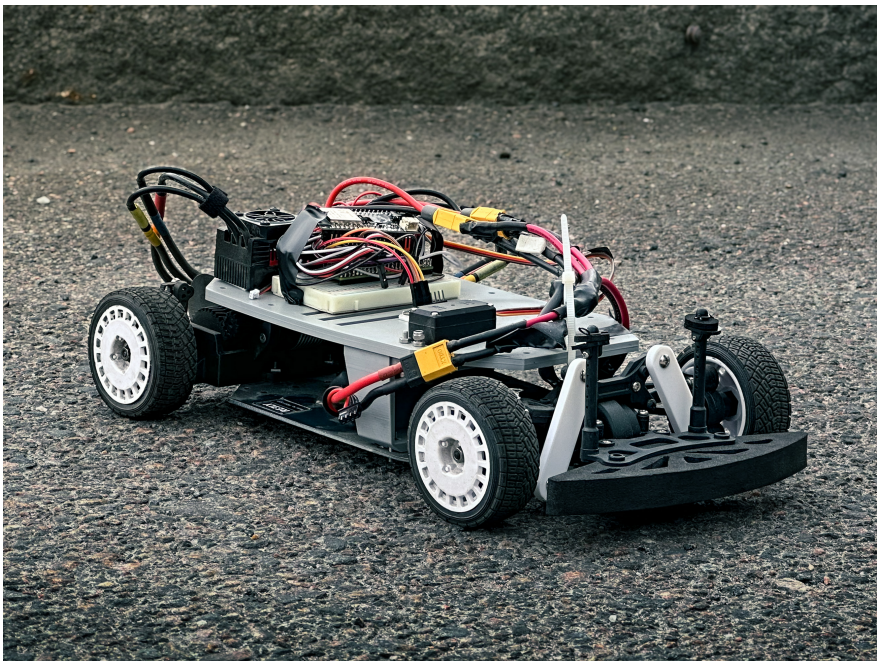
CHALMERS



Development of a small-scale electric vehicle for evaluation of torque vectoring and MPC path tracking

A study about developing a dual motor small-scale electric vehicle running on microcontrollers, with an emphasis on a proof of concept of torque vectoring and MPC path tracking on weak hardware

Bachelor's thesis in Electrical Engineering

Alfred Bäckborn, Ludvig Eriksson, Kristina Harb,
Houd Nasir, Gustav Ohlin, Johannes Solibi

DEPARTMENT OF ELECTRICAL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

Bachelor thesis 2026

Development of a small-scale electric vehicle for evaluation of torque vectoring and MPC path tracking

A study about developing a dual motor small-scale electric vehicle running on microcontrollers, with an emphasis on a proof of concept of torque vectoring and MPC path tracking on weak hardware

Alfred Bäckborn, Ludvig Eriksson, Kristina Harb,
Houd Nasir, Gustav Ohlin, Johannes Solibi

CHALMERS

Department of Electrical Engineering
Division of Systems and Control
EENX16-26-11
Chalmers University of Technology
Gothenburg, Sweden 2026

Development of a small-scale electric vehicle for evaluation of torque vectoring and MPC path tracking. Alfred Bäckborn, Ludvig Eriksson, Kristina Harb, Houd Nasir, Gustav Ohlin, Johannes Solibi

Supervisor: Wenliang Zhang, Department of Electrical Engineering
Examiner: Nikolce Murgovski, Department of Electrical Engineering

Bachelor Thesis 2026
Department of Electrical Engineering
Division of Systems and Control
EENX16-26-11
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: The small-scale electric vehicle platform developed for torque vectoring and MPC.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2026

Development of a small-scale electric vehicle for evaluation of torque vectoring and MPC path tracking

A study about developing a dual motor small-scale electric vehicle running on microcontrollers, with an emphasis on a proof of concept of torque vectoring and MPC path tracking on weak hardware

Alfred Bäckborn, Ludvig Eriksson, Kristina Harb, Houd Nasir, Gustav Ohlin, Johannes Solibi

Department of Electrical Engineering
Chalmers University of Technology

Abstract

In this thesis, a small-scale electrical vehicle with independently driven rear wheels is developed to implement and test torque vectoring and MPC using microcontrollers. The project investigated how torque distribution affects stability, maneuverability and trajectory tracking during representative driving maneuvers. A scale RC car chassis was purchased and equipped with custom mechanical components, electric motors, electronic speed controllers, battery, microcontrollers and an internal measuring unit to form a closed-loop control platform. The control system combined MPC using the TinyMPC library for path tracking with a PID-controller based torque vectoring strategy to influence yaw behavior. The system was evaluated through simulations and practical testing, where vehicle trajectories were extracted from video recordings using image processing.

The results showed that torque vectoring generated noticeable but moderate handling improvements. However, performance was limited by hardware constraints, simplified modeling, incomplete PID-controller tuning, limited torque authority and sensor limitations. Nevertheless, the project demonstrates that torque vectoring can improve controllability on a low-cost small-scale electric vehicle platform, as well as validating that MPC can be implemented on limited hardware, such as microcontrollers. It also shows that further development is required for more robust and quantitatively reliable results. Recommended improvements include systematic controller tuning, improved state estimation, onboard data logging, slip-angle control implementation, PWM-to-torque mapping, and more capable actuators.

Keywords: electrical vehicle, torque vectoring, model predictive control, microcontroller, autonomous driving.

Acknowledgements

We gratefully acknowledge the support of our supervisor Wenliang Zhang for his expertise and guidance throughout the project.

We would also like to thank our examiner Nikolce Murgovski for the constructive feedback and thoughtful evaluation of our research.

Alfred Bäckborn, Ludvig Eriksson, Kristina Harb, Houd Nasir, Gustav Ohlin, Johannes Solibi

Gothenburg, May 2026

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

CAD	Computer Aided Design
CG	Center of Gravity
CPU	Central processing Unit
ESC	Electronic Speed Controller
IMU	Inertial Measurement Unit
LQR	Linear Quadratic regulator
MPC	Model Predictive Control
PETG	Polyethylene Terephthalate Glycol
PLA	Polylactic Acid
PID	Proportional Integral Derivative
RPM	Revolutions Per Minute
TV	Torque Vectoring
Wi-Fi	Wireless Fidelity

Nomenclature

Below is the nomenclature of parameters and variables that have been used throughout this thesis.

Parameters

ℓ_f	Distance from CG to front axle
ℓ_r	Distance from CG to rear axle
$C_{\alpha f}$	Front tire cornering stiffness (per tire)
$C_{\alpha r}$	Rear tire cornering stiffness (per tire)
m	Vehicle mass
I_z	Yaw moment of inertia about CG (z-axis)

Variables

y	Vehicle lateral position at CG
$\dot{y}$	Lateral velocity at CG
ψ	Vehicle yaw angle
$\dot{\psi}$	Yaw rate
V_x	Longitudinal velocity at CG
δ	Front wheel steering angle

Contents

List of Figures

1
Introduction

The rapid development of electric vehicles in recent years has enabled new possibilities for vehicle control that are difficult to achieve with conventional combustion engine vehicles. In particular, the use of independently actuated electric motors at the rear wheels allows direct control of the individual wheel torques. This enables advanced control strategies such as torque vectoring (TV) where torque is actively distributed between wheels to improve vehicle stability and handling [2]. As vehicle performance and safety requirements continue to increase, there is a growing need for control methods that can operate effectively during aggressive maneuvers and under varying driving conditions.

Several approaches have been developed to improve vehicle stability. A common method is electronic stability control, which uses differential braking to correct vehicle motion [3]. Electronic stability control is widely implemented and effective in preventing loss of control, but relies on braking interventions, which can reduce vehicle speed. Another approach is active steering, where the steering angle is adjusted automatically to stabilize the vehicle [4]. While this method can improve handling, it is limited by the physical constraints of the steering system and may not respond sufficiently fast in critical situations. Compared to these methods, TV allows direct manipulation of wheel torques without relying on braking or steering which enables faster response and improved energy efficiency.

TV has therefore attracted significant attention in both academia and industry due to its potential to enhance safety and performance during demanding driving scenarios such as high-speed cornering. By redistributing torque between inner and outer wheels the system can counteract understeer and oversteer by optimizing tire grip usage. This allows improved stability without reducing output torque and providing more precise control compared to traditional stability systems [5]. Improved vehicle stability is directly linked to driver safety as loss of control is a major contributing factor in traffic accidents. By assisting the driver in maintaining stable vehicle behavior during critical maneuvers, TV can reduce the risk of skidding and unintended lane deviations which may help prevent accidents.

Despite its advantages, the implementation of TV presents several challenges. Determining how torque should be distributed between wheels requires accurate estimation of vehicle states and tire forces which are nonlinear and difficult to measure directly [6]. The control system must also remain robust under varying speeds and road conditions while avoiding instability or excessive actuator demand. These chal-

lenges highlight the need for systematic control strategies that can handle nonlinear dynamics and uncertainties in real time [7].

Another consequence of the rapid development of electric vehicles is the possibility for autonomous driving. Different control methods for making vehicle systems function without direct driver input have been researched, among these is the Model predictive controller (MPC) [8]. By using a mathematical model of the car as well as estimation of states to predict how control inputs will effect vehicle behavior during future discrete time steps, the MPC provides a great framework for trajectory tracking. A possible detriment of the MPC however is the relatively high computational load [9].

Based on these considerations, the objective of this work is to develop and evaluate a control strategy for TV that improves vehicle stability while remaining efficient and robust under varying driving conditions, as well as implementing a MPC that can run beside and integrate with the TV on limited hardware.

1.1 Purpose

The purpose of this thesis is to design, implement, and experimentally evaluate a TV control system for a small-scale electric vehicle with independently driven rear wheels. The study focuses on how torque distribution affects vehicle stability, maneuverability, and trajectory tracking during representative driving maneuvers. A model-based control framework is used, where a MPC controller generates vehicle level commands that are distributed to the rear wheel motors. An experimental platform is developed to validate the approach and evaluate both system performance and real-time feasibility on embedded hardware.

- The system controls a small-scale vehicle with at least two independently driven motors and enables TV through differential torque distribution.
- The control system includes MPC based trajectory-tracking.
- The system uses a small-scale platform of 2–4 kg, operates within 3.3–24 V, and includes motors, sensors and a controller. The total cost does not exceed 5000 SEK.
- Vehicle states are estimated using inertial measurement unit (IMU) data. The design accounts for sensor errors and system limitations. Experimental tests are used to evaluate performance and identify limitations.

1.2 Limitations

To be able to make the thesis feasible, the following limitations were applied:

- The vehicle will be in RC-format and therefore the dimensions will be restrained to approximately 400×200mm.
- The car will be tested at slower speeds than normal cars, keeping the maximum velocity under 10m/s.
- An optimized library based MPC controller will be used, therefore building a MPC controller from ground up will not be discussed.
- For physical testing, only an offline MPC will be used. An offline MPC only calculates the more complex calculations once per run, which restrains each test run of the MPC to be set with a constant speed.
- No optimization of the MPC will be done, instead basic settings will be used to simplify the process. This also means that no consideration in the results will be taken to different MPC settings.
- The aim of the MPC is path tracking, not autonomous driving. Therefore no sensors of the surrounding environment will be used. However, internal positioning systems will be used.
- The dynamic bicycle model is used for vehicle dynamics estimations.
- Finally, the thesis's timespan is limited to 16 weeks and with a budget of 5000 SEK.

2
Theory

This chapter presents the theoretical foundations underlying the implementation of TV and MPC on a multi-motor electric vehicle platform. Small-scale experimental platforms provide a practical and cost-effective method for evaluating vehicle dynamics control strategies. Although the dynamic behavior differs from that of full-scale vehicles, scaled platforms still allow the interaction between sensing, control and actuation systems to be studied experimentally. They also enable rapid prototyping and repeated testing under controlled conditions. The purpose of this chapter is to establish the fundamental principles that govern the hardware architecture and control system design.

2.1 Overview of hardware layout

The implementation of TV on a small scale electric vehicle platform imposes specific requirements on the hardware architecture. In order to enable independent torque control of each wheel, the drivetrain must provide fast and repeatable torque response, accurate state measurements, and sufficient computational capability for real-time control [2]. The system integrates several hardware subsystems including electric motors, motor controllers, sensors, batteries, and an onboard microcontroller responsible for executing control algorithms.

The microcontroller first fetches the initial instruction stored in its memory, usually in flash memory [10]. Flash memory means that data remains stored even when the power is turned off [11]. This marks the beginning of the program that the microcontroller will execute. The Central processing unit (CPU) then continuously reads instructions from the program memory one at a time. These instructions determine what operations the microcontroller should perform, such as reading data from a sensor or controlling an external device. After an instruction is fetched, the CPU decodes it to determine the required action. The instruction may involve mathematical calculations, logical operations, or transferring data between different parts of the system. Once decoded, the CPU executes the instruction and stores the result in a register, sends it to a peripheral such as general purpose input/output pins, or uses it to control connected hardware components.
The mechanical structure of the platform is provided by the vehicle chassis, which supports the drivetrain components and electronic hardware. Each rear wheel is

driven by an electric motor that can be controlled individually through motor controllers. Independent control of the wheels allows the system to actively distribute torque between the wheels to influence vehicle motion. This principle forms the basis of TV, where different forces applied to the wheels generate a yaw moment that affects stability during cornering.

Electric speed controllers (ESCs) regulate the electrical power supplied to the motors and thereby control the produced torque [12]. These controllers receive command signals from the onboard microcontroller and convert them into the appropriate current supplied to the motors. By adjusting the torque delivered to individual wheels, the control system can influence the vehicle's handling characteristics and improve stability during dynamic maneuvers. Proper torque allocation strategies are therefore an important part of TV systems in electric vehicles.

The vehicle is equipped with sensors that measure relevant physical parameters. To do this, an inertial measurement unit (IMU) is mounted in the center of gravity (CG) of the platform to provide the best estimation of the vehicle. IMU's measure quantities such as acceleration and rotation. These measurements are used to estimate the vehicle's motion and provide feedback to the control system.

2.2 Control system variables and parameters

A mathematical model is required to analyze the motion of the vehicle and to develop control strategies for TV and trajectory tracking. In this thesis, the lateral vehicle dynamics are described using the dynamic bicycle model [13]. This model is commonly used in vehicle control because it captures the main behavior of a vehicle during cornering while remaining simple enough for simulation and real-time control [14]. The dynamic bicycle model represents the four-wheel vehicle as a two-wheel system. The left and right wheels on each axle are combined into a single equivalent wheel. This results in one front wheel and one rear wheel. The model assumes that the vehicle behaves symmetrically around its longitudinal axis. Under this assumption, the lateral dynamics can be studied without modeling each wheel individually[15].

The vehicle motion is described relative to the center of gravity (CG) [16]. The lateral position of the vehicle is denoted by y. The vehicle orientation is described by the yaw angle ψ. The yaw rate $\dot{\psi}$ represents the rotational motion around the vertical axis [14]. The lateral velocity $\dot{y}$ describes motion perpendicular to the forward direction. The vehicle moves forward with longitudinal velocity V_x. The steering angle of the front wheels is denoted by δ. These variables influence the lateral forces generated by the tires and therefore affect the vehicle motion [14].

Several physical parameters define the vehicle properties. These include the vehicle mass m and the yaw moment of inertia I_z. The geometric properties are given by the distances from the CG to the front and rear axles, ℓ_f and ℓ_r. Tire behavior is represented by the cornering stiffness coefficients $C_{\alpha f}$ and $C_{\alpha r}$. These parameters describe how lateral tire forces develop when the tires experience slip angles [14].

For controller design, the longitudinal velocity V_x is assumed constant during the maneuvers considered in this work. Under this assumption, the lateral vehicle dynamics can be represented linearly. The model states are the lateral position y, lateral velocity $\dot{y}$, yaw angle ψ, and yaw rate $\dot{\psi}$. The steering angle δ is treated as the control input. The longitudinal velocity V_x acts as an operating input. Using these parameters, the linearized dynamic bicycle model can be written in state-space form.

$$\frac{d}{dt}\begin{bmatrix} y \\ \dot{y} \\ \psi \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\dfrac{2C_{\alpha f}+2C_{\alpha r}}{mV_x} & 0 & -V_x - \dfrac{2C_{\alpha f}\ell_f - 2C_{\alpha r}\ell_r}{mV_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\dfrac{2\ell_f C_{\alpha f} - 2\ell_r C_{\alpha r}}{I_z V_x} & 0 & -\dfrac{2\ell_f^2 C_{\alpha f} + 2\ell_r^2 C_{\alpha r}}{I_z V_x} \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \\ \psi \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \dfrac{2C_{\alpha f}}{m} & 0 \\ 0 & 0 \\ \dfrac{2\ell_f C_{\alpha f}}{I_z} & \dfrac{1}{I_z} \end{bmatrix} \begin{bmatrix} \delta \\ M_z \end{bmatrix} \tag{2.1}$$

Equation 2.1 describes the lateral and yaw motion of the vehicle as a function of the steering input. The M_z term is the added moment used for TV. The model is used as the plant model for the model predictive controller implemented. The physical variables are measured using scales and rulers, with the notable exception that the cornering stiffness of the tires is approximated from several online sources. This fact and it's consequences is discussed later in the report.

To control the steering of the vehicle, a MPC will be used to enable autonomous path following based on the estimated vehicle position and a predefined track. MPC is a model-based control method where a mathematical model of the system is used to predict future vehicle behavior. The controller calculates a sequence of control inputs that minimizes the difference between the predicted vehicle motion and the desired reference trajectory. The first optimal input is applied in the current timestep, for the optimization to be calculated in the next timestep. The MPCs precision relies on selected prediction horizons and control horizons which control how complex each calculation will be in each time step [17]. The prediction horizon defines the number of timesteps the model predicts into the future, while the control horizon specifies the number of timesteps for which the controller calculates optimal control inputs. Therefore, a compromise between accuracy and computational speed is necessary.

By being able to choose both soft and hard controller limits, the MPC computes the optimal outputs according to chosen constraints. This ability makes the MPC well suited for trajectory following applications since it aims to follow the desired path while considering physical limitations. The soft limits are suitable for desired

limitation but in some cases can be neglected, an example of this might be maximal steering angle acceleration in a self driving vehicle [17]. Another control strategy is where solving the optimization problem offline before the controller is deployed. The resulting solution is stored as a set of control laws defined over different regions of the state space. During runtime, the controller only performs a lookup operation and a simple computation to determine the control input, resulting in low computational cost. However, this advantage comes at the expense of increased memory usage, since all regions and corresponding control laws must be stored. This strategy is therefore particularly suitable for resource restrained systems with strict real-time requirements [18].

3
Methodology of development

In this chapter, the development of the TV control platform is presented. The focus is on the implementation of the system components and their integration into a closed-loop control structure. Each subsystem is described in terms of its function and role within the overall system. All software developed for this thesis that is not included in this report is available in the thesis's GitHub repository and can be provided upon request. Below is the list of components used.

Table 3.1: Hardware components used in the vehicle platform.

Component name	**Description**
Teensy 4.1	Microcontroller used as the main onboard computer.
ESP32-wemos-lolin32	Microcontroller used for wireless communication and IMU data handling.
GY-BNO055	IMU used for position and orientation measurements.
Modr brushless 3900KV sensorless	Electric motors mounted on the rear wheels.
Modr brushless speed controller 45A	Electronic speed controller used to regulate motor torque.
Modr Li-Po 11.1V (3S) 3000mAh 30C	Battery used to supply power to the motors.
Generic powerbank	External power source for the microcontrollers.
MG995 Servo	Servo motor used for steering control.

The ESP32 was chosen for its relatively low price and capabilities to WI-FI and Bluetooth and with a processing clock power of 240 MHz. It also has the capabilities to deliver PWM signals and receive the I2C signal protocol from the IMU. It has a dual core 32 bit processor which means that it can run the radio controller simultaneously as sending and receiving signals from the IMU or sending to the Teensy 4.1 [19]. The chosen ESP32 was the Wemos Lolin32. However, in a earlier testing phase it became apparent that the program would take most of the 1.3 Mb available for programs which is why the Teensy 4.1 will run the more demanding calculations.

To have high enough processing power to handle the MPC algorithm, a Teensy 4.1.

It has a clock speed of 600 MHz [20], 8 Mb of flash memory and 1 MB of RAM [20] compared to the ESP32 with 520 Kb [19]. The Teensy 4.1 was bought due to the uncertainty around the required processing power of the microcontroller.

3.1 Component and system layout

The primary function of the IMU was to continuously measure the vehicle's yaw angle in real time, enabling accurate determination of its heading relative to the initial reference position. The IMU used was the GY-BNO055. Communication between the ESP32 microcontroller and the GY-BNO055 was established according to the wiring configuration shown in Figure 3.1. To enable I2C (Inter-Integrated Circuit) communication protocol, the onboard jumpers on the IMU module was soldered, as they serve to select I2C mode.

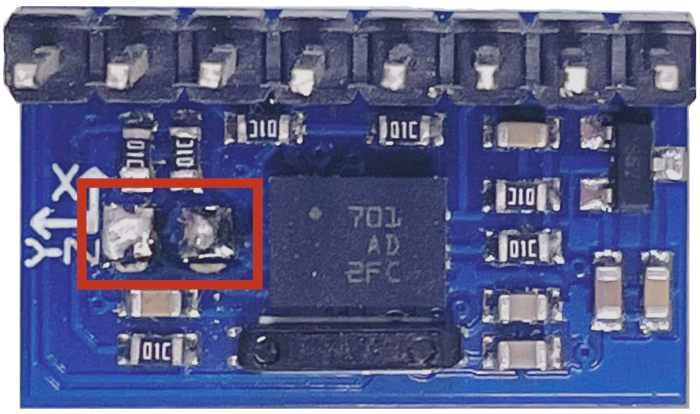


Figure 3.1: A visualization of the IMU used in the vehicle platform. The red box highlights where soldering was performed on the IMU.

The system was designed as a closed-loop control architecture in which sensing, control, and actuation were integrated to enable TV functionality. The onboard microcontroller (ESP32) continuously received motion data from the IMU. It was then transmitted to the Teensy 4.1 which used it to estimate the vehicle's current dynamic state. Based on this state estimation, the control algorithm calculated the required torque distribution and transmitted the corresponding commands to the ESCs. The ESCs then regulated the torque delivered by each motor by regulating the voltage from the battery, allowing the system to directly influence vehicle dynamics.

This architecture enabled independent wheel torque control, which was essential for

generating controlled yaw moments. Through continuous sensor feedback and real-time control updates, the system was able to adapt to changes in vehicle motion dynamically. Orientation data from the IMU was transmitted to the ESP32 according to the communication architecture shown in Figure 3.7, after which the data was forwarded to the Teensy 4.1. The Teensy 4.1 served as the primary processing unit for the MPC and TV algorithms. The ESP32 was primarily responsible for receiving Wi-Fi signals from the remote control and transmitting these as well as IMU data to the Teensy 4.1. In this configuration, the ESP32 functioned mainly as a communication interface, while the Teensy 4.1 managed all core control computations related to TV and MPC.

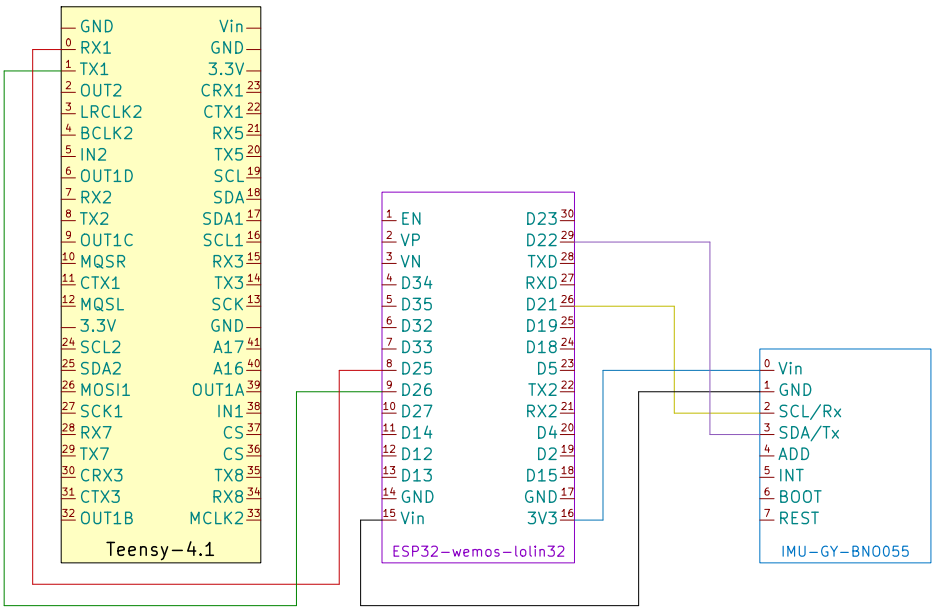


Figure 3.2: Pin wiring diagram of Teensy, ESP32 and the IMU sensor.

The ESCs were connected to the battery in parallel, as shown in Figure 3.3, ensuring that each ESC received the same supply voltage. This configuration was essential, since the rotational speed of brushless motors is directly proportional to the applied voltage. The motors were controlled using 45A brushless ESCs [21]. A hardware test using the physical setup shown in Figure 3.5 was conducted to verify whether the motors could operate simultaneously at high RPM. The motors used were 3900 KV sensorless brushless DC motors [22]. A motor rating of 3900 KV indicates that, under no-load conditions, the motor speed increases by approximately 3900 RPM per applied volt. For example, applying 2 V would theoretically result in a rotational speed of 7800 RPM.

These motors were selected because the vehicle required sufficiently high speeds to produce situations where tire grip was not sufficient, which were necessary to observe measurable differences between operation with and without TV. However, motors designed for high RPM typically produce relatively low torque, which limits the vehicle's ability to generate sufficient driving force. To address this issue, a gear

reduction system was implemented, as described in section 3.5.4. Additional testing was also conducted to verify that each motor could be controlled independently at different speeds while operating simultaneously.

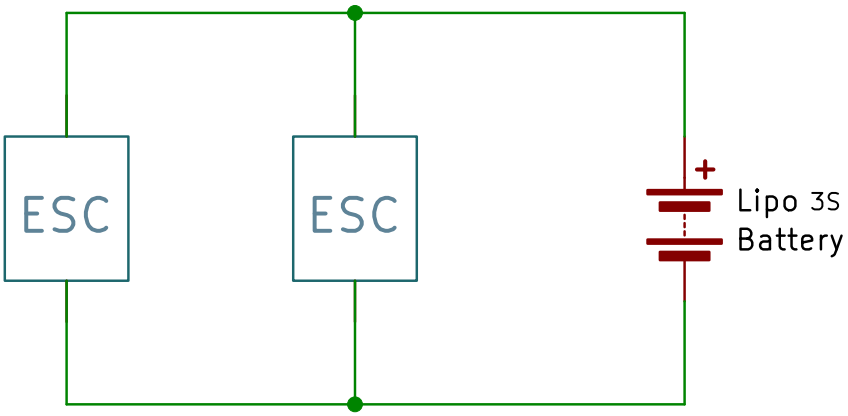


Figure 3.3: Parallel wiring between ESC and battery to provide equal voltage to both rear wheels.

The ESCs were connected to the Teensy 4.1 to enable direct motor control through PWM signals. Based on the TV algorithm, the Teensy 4.1 generated control signals that were transmitted to each ESC, which then regulated motor speed and torque by controlling the switching of the motor phases. PWM-compatible pins were selected on the Teensy 4.1, since these pins support the precise signal generation required for ESC communication. This configuration enabled fast and accurate updates of motor commands, which was essential for real-time vehicle control. By interfacing the ESCs with these PWM outputs, the control system could directly convert calculated control actions into physical torque at each wheel.

A separate power bank was used to supply power to the ESP32 and Teensy 4.1, as shown in Figure 3.4, ensuring stable operation of the control electronics independently from the drivetrain power system. The main propulsion battery selected for the vehicle was an 11.1 V 3S 3000 mAh Li-Po battery [23]. This battery was chosen because it was considered sufficiently powerful to supply the motors while remaining rechargeable and electrically compatible with the ESC connectors.

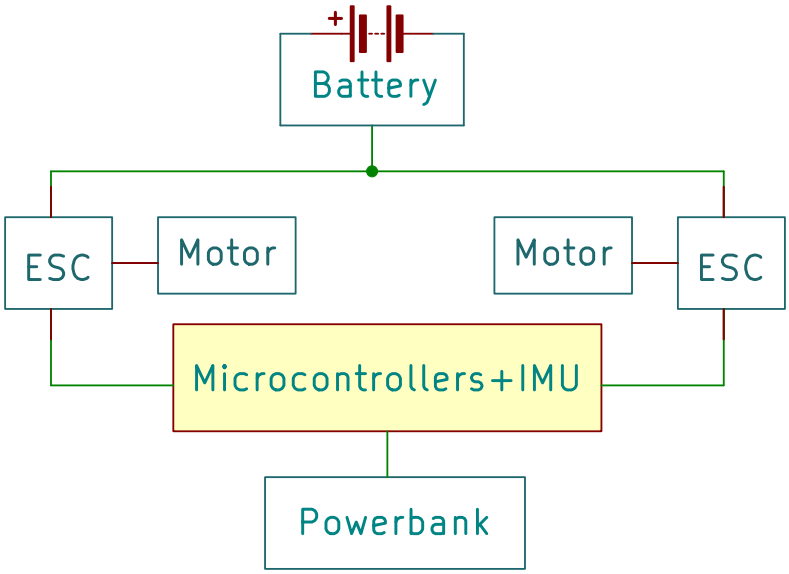


Figure 3.4: Wiring diagram of entire system including all components listed in Table 3.1.

3.2 Hardware evaluation

Motor speed control was achieved by developing code to generate and transmit PWM signals. Therefore, it was important to connect the ESC signal wire to a PWM-compatible pin. The hardware was connected as shown in Figure 3.5 to perform both individual and simultaneous hardware testing.

Figure 3.5: Physical connections of battery, adapter, and ESC to test setup.

To test the servo and IMU, both components were connected to the ESP32. A simple program was then written so that rotation of the gyroscope around the z-axis produced a corresponding rotation of the servo. This test verified the basic functionality of both components and served as an initial step toward integrating multiple system components.

To ensure proper power distribution to the ESCs, a custom power adapter was constructed. The battery could only connect to one ESC which would then be used to control one drive motor while simultaneously supplying power to auxiliary electronics such as the receiver and steering servo. Therefore, implementing a dual-motor TV system required a modified power distribution technique, with two ESCs. The adapter was created by soldering multiple wires in parallel with XT60 connectors on each end, which can be seen in Figure 3.5.

3.3 Software

In this section, the software tools that are utilized are presented and motivated.

3.3.1 Matlab and Simulink

For the purpose of developing a mathematical model, a model was built in Matlab [24] Simulink to simplify the design of the MPC controller and the PID, controller and to simulate the controllers before their initial use. The dynamics of vehicles was analyzed using the dynamic bicycle model [13] and the Simulink MPC toolbox. A scenario was built using the Driving Scenario Designer app to build the modified Moose test as seen in 5.2. The weights in the MPC were set to 10 on the lateral position and 1 on the yaw angle, the prediction horizon was 20, the control horizon was 3 and the hard constrains were set to ± 30 degrees for maximum steering angle.

3.3.2 Arduino IDE and PlatformIO

Another piece of software that was used is the Arduino IDE to write the code for the ESP32. The Arduino framework, together with additional libraries, can be used to program the controller. These libraries will simplify the handling of outputs such as servos and motors, as well as the handling of matrices and MPC.

There is an alternative software for controlling the microcontroller, PlatformIO, which is another IDE besides the Arduino IDE which runs as an add on in Visual Studio-code [25]. The advantage of using it over the Arduino IDE is that it provides suggestions for functions and variables that have been created while still utilizing the Arduino framework.

3.3.3 CAD software

The CAD modeling was carried out primarily using Autodesk Inventor PRO 2026. The software can be used to design and develop custom mechanical components. Modeling allows dimensions to be adjusted based on testing results and fitment constraints. The CAD environment also makes it possible to position and integrate components within limited available space, ensuring proper alignment and compatibility between components. The final designs are exported to STL-files and sliced using PrusaSlicer [26].

3.3.4 OpenCV and Matplotlib

The Python library OpenCV is an open source software mainly used for picture processing. In this thesis the software is used to save data from testing to later be analyzed. The initial recording can be processed as shown in Figure 3.6, using the following three main steps:

- frame-by-frame analysis of the recorded video,
- correction of the camera perspective,
- tracking of the vehicle position.

VideoCapture [27] is a tool within the OpenCV library used to analyze recordings frame by frame. To be able to draw conclusions from recordings on an angle, transforming the recordings to a top-down view is a must, which can be done using WarpPerspective [28]. From the first frame of the recording, a rectangular area in real life is marked, which makes the picture transformed to be analyzed from the desired view. This tool can be applied for each frame of the video and the four corners points of the rectangle can be saved, which enables consistent transformation over several recordings.

TrackerCSRT tool [29] is a tracking tool which relies on the user choosing the desired object to track as well as the searching area for the tracker. The tools runs for each frame of the recording and creates a dataset of points in pixels where the object was tracked. Since the the stored data are initially saved in pixel coordinates, a scaling factor needs to be applied in both direction. To do this, markers on the test area can be marked with standard measurements. The data is time stamped, which makes it possible to calculate velocities and accelerations [29]. The processed CSV data can then plotted using the Python library Matplotlib. This makes it possible to visualize both the vehicle trajectory and the corresponding velocity during each test run and a comparison between runs.

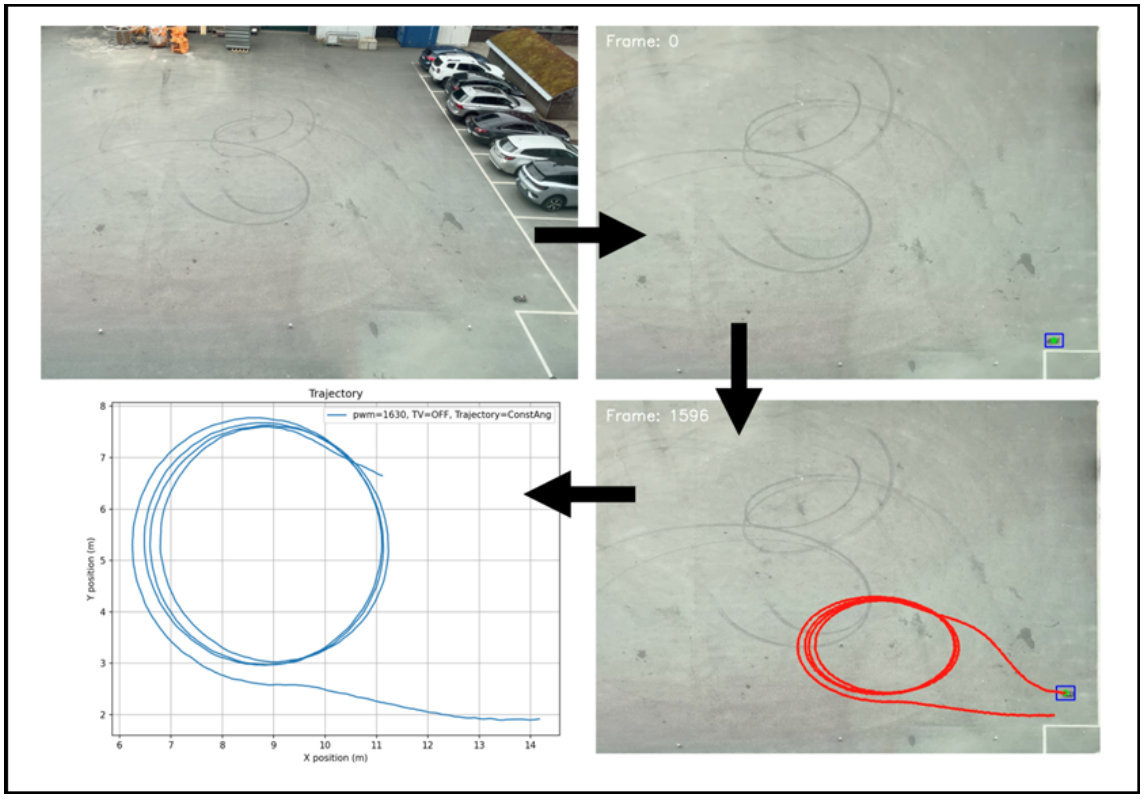


Figure 3.6: Overview of the video processing procedure used to extract the vehicle trajectory. The upper left shows the raw video. The upper right shows the transformed image with the vehicle selected. The lower right shows the tracked trajectory. The lower left shows the scaled plot.

3.4 Design and system implementation strategy

Testing requires different software to cooperate to create a fully functioning program and the program then has to be further tuned to accommodate hardware specifications. When in automatic driving mode, the MPC determines the optimal steering angle based on the reference trajectory and current state. Otherwise, when in manual mode, the steering angle is taken from the manual controller input. All other implemented software are designed to operate within this framework.

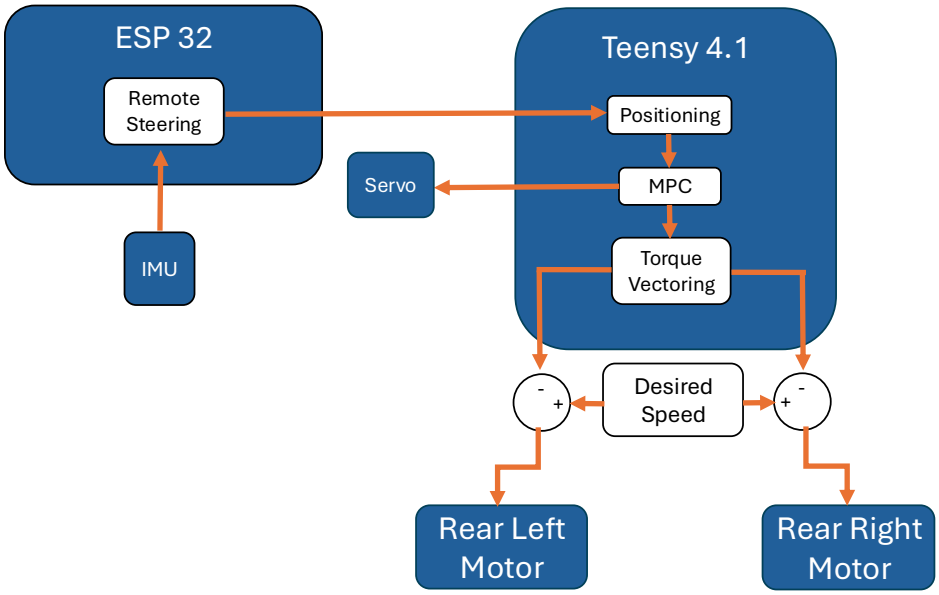


Figure 3.7: Full signal diagram for all operational software.

TV aids in stability and maneuverability by ensuring that the vehicle rotates at an appropriate rate for a given steering angle and speed. With this assumption, and from the dynamic bicycle model (see Equation 2.1), a linear relation between steering angle δ and a desired yaw rate $\dot{\psi}_{des}$ can be derived [13]

$$\dot{\psi}_{des} = \frac{V_x}{\ell_f + \ell_r + \frac{mV_x(\ell_r C_{ar} - \ell_f C_{af})}{2C_{af}C_{ar}(\ell_f + \ell_r)}}\delta. \tag{3.1}$$

A PID-controller was chosen for the TV system. The controller was verified in Simulink, with specific PID-tuning considered as part of the initial full-vehicle testing process. The equations that were written in Matlab and Simulink were transferred to C++ to be usable on the microcontroller. A library called PID_V1 was used for the controller.

Another library used for more complex mathematics and matrix handling for the MPC was the Eigen library, which was imported by the Bolderflight controller Eigen library.

TinyMPC is an open-source MPC solver optimized for systems with limited computational resources [30]. It achieves real-time performance by pre-computing the most demanding calculations at setup, leaving only lightweight operations to run onboard during testing. This pre-computation assumes that every step in the prediction horizon has a free control input, which means a separate control horizon cannot be configured. Constraints such as maximum steering angle are handled separately from the main optimization, keeping the online computation minimal. A consequence of the pre-computation is that the dynamics matrices are calculated for a fixed operating condition, which in this case requires the vehicle to maintain approximately constant velocity.

TinyMPC was implemented in the main code, where the functions for controlling the motors, IMU, servo, and calculations of torque for each back wheel was also performed. It controlled the steering of the car when running in automatic mode, and the throttle was always controlled manually from the controller for safety.

In contrast to the standard procedure for utilizing libraries in PlatformIO, the TinyMPC library must first be configured within MATLAB, where all relevant data and matrices are defined. The characteristics of the MPC were also defined, such as the prediction horizon which was 20 steps and the hard steering angle limits were set to ±30 deg. The weights were set to 100 at the lateral position, 1 for the lateral velocity, 10 for the yaw angle and 1 for the yaw rate. Subsequently, the configured setup is exported as a map, from which a subset of the generated files must be transferred into the PlatformIO project. Once this process is completed, the library can be employed in PlatformIO in the same manner as a conventional library.

The Adafruit Unified Sensor library [31] together with the Servo library [32] simplifies the handling och sensors, servos and motors, allowing the controller to send PWM signals via the digitalWrite() function. Unlike most commonly used PWM, where the duty cycle is part of the period time, the ESC instead reads how many microseconds the high value is between 1000 and 2000, where 1500 is neutral, at a frequency of 50 Hz, which means 50 pulses are sent per second.

The Teensy 4.1 communicates with the ESP32, which in turn connects it to the IMU, where communication is done serially. That is when data is sent bit by bit instead of sending several bits in parallel [33]. The IMU communicates with the Teensy 4.1 via the I2C protocol with a clock wire and a signal wire [34], which the Adafruit BNO055 library then handles.

The desired torque difference ΔT between the rear wheels is calculated using Equation (3.2) where the r_w is the radius of the wheel, l_w is the track width of the vehicle, G_r is the gear ratio between the wheels and motors and M_z which is the desired added yaw moment [35].

$$\Delta T = \frac{r_w}{l_w G_r} M_z \tag{3.2}$$

In the implemented code, this equation was used but without providing functional value. What was controlled was not the torque but the PWM difference, seen in the code snippet below. This was calculated based on the desired added yaw moment, after which a scaling factor could be applied for calibration purposes if required. While this method provided functional control, seeing as M_z was set by the PID-controller, it was not mathematically rigorous since torque and PWM represent different physical quantities and are not directly equivalent. The adjusted PWM signals were then transmitted to the ESCs through the motor control function.

```
1  void Torquediffrence(float Mz, int PWMInput) {
2
3      float torqueDiff = rw/(lw*Gr)*Mz;
4      float torqueScale = 1;
5      float diff = torqueDiff * torqueScale;
6
7      int leftTorque = PWMInput - round(diff / 2.0);
8      int rightTorque = PWMInput + round(diff / 2.0);
9
10     setMotorSpeeds(leftTorque,rightTorque);
11 }
```

3.5 Mechanical platform development

To construct the vehicle platform and include all the necessary components, an RC-car chassis (Reely TC-04) was used [36]. This chassis included various components for a four-wheel-drive setup with a single motor, such as front and rear differentials, steering, and suspension. However, due to the nature of this thesis, the chassis had to be modified to fit the new components, as the original chassis was not intended for a multi-motor configuration with an independent TV setup. The majority of these modifications were made using Autodesk Inventor to develop new parts, which were then 3D-printed.

3.5.1 Original chassis configuration

A key limitation of the original chassis was the rear drivetrain setup, which included the drivetrain, suspension, and structural components. The drivetrain consisted of the rear differential and drive shafts, while the suspension system includes control arms, camber links, and shock absorbers. These components are mounted to the chassis through elements such as the differential housing and shock tower. The rear assembly also includes hub carriers and axle components for wheel attachment. All of these components, except for the hub assembly and axle components, were disassembled to make way for the new customized components.

3.5.2 Rear drivetrain modifications

The shock absorbers and springs were replaced with customized stiff rods, as seen in Figure 3.8, making the suspension rigid. This was done to eliminate suspension-induced dynamics, such as vertical load transfer and transient oscillations, which could otherwise influence the vehicle's behavior. By removing these effects, the impact of TV could be isolated more effectively, allowing for a clearer and more controlled comparison between configurations with and without TV.

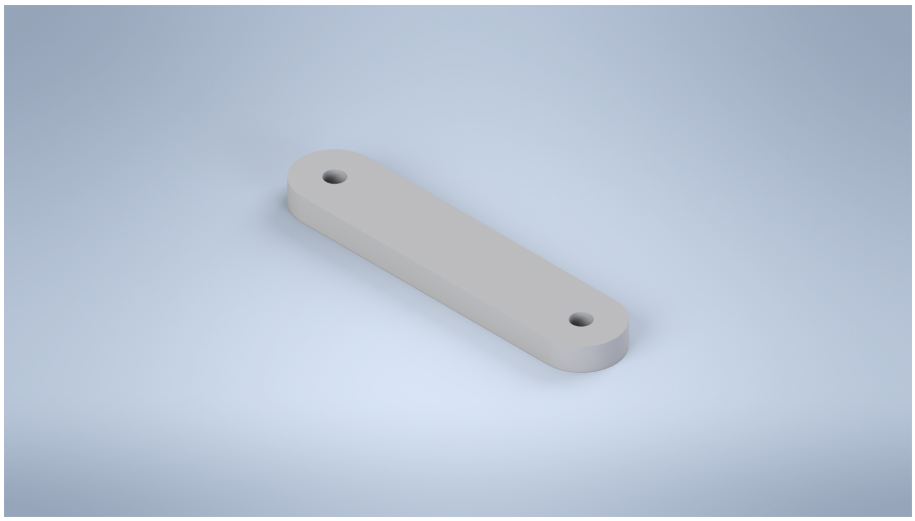


Figure 3.8: Stiff spring replacement.

A custom rear motor mount was designed to rigidly position two motors in parallel, with each motor directly connected to a rear wheel, as shown in Figure 3.9. Customized adapters were also used to connect the motors to the hub assembly. The motor mount, along with several other components, underwent multiple design revisions due to calculation errors. However, the finalized components met the functional requirements.

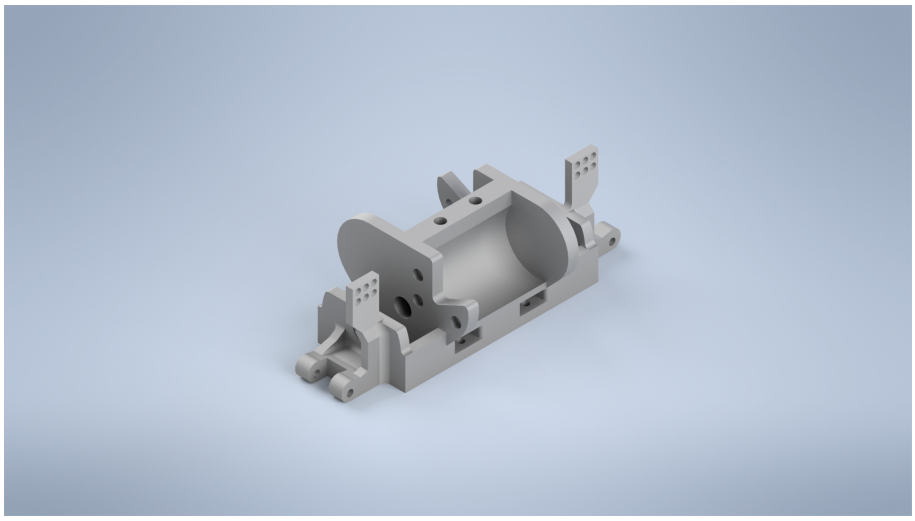


Figure 3.9: Customized mount for motors and gears.

3.5.3 Electronics mounting plate

In addition to drivetrain modifications, the integration of electronic components required further spatial adaptation. The available space within the original chassis was insufficient to accommodate all required components, including the ESP32, Teensy 4.1, IMU, servo, battery, and the two ESCs. To address this limitation, an elevated plate was designed and mounted above the chassis, as seen in Figure 3.10. The customized plate was mounted above the RC chassis and provided sufficient

space to securely accommodate all required components. PETG was chosen as material because of its high strength and less brittle properties.

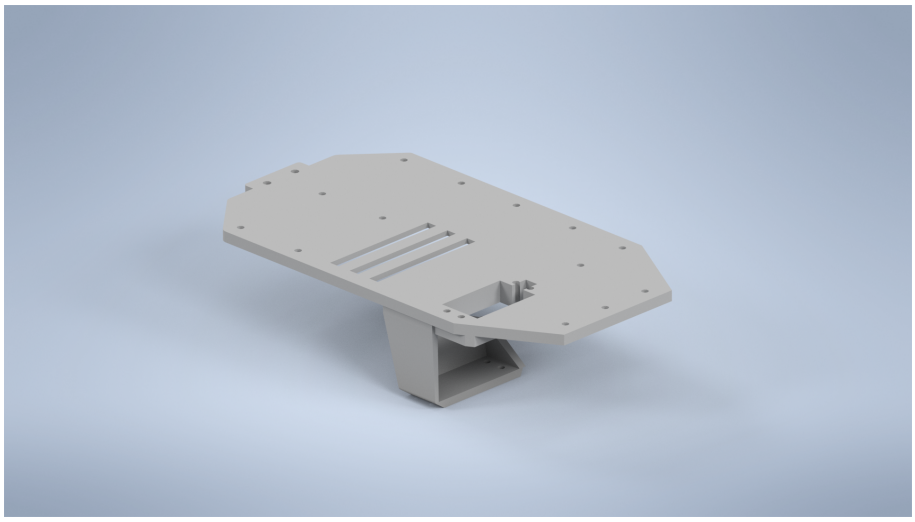


Figure 3.10: Elevated plate for electronics mounting.

To ensure precise fit of each part, a complete assembly of the created parts and the relevant bought components were made as shown in Figure 3.11. This also gave insight in scale of each component and finding a reasonable ground clearance.

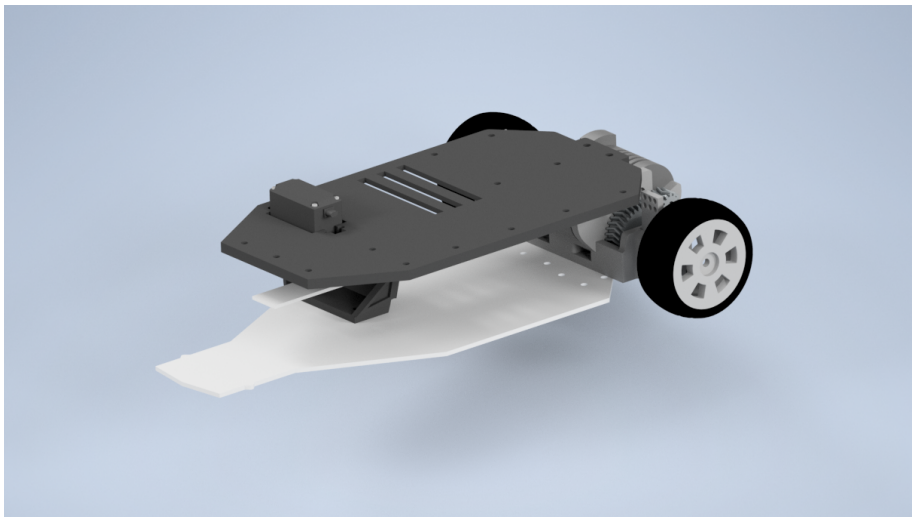


Figure 3.11: Rear assembly with motor mount, gearing system and plate.

3.5.4 Gear reduction system

Initial testing showed that the electric motors exhibited high rotational speeds but insufficient torque output. This limited their ability to propel the vehicle from a standstill. While the motors were capable of achieving high RPM under low load conditions, their low torque characteristics resulted in inadequate force at the wheels during startup. Consequently, the motors were unable to overcome the static resistance of the vehicle. Thus, a gear reduction mechanism was introduced to increase

the available torque at the wheels.

The gear ratio was designed with consideration of the high rotational speeds of the electric motors, with the primary objective of reducing rotational speed in order to increase torque output. The gear reduction was maximized within the geometric constraints defined by the available space between the ground and the drive shaft, as seen in Figure 3.12. The final gear ratio was 1:3.

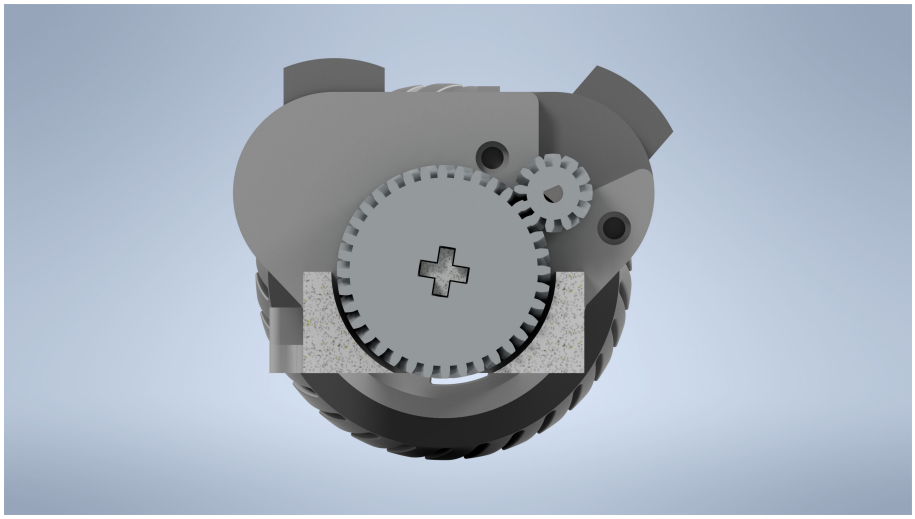


Figure 3.12: Cross-section view of the gear reduction system.

Given that the drivetrain components were manufactured using 3D-printing, particular attention was given to gear design. A relatively large gear module and a high number of teeth were selected, especially for the pinion gear mounted on the motor shaft, in order to improve durability and reduce the risk of failure. The gears utilize a herringbone tooth profile that improves axial stability and greatly reduces the risk of lateral slippage during testing. Material selection was constrained to PLA or PETG, as specified by the thesis requirements. PLA was used in the initial implementation, with PETG utilized for further testing due to its potentially improved mechanical properties.

An advantage of the selected drivetrain configuration is that the motors can be mounted parallel to the wheel axle. This results in a compact design in the lateral direction, which was important due to the limited space available on the vehicle platform. However, this configuration also increases the number of exposed moving components, which may reduce the overall robustness of the system. The motor mount integrates both the spur and pinion gears. The pinion gears are mounted directly on the motor shafts, while the spur gears transfer torque to the hub assembly through customized adapters.

Since the drivetrain design had to remain compact, a gear module of 1 mm was selected. This resulted in relatively small gear teeth, which required high manufacturing accuracy. To achieve sufficient tooth precision, the gears were printed on their side. In this orientation, the slicer generates the outer contour of each layer

with continuous filament paths, which improves the dimensional accuracy of the gear profile. This print orientation also reduces the dependence on inter-layer adhesion in the gear teeth.

However, the chosen print orientation introduces mechanical limitations. A study by Gonabadi [37] shows that components printed on their side can exhibit reduced tensile and shear strength. Since the drivetrain components are subjected to torque during operation, the mechanical strength of the printed parts had to be considered carefully. As the print orientation of the gears had already been determined by the required gear accuracy, several components were instead divided into separate parts to improve manufacturability and mechanical performance.

The axle adapter, shown in Figure 3.13, required both high dimensional accuracy and a relatively fine surface finish. To avoid support material on functional surfaces, the adapter was split into two separate parts and glued together after printing. The same method was applied to the axles connecting to the spur gear toward the center of the vehicle. This approach allowed the parts to be printed in orientations that improved surface quality and dimensional accuracy while still satisfying the geometric and strength requirements of the drivetrain.

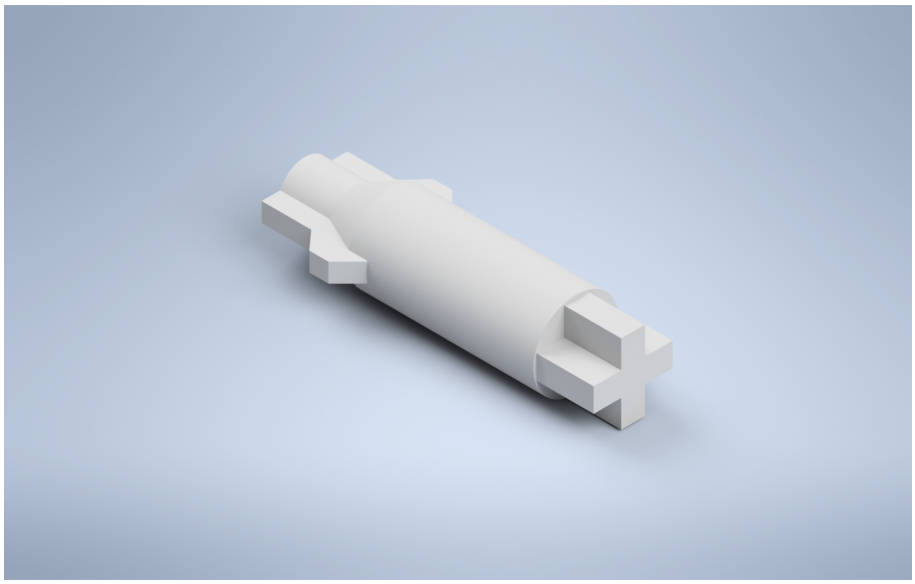


Figure 3.13: Customized axle adapter used to transfer torque from the gear to the hub.

After the drivetrain, electronics mounting plate, motors, gears, and axle adapters had been integrated, the mechanical platform formed a complete functional chassis for the experimental vehicle. The final assembly provided the required layout for two independently driven rear wheels while keeping the components compact enough to fit within the original RC chassis dimensions. With all of the components mounted to the chassis, the final weight of the vehicle was 2.33 kg, which was within the specified thesis requirement of 2–4 kg.

3.6 Remote vehicle control

The setup consists of two microcontrollers that enable wireless vehicle control through Wi-Fi communication. The ESP32 acts as the communication interface by receiving commands over Wi-Fi and forwarding them to the Teensy 4.1 through a wired serial connection. The Teensy 4.1 interprets the commands and controls two ESCs using PWM through the Servo library. After initialization, the ESCs are armed with a neutral signal, and the motor outputs are updated based on the received command. This allows forward motion, turning and stopping within safe operating limits.

Stable motor control is an important first step before implementing higher-level control strategies. Manual command input also makes it possible to verify communication between the microcontrollers and observe motor response before more advanced control algorithms are added. For ease of manual control and testing some functions were added to the controller, whose interface is visible in Figure 3.14. The steering trim function serves to center the steering by adding a constant value to the PWM signal given to the servo. The max PWM function limits the motor PWM at max throttle, aiding in holding a constant speed. Live PID tuning was also implemented on the controller.

Figure 3.14: Interface on the WI-FI remote controller.

4
Methodology of testing

To evaluate vehicle handling characteristics, MPC performance, and TV implementation, standardized tests were conducted to validate that the thesis's objectives were achieved.

4.1 Testing with simulations

To validate the accuracy of the mathematical vehicle model and MPC implementation, simulations were carried out in MATLAB and Simulink. The MPC was implemented using the integrated MPC function block in Simulink and was manually tuned. For the design of the test scenario and reference path, the MATLAB toolbox "Driving Scenario Designer" was used to generate a trajectory analogous to the standardized moose test, illustrated in Figure 4.1.

The moose test, also referred to as a double lane change or obstacle-avoidance maneuver, is standardized according to ISO 3888-2 [38], and is commonly used to assess a vehicle's dynamic response to sudden evasive steering inputs with the objective of evaluating properties such as road holding ability and vehicle dynamic behavior. However, it was necessary to adjust the approach since it was a scaled vehicle. The test was conducted at a speed of 5 m/s with the course more augmented than it would be in the scaled case. In order to evaluate both the stability of the MPC and its reaction to sudden maneuvers. These MPC properties, including weights, constraints and control horizon, are then exported to TinyMPC and implemented in the physical vehicle model.

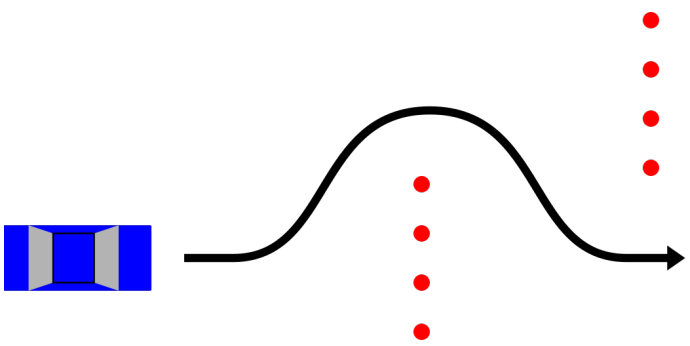


Figure 4.1: Illustration of the vehicle path during a moose test [1].

4.2 Physical testing

To evaluate the performance of the car and TV, a decision was made to make a simple initial test, such as the skidpad test, without MPC and with the use of a radio controller instead. The skidpad test, illustrated in Figure 4.2, is a vehicle dynamics test used to evaluate cornering performance. It is typically performed on a circular track with a constant radius or on a figure-eight track, where the vehicle is driven under controlled conditions to maintain a steady path [39].

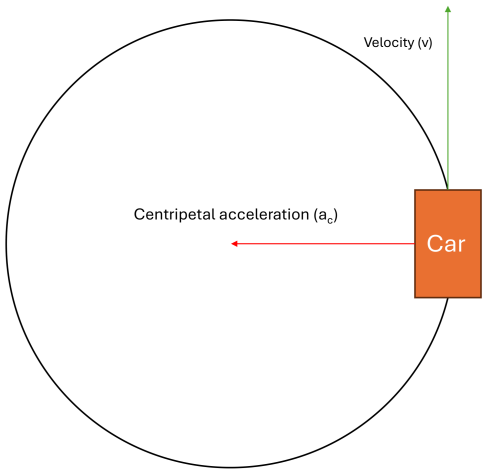


Figure 4.2: Illustration of the vehicle path during a skidpad test.

However, since it is a small scale vehicle which was difficult to control, some adaptations had to be made to the test. For example, during a regular skidpad test, the driver gradually increases speed while maintaining the circular trajectory. At a certain point, the vehicle begins to lose grip and exhibits either understeer or oversteer, depending on the distribution of lateral forces between the front and rear axles. In the performed testing however, we maintained a constant speed while slowly increasing the steering angle. This resulted in an similar dynamic response as during the skidpad test. To study the TV-systems influence on handling abilities the modified skidpad test was done both with and without TV.

To further study the car's behavior a simple test was conducted by trimming the steer angle from its straight position to a few degrees to the right. This made the car turn at an constant angle which made it possible to study the speed of the car while cornering as well as the effect of the TV when turning slowly. To analyze the MPC a simple reference trajectory was produced to simplify the trouble shooting and analyzing of results. Therefore a circle with 5 m radius was constructed as the reference path. Since the MPC relies on a specific speed, in this case 5 m/s the car

could not start stationary and instead was turned on mid run.

To evaluate the performance of the vehicle during experimental testing, an external motion tracking system was implemented. The system used a phone camera together with the Python library OpenCV, to extract the vehicle trajectory from recorded test runs, as described in section 3.3.4. The test area was arranged as shown in Figure 4.3, using three cones placed with a distance of 5 m between each cone and an angle of 90 degrees between the reference directions.

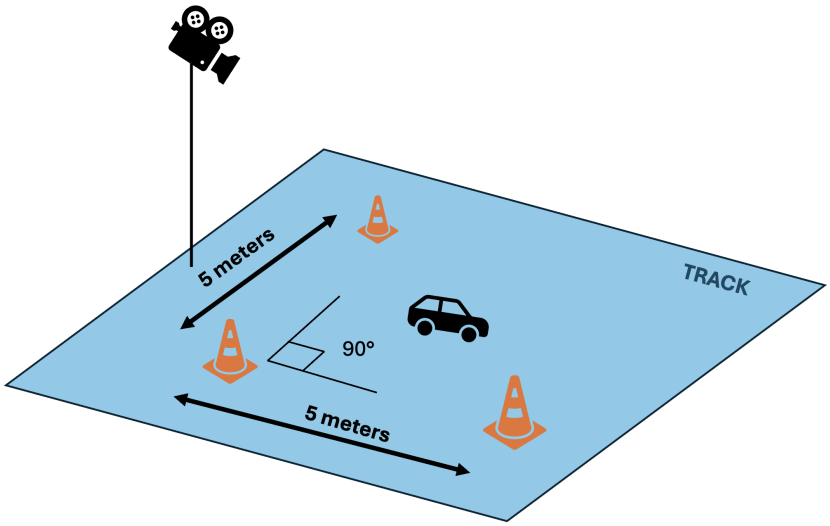


Figure 4.3: Overview of the experimental test area.

A fixed camera rig was constructed to ensure that the camera position and viewing direction remained consistent between tests. The camera recorded at 60 frames per second, which increased the number of available data points and improved the resolution of the measured vehicle motion during fast maneuvers.

5
Results

The results from both simulation and practical testing is presented with focus on numerical data illustrated in graphs.

5.1 Simulations

Simulations were conducted on a modified version Moose test during 10 s to evaluate both the effects of the TV and MPC. In Figure 5.1, it is shown how the maximum moment generated by the TV is 0.10 Nm at 3.70 s and the lowest is -0.10 Nm at 3.70 s. It is visible how the TV follows the curve in Figure 5.2.

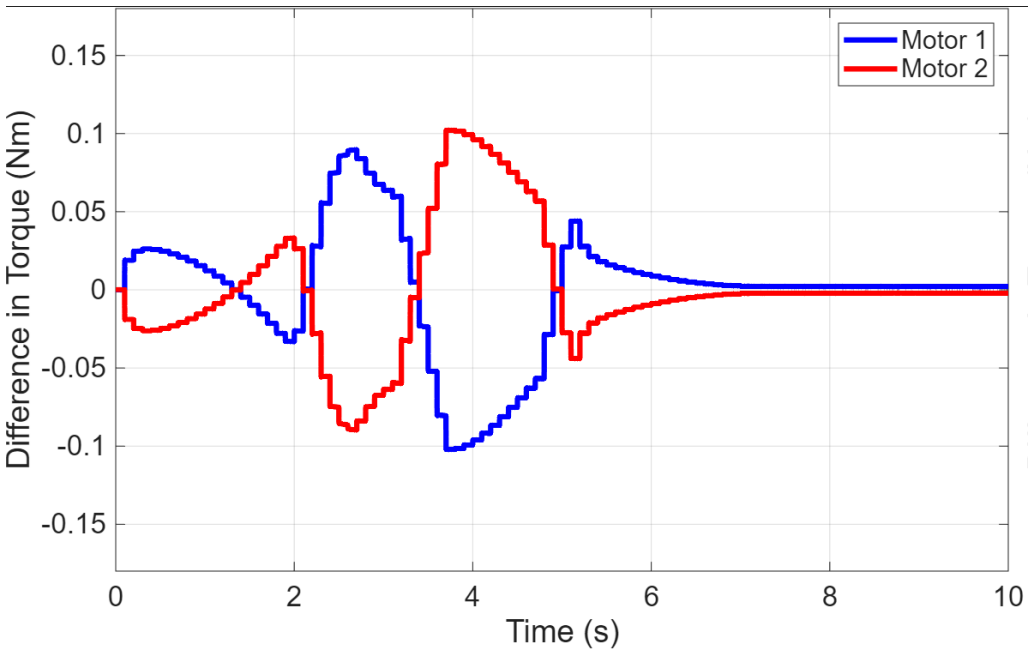


Figure 5.1: The figure shows the difference in torque between the two motors in the rear section.

In the simulations of the lateral position in Figure 5.2 the lateral position of the car, as well as the reference lateral position. The peak value for the reference position is 3.56 m at 2.66 s and the peak value for the Vehicle position is 3.22 m at 3.40 s which gives a time difference between the reference position and vehicle position of

0.75 s in time difference. The minimum point for the reference position is -1.45 m at 4.33 s and for the vehicle position -1.23 at 5.00 s which gives a time difference at 0.67 s which means that the MPC closes the gap. The more the time goes the closer does the MPC come to the reference value.

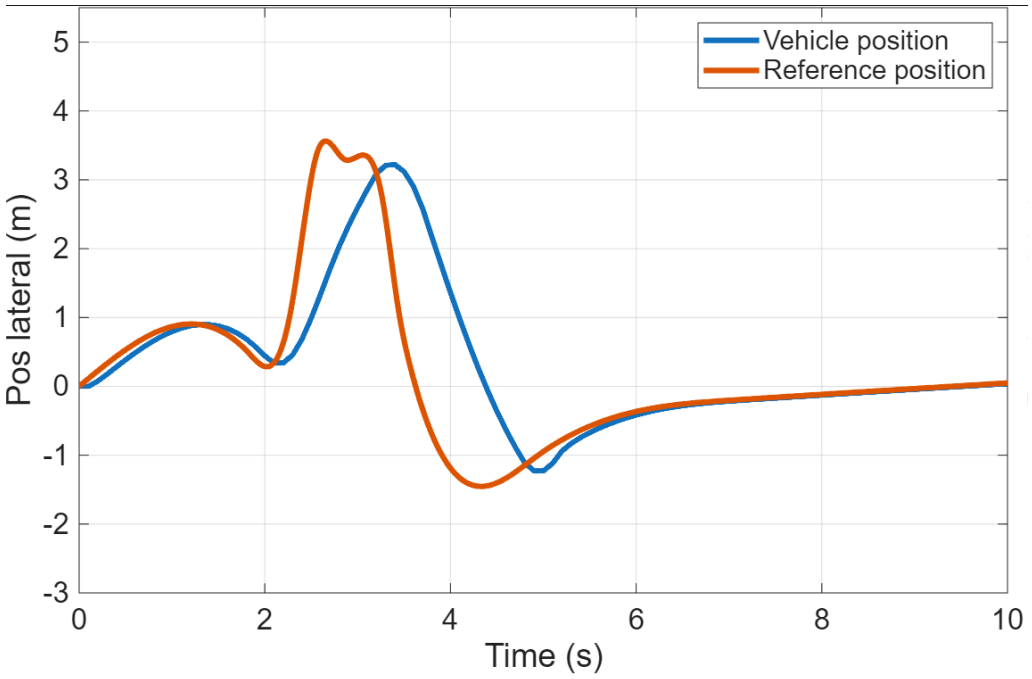


Figure 5.2: The figure shows how well the MPC follows the reference path.

5.2 Testing of vehicle and program

The graph illustrates the speed achieved at two PWM input values that are close to each other. It also shows the ability to maintain the same speed time for different PWM inputs 5.3.

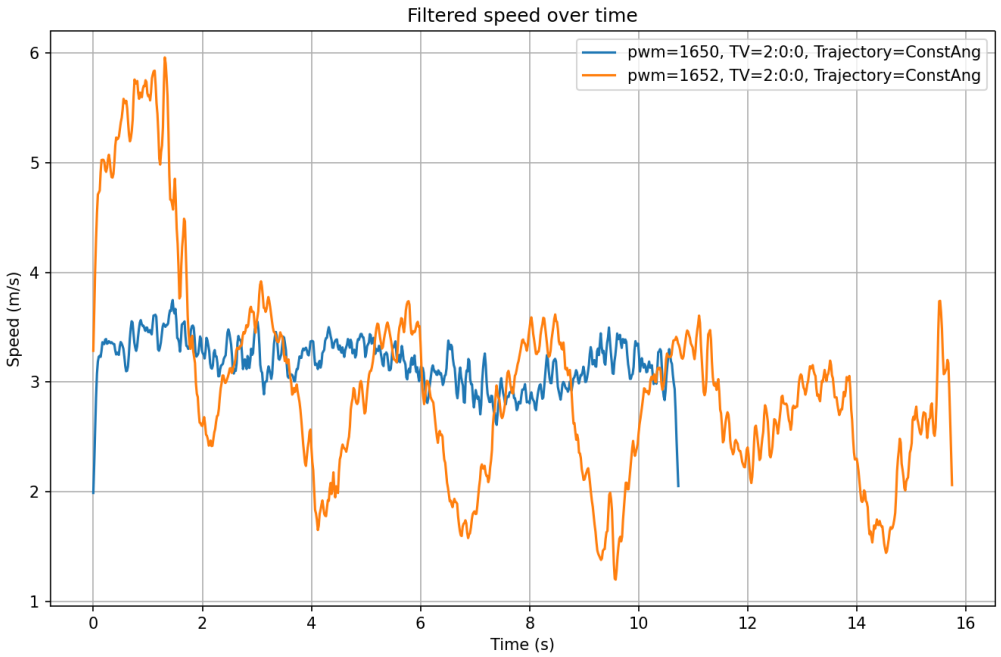


Figure 5.3: The figure shows how the velocity correlates to different PWM input values.

The car's driving trajectory during the constant angle test using the steering trim function at two different velocities is seen in Figure 5.4.

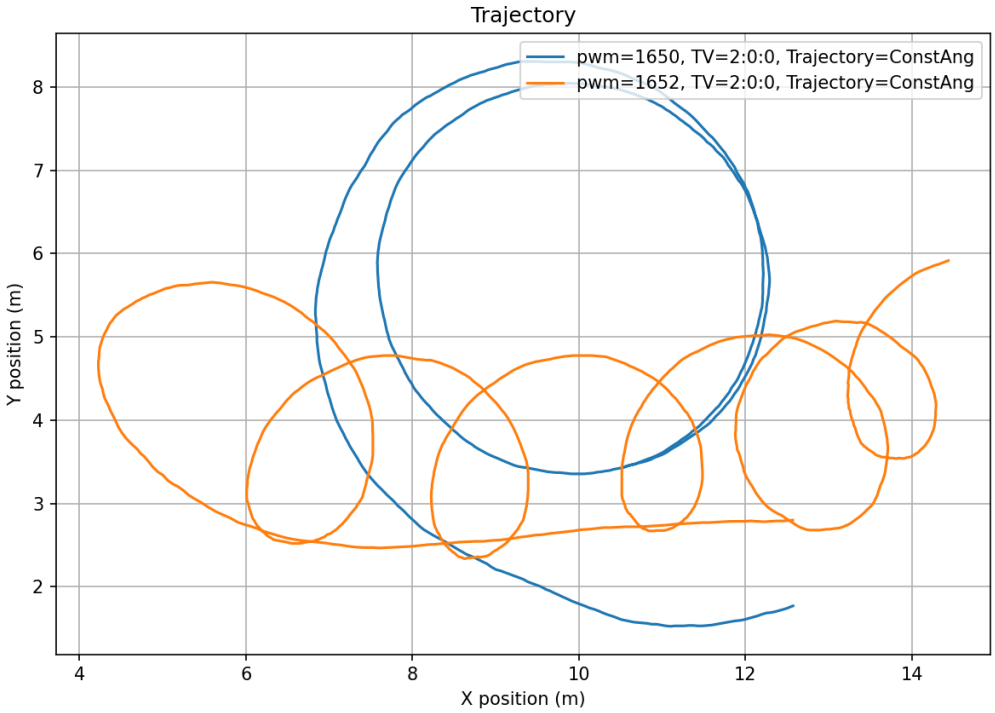


Figure 5.4: The figure shows the constant angle test using the steering trim function at two different PWM input values.

5.3 Practical maneuvers testing

In this section, the results related to the practical testing of the systems aiding in driving maneuvers are presented.

5.3.1 MPC

The MPC was evaluated by commanding the vehicle to follow a circular path with a radius of 5 m while assuming a constant vehicle speed of 5 m/s. During testing, the vehicle speed was software-limited to a PWM value of 1630, which was maintained throughout each run. Two separate tests were conducted using different TV PID settings: 2,0,0 and 3,0,1. Figure 5.5 presents the vehicle speed over time for both tests.

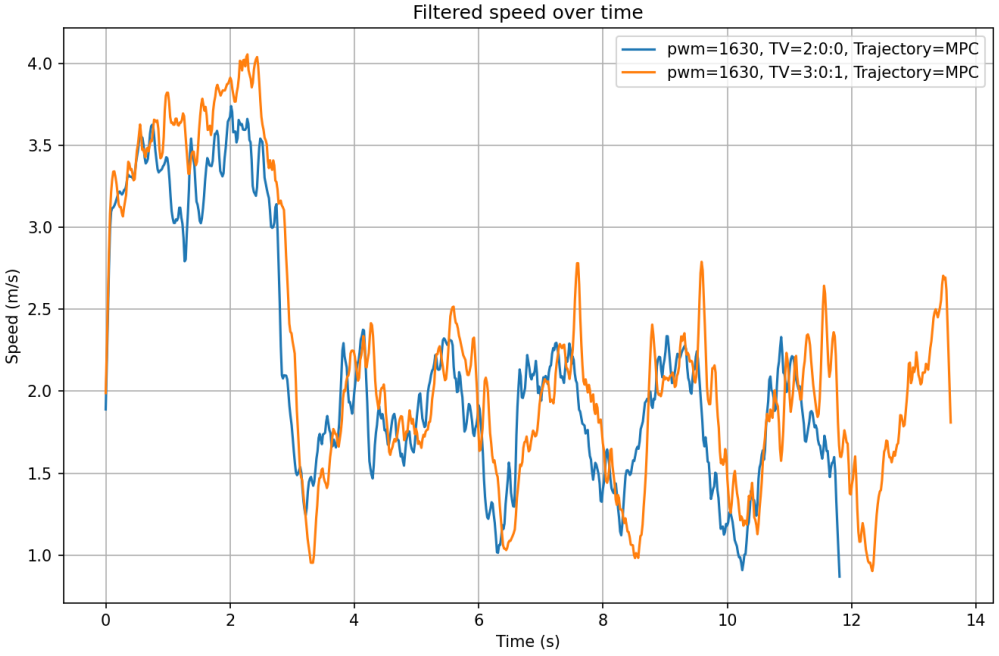


Figure 5.5: Speed of the car during the two tests, with respect to time.

The trajectories corresponding to the speeds in Figure 5.5 are illustrated in Figure 5.6.

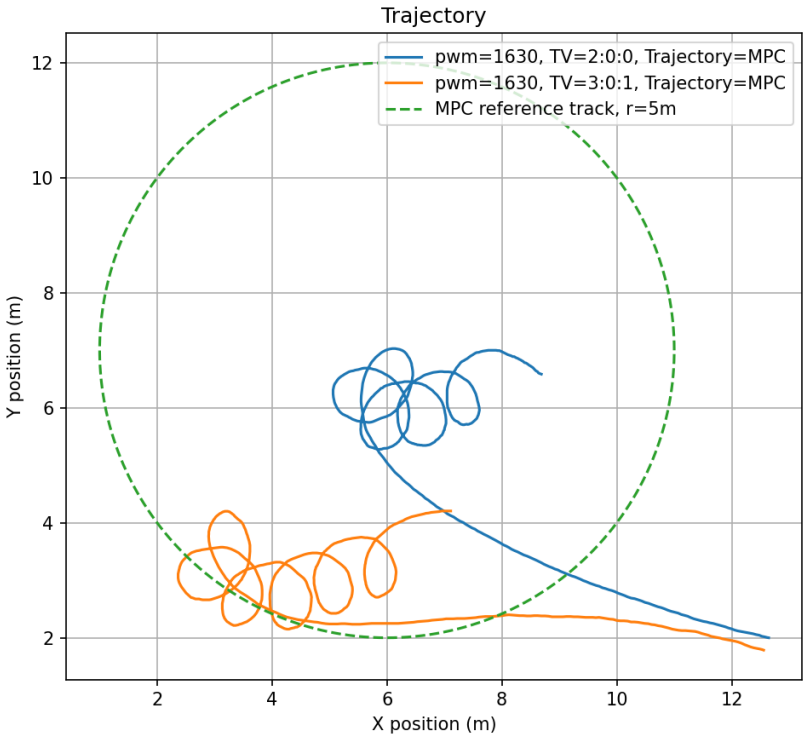


Figure 5.6: Trajectory of the car during the two tests, shown on a grid, starting at the bottom right corner before accelerating and enabling the MPC.

5.3.2 TV

The TV algorithm was tested using the steering-trim function of the remote control, setting a constant steering angle which remained constant over two scenarios. The first scenario drove without TV enabled, and the second enabled TV with a PID-tuning of 2,0,0. In Figure 5.7, the difference in trajectory between the tests with and without TV is shown.

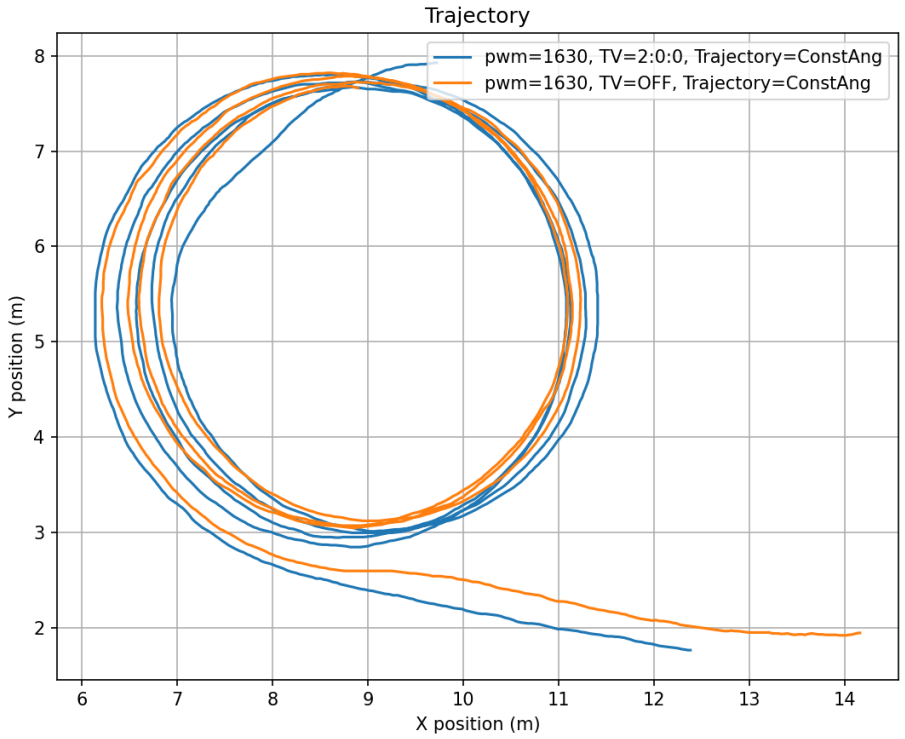


Figure 5.7: Trajectory of the car during the two tests, shown on a grid, starting at the bottom right corner before accelerating and enabling the constant steering angle.

In Figure 5.8 the difference in speed between the two scenarios is presented.

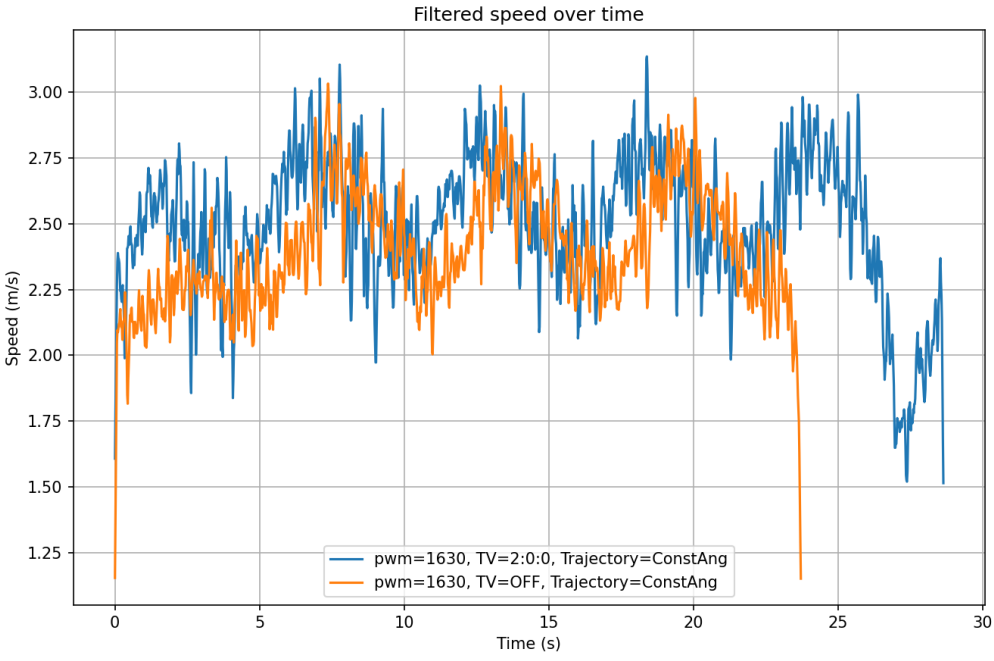


Figure 5.8: Speed of the car during the two tests, with respect to time.

A skilled driver performed the modified skidpad test. The PID-tunings were set to different values. The tested values were 10,0,0 and 10,0,4, hence adding derivative action. The tests were both done with motor PWM set at 1660. In Figure 5.9, the trajectories of the car during the two modified skidpad tests are shown.

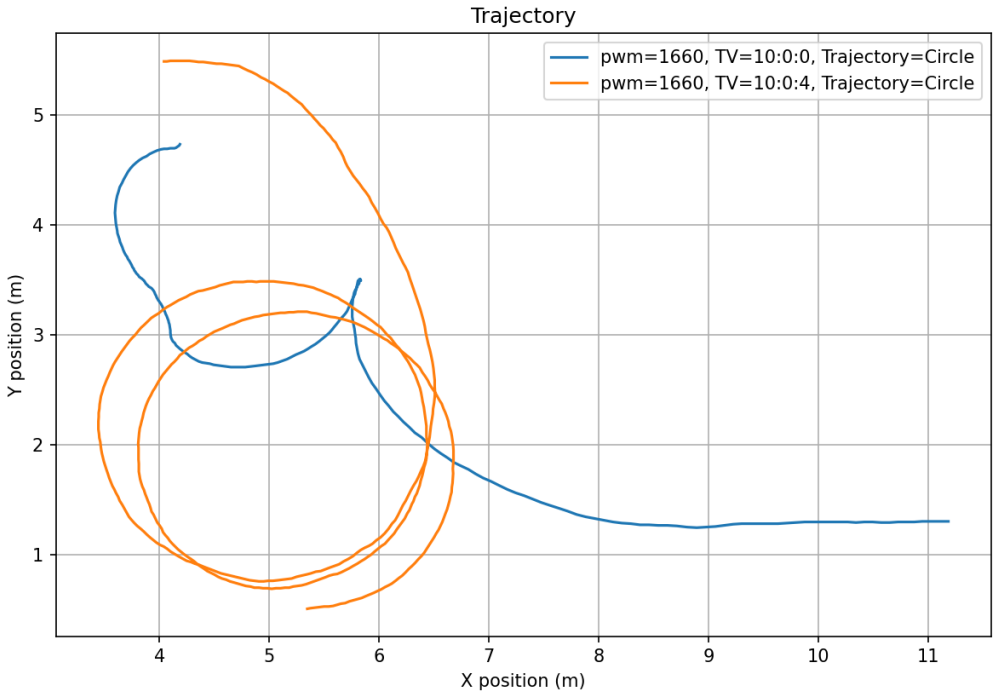


Figure 5.9: Trajectory of the car during the modified skidpad tests, shown on a grid, starting at the bottom right corner and accelerating before beginning the steering maneuver.

In Figure 5.10, the different speeds achieved by the car during the two tests are shown.

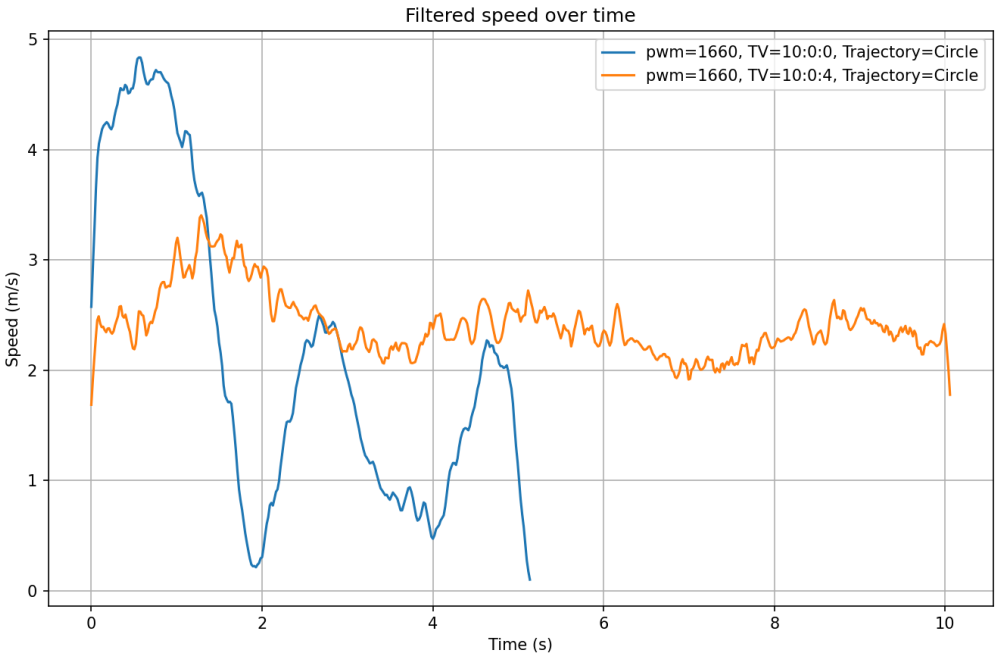


Figure 5.10: Speed of the car during modified skidpad tests.

6
Discussion

The results are evaluated in terms of performance and accuracy, while potential improvements and directions for future work are explored. In addition, cost and portability are assessed, followed by a discussion of relevant ethical and societal considerations.

6.1 Experimental test compared to simulation

The simulated moose test not only reveals the usefulness of the TV and how it distributes torque between the two wheels, but also tests how well the MPC works and how well it follows the reference path. Figure 5.1 shows how the PID regulates torque. However, the torque appeared to be in the lower ranges of only 0.10 Nm, which may have been caused by several factors, such as vehicle parameters not being accurately modeled or suboptimal PID tuning. Another visible feature in Figure 5.1 is that it was generated in discrete time compared to the other graphs. This is due to the fact that the PID in Matlab operates in discrete time, resulting in this behavior.

Comparing Figure 5.1 and Figure 5.2, it can be observed that the torque output generated by the PID controller increases when the rate of change in lateral position becomes more rapid. This is expected, since a faster change in lateral position indicates a sharper turning maneuver, where TV becomes more beneficial compared to a system without differential torque distribution. In these situations, the controller must generate larger corrective yaw moments to improve stability and trajectory tracking.

As discussed in Section 5.1, there is an approximate time difference of 0.75 s between the peak of the reference trajectory and the corresponding peak of the vehicle trajectory. This delay is likely caused by the MPC requiring time to react to rapid changes in the reference path. The faster the reference changes, the greater the temporary tracking delay becomes. However, the MPC gradually reduces this error over time, and at the minimum point the delay decreases to approximately 0.67 s. As the simulation approaches 10 s, the vehicle trajectory increasingly converges toward the reference trajectory as the lateral position changes become less aggressive.

A difference can also be observed in the maximum lateral displacement, where the reference trajectory reaches 3.56 m, while the vehicle only reaches 3.22 m. A likely explanation for this is the steering limitations imposed on the physical vehicle model. Since the steering angle is constrained, the vehicle cannot rotate as aggressively as the reference trajectory demands. This limits the achievable lateral displacement and may explain why the vehicle does not fully reach the same lateral position before returning toward the centerline.

One explanation for the unevenness of the generated path may be the characteristics of the Driving Scenario Designer in MATLAB. When designing the reference track, discrete points are manually placed to define the intended road geometry, after which the software generates a smooth interpolated path between these points. While this smoothing process simplifies path generation, it can also introduce small variations in curvature. To achieve specific corner radii, additional control points often had to be added, which may have unintentionally created minor irregularities in the resulting trajectory. These small inconsistencies likely contributed to the uneven shape of the simulated path.

The live experimental testing proceeded satisfactorily. However, in contrast to the simulation-based studies, the acquisition of quantitative data was more challenging. This limitation was primarily due to the fact that no systematic data-logging strategy had been defined in advance, and there was a lack of sufficient time to implement an appropriate logging solution. As a result, the most practical method for capturing performance data was to mark reference locations on the ground using paint or positioning cones at known distances, around which the vehicle could then be driven.

Due to limitations with the movement of the car and the repeatability of the experiment, the MPC was not precise enough due to drifting in the controller, and manual steering of the car would not be precise enough. This meant that it would not be possible to replicate the tests. Therefore, it was decided to have simpler paths than in the simulations, where the paths were created with the built in Driving Scenario Designer, allowing the road to be designed after the moose test.
In physical experiments, sensor configuration and inherent measurement errors must be taken into account. For example, the GY-BNO055 sensor, which incorporates an internal filtering algorithm, exhibits small variations in the measured acceleration on the order of 0.1 m/s^2. Over longer test durations, such deviations can accumulate and result in substantial overall measurement errors.

6.2 Effect of torque vectoring and MPC on performance

This section discusses the effects of TV and MPC on vehicle performance based on the results obtained during testing.

6.2.1 TV performance

A successful TV implementation should noticeably improve stability and/or handling capabilities. In Figure 5.7 the difference in trajectory while driving with the same steering angle is presented. The blue line, representing the test using TV, has a slightly larger turning radius than the scenario without TV. Seeing as the car was in a stable condition during the test without TV this is alarming, as it should have done nothing if the car was rotating appropriately, and even help turn the car quicker if there was under-rotation.

After taking a closer look at the code, it was apparent that the steering trim does not impact the steering angle from which the TV calculates $\dot{\psi}_{des}$, which is reasonable from the perspective of the original purpose of the steering trim function. This in turn means that the TV algorithm perceives the car as not steering, and subsequently decides that it should not rotate. This explains the understeer visible in Figure 5.7. Due to time constrains, there was no opportunity to patch the code and re-run the physical tests.

In Figure 5.8 it is hard to notice anything but a slight speed difference between the tests, which is more likely to be explained by the slightly larger radius than TV adding much greater cornering grip. This implies that the circular trajectory the car was on did not use the tires to it's fullest, and that it did not encounter sliding on either front or rear axle during the maneuvers in Figure 5.7. The difficulty encountered in attempting to test the car on the limit necessitated a more conservative testing approach, even though the impact from TV would be most obvious at the limit of grip.

In Figure 5.9, the trajectory that the car followed during manual driving with different TV PID-settings is shown. The main takeaway from this figure is that with a bit of derivative action, the car becomes much more controllable. Looking at the orange line, the car was easily drivable in circles as intended. Without derivative action, it became much harder to control. Since the transient from going straight to turning is quite violent, the derivative action made the car understeer initially at turn-in to aid in settling the car in the turn without a big moment of oversteer. Figure 5.10 suggests that the car without derivative action had a higher speed at turn-in, which could lead to instability. However, since the car still did not become stable after slowing down to below the speed of the derivative action test, it is most likely that the PID-tuning made the difference.

In theory, the implemented TV algorithm relies on estimating the added moment

M_z needed to either counter or enhance the rotation of the vehicle. In the implementation however, Equation 3.2 and the magnitude of M_z serve only to scale the PWM signal, and has therefore lost it's relation to physical quantities. The assumption was that the PID would provide adequately reliable control, without the need to map torque and PWM values to each other at different speeds. This logic turned out to be flawed since, as is discussed in Section 6.3.2 the ESCs reacted nonlinearly to PWM signals. This would make it difficult to tune the PID sufficiently, and make a compromise the best-case scenario. Mapping PWM signals to torque would solve this, and result in a more accurate and predictable TV scheme.

Another possible fault in the TV algorithm, is that the cornering stiffness of the tires is an approximation. Since no data was provided with the tires when purchased, a guess had to be made. Since the cornering stiffness is a big part of the TV calculations, a faulty approximation would result in a $\dot{\psi}_{des}$ calculation error, and therefore a faulty desired M_z. Since this would set the wrong target for TV control, and not impact the actual controller, it most likely did not impact the tests all to negatively. Future tests should of course strive to amend this issue.

The TV algorithm did succeeded in differentiating the amount of torque each rear wheel was given in a way that altered the handling characteristics of the vehicle. It is also apparent that without sufficient tuning and optimization, and care taken to counteract the nonlinear and unpredictable relation between PWM and torque, the TV algorithm can impact the handling characteristics in an unpredictable way. This could results in a car that is less stable, both for manual control and for the MPC.

6.2.2 Trajectory tracking and MPC

To determine the quality of the trajectory tracking with the greatest accuracy an onboard system for measuring and comparing the actual path to the desired path would need to be in place. Since that was not implemented a more rudimentary approach was used, namely the comparison between the observed steering trajectory and the radius of the circle which the MPC was given to follow. The observed steering trajectory was ascertained from the graphs produced by the testing videos discussed in Section 4.1. Evident in Figure 5.6 is that for both TV PID-settings in the compared experiments, the steering radius was significantly smaller than the 5 m that was input into the MPC.

It is also apparent that the different circles are not centered and not all that circular. This is mostly due to the fact that the car was not in a stable condition during the maneuvers, something that becomes apparent when looking at Figure 5.5. The car increases in speed and then rapidly loses most of it during a moment of severe oversteer, resulting in erratic movement of a periodic nature. Initially when the car starts turning a semblance of a large diameter circle appears, but as soon as the oversteer begins only smaller radius turns remain.

What is troubling is that the instability is present with TV enabled, and at a low speed. The MPC is set to believe that the car is traveling at 5 m/s, and this was not achieved even before initial turn-in. A lower speed should in a stable condition result in a smaller diameter circle, but not to the extent visible in Figure 5.6. The difference between the desired and actual speeds naturally creates a challenging situation for the trajectory following, and impedes it's ability to self-adjust, though in this cases the instability is likely the biggest culprit, as it enhances differences between the simple dynamic bicycle model and the real vehicle.

Several factors could be contributing to the instability, both in the mechanical system as in chassis and tire, and in the software system, such as PID-tuning and MPC/TV implementation. Seeing as the entire car was assembled and programmed for testing in a short time-span, and without the ability to slowly phase in and test systems individually in a whole-car context, it is difficult to determine what the main problem is.

The fact that the cornering stiffness of the tires was not accurately modeled, the MPC most likely responded believing it had a different level of grip than it actually had. This might have impacted the trajectory tracking capabilities, most likely however is that even with accurate cornering stiffness, the results of testing would be the same. This is because of the instability mentioned above.

Seeing as MPC is a computationally heavy control strategy, achieving MPC without fault on a microcontroller is an achievement. The TinyMPC library was used, which is optimized for lesser hardware, and it certainly yielded results. Only a small amount of the Teensy memory was used, so little in fact that lesser microcontrollers might have sufficed. There were also no signs of the car being slow to react because of needing to calculate. This implies that MPC is a viable control strategy for trajectory tracking using microcontrollers. It might even be possible to attempt with complete autonomous driving, with the caveat that increased sensor data and the onboard image processing often used for autonomous vehicles would add additional load to the microcontrollers that would need to be researched further.

6.2.3 Practical performance

The implementation of TV demonstrated slight but consistent improvements in vehicle stability and maneuverability during manual testing. Compared to driving without TV, the vehicle generally showed improved cornering behavior, reduced understeer, smoother yaw response, and better responsiveness during directional changes as well as straight-line driving. These improvements were most apparent during tests involving rapid steering transitions and sustained cornering, where differential torque distribution assisted in generating corrective yaw moments. With TV enabled, the vehicle was less prone to sliding and was better able to follow the intended path with a tighter and more controlled trajectory.

However, the overall difference between driving with and without TV was not large. The observed improvements were noticeable, but not dramatic, and the practical

performance was strongly affected by the tuning of the TV controller. The implemented TV system was based on a PID controller, but the tuning process was not fully completed before the final tests. Different PID settings had a noticeable effect on the vehicle behavior, especially during rapid transitions between turning and straight-line driving. Integral action was generally avoided, since it could have introduced a delayed response where the controller continues to request yaw correction even after the maneuver has ended. This could have caused the vehicle to continue turning after the desired path had changed.

Derivative action was also tested, but it sometimes produced aggressive motor behavior during sudden changes in the control error. For example, when the vehicle transitioned from cornering to driving straight, the rapid change in the error signal could cause one of the rear wheels to accelerate abruptly. This made the vehicle response less predictable and showed that the derivative term required more careful filtering and tuning. As a result, the final controller settings represented a compromise between responsiveness and stability rather than a fully optimized TV implementation.

Furthermore, the vehicle was tested at relatively low speeds and under practical constraints, meaning that it may not have reached conditions where the full stabilizing effect of TV becomes most apparent. Previous research shows that TV is especially relevant during transient or near-limit handling situations, where corrective yaw moments can significantly influence vehicle stability [6]. This also explains why the observed improvements were noticeable but not dramatic. Despite these limitations, the initial motor tests showed that the vehicle produced sufficient torque when the motors were operated at the highest speed setting. Under these conditions, the vehicle was able to start moving without being manually pushed. This indicated that using motors with adequate power was important for the evaluation of TV, since higher available torque made it easier to produce noticeable differences in vehicle behavior compared to a setup with weaker motors. However, the vehicle was still operated at lower speed settings during testing in order to maintain better control and reduce the risk of instability. This also reduced the available driving force, which made the vehicle weaker during startup. As a result, the vehicle often required a manual push to begin moving from a standstill.

The results can therefore be interpreted as a proof of concept rather than a complete performance validation. Although the improvement was moderate, the repeated trend across the practical tests indicates that TV had a positive effect on the vehicle's controllability. At the same time, the results show that further PID tuning, implementation of slip-angle regulation, torque mapping, and more systematic controller validation would likely be required to achieve more consistent and clearly measurable improvements. Overall, TV improved the vehicle performance in a noticeable but not transformative manner.

6.3 Electrical and mechanical hardware

This section discusses the limitations of the electrical and mechanical systems and their impact on the vehicle performance.

6.3.1 Motors

One limitation observed during testing was the difficulty of regulating the motor speed at low RPM using the Arduino-based control. The motors could not be reliably operated below approximately 10% of their maximum rotational speed. The motors have a rating of 3900 KV and were powered by a 3S LiPo battery with a nominal voltage of 11.1 V. The maximum theoretical RPM can be estimated from the relationship between KV and voltage, and 10% of this value is then determined:

$$\mathrm{RPM}_{\max} = 3900 \cdot 11.1 = 43\,290\,\mathrm{RPM}$$

$$0.10 \cdot 43\,290 = 4\,329\,\mathrm{RPM}$$

Thus, 10% corresponds to approximately 4 329 RPM. It should be noted that these values represent no-load conditions, meaning that the motors are spinning freely without any mechanical load. In practical operation, the achievable RPM will be lower due to mechanical and electrical losses.

After further adjustments to the control software, it was possible to operate the motors below this initial 10% threshold. However, the primary limitation of the motors was their low torque output. Although the motors were capable of achieving high RPM, which is beneficial for reaching speeds where differences between operation with and without TV can be observed, the lack of torque made it difficult to drive the vehicle when the motors were directly mounted to the wheels.

This issue was addressed by introducing a gear reduction system, as discussed in Section 6.4. While this enabled the vehicle to become drivable, it was still often necessary to apply a manual push to initiate motion, as the motors struggled to overcome the static resistance from a standstill even at full throttle.

6.3.2 ESC

The ESC exhibited a limitation during driving. The vehicle could be driven twice consecutively, but after the second stop, the ESC required a reset before further operation was possible. This suggests that the ESC entered a protection or fault state (e.g., due to voltage, current, or thermal limits), requiring a restart to resume normal operation. An additional observation was that the ESC did not respond linearly to the PWM input signal. While the system operated appropriately at a PWM value of 1650, however the response beyond this point became highly irregular and unpredictable. This non linear behavior suggests that the ESC may enter a different operating scale, which significantly affects controllability. This phenomenon should be noted, as it may affect both performance consistency and the reliability of control strategies that rely on proportional PWM input.

During the constant angle test using the steering trim function, an attempt was made to increase the velocity of the vehicle to see where the threshold for skidding would be. A slight increment from 1650 to 1652 PWM changed the car's behavior completely, it was unable to keep the same circular radius and did not skid 5.4. Instead of skidding, the car was unable to follow the intended drive path due to the inconsistent velocity of the car which is unusual given that the throttle is at maxed settings in both test at different PWM. As illustrated in 5.3, the blue graph show a relative consistent velocity at full throttle settings while the PWM input is 1650, however this is not the case when the same conditions are given for when the input is 1652. The orange graph shows a oscillating behavior which is abnormal and has no logical explanation to why the velocity of the car is not linear to the PWM input.

6.3.3 IMU and microcontrollers

Throughout the experimental phase, issues were encountered with the registration and logging of data from the IMU. These problems affected the consistency and availability of sensor measurements used for state estimation. As a result, the reliability of the collected data for the whole thesis may have been compromised, which could have influenced the accuracy of the results and the overall evaluation of the system performance.

An issue that did not appear in the simulations was that the IMU failed to operate when connected to the Teensy 4.1. The exact cause could not be determined, and it remained unclear whether the problem originated from software or hardware limitations. To resolve this, the IMU was instead connected to the ESP32, where it had previously been verified to function correctly.

This modification introduced a drawback, as communication between the ESP32 and the Teensy 4.1 had to be handled via serial communication. Compared to a direct connection, this approach introduced additional latency and complexity in the data transfer. However, experimental testing showed that, although suboptimal, the solution was sufficiently reliable. The added delay did not appear to have a significant impact on the performance of either the MPC or the TV algorithms.

6.3.4 Pre-manufactured mechanical components

The mechanical design introduced several limitations that affected the performance and repeatability of the tests. The vehicle was based on a prebuilt RC chassis, which provided a practical starting point since it already included a base plate, steering assembly, drivetrain components, and wheel suspension. This reduced the amount of custom manufacturing required and made it possible to focus on the TV drivetrain. However, using a prebuilt chassis also limited the control over some important mechanical properties, especially the steering geometry and the stiffness of several components.

The original plan was to use three independently driven motors, with one motor

driving the front axle and two motors driving the rear wheels. This would have required a front differential and a method for transferring torque through joints while still allowing steering motion. Since these components were expensive to purchase separately, the prebuilt chassis was selected as a compromise. After one of the ESCs failed during testing, the final vehicle configuration was instead limited to two rear motors. This simplified the mechanical layout but also reduced the available TV capability compared to the original concept.

One of the main limitations of the chassis was the steering mechanism. The steering assembly showed noticeable play and limited precision, which made the vehicle difficult to align and caused deviations during straight-line driving. Since the steering system was inherited from the prebuilt chassis, the exact relationship between servo input and wheel angle was not fully controlled. This reduced the accuracy of the MPC implementation, since the controller assumed a predictable steering response. During testing, part of the steering mechanism also failed and had to be repaired. Although the repair allowed testing to continue, it further reduced the mechanical precision and repeatability of the system.

The aluminium base plate of the chassis performed well and provided sufficient structural stiffness for the scale vehicle. The stiffness of the base plate meant that the custom rear drivetrain components did not need to provide significant additional chassis rigidity. The rear section of the original chassis also offered several mounting options, which simplified the integration of the custom motor mount and drivetrain assembly. However, the overall build quality of the prebuilt chassis was limited, and some components were not designed for the loads and modifications introduced by the custom TV system.

The wheels provided sufficient grip under clean surface conditions, but their performance changed significantly during testing. The rubber tires quickly collected dust and small particles, which reduced traction and made the vehicle more prone to sliding. This made indoor testing difficult, as the tires both lost grip and left black marks on the floor. As a result, most practical testing had to be performed outdoors. Outdoor testing provided better conditions for the tires, but it also introduced surface irregularities that affected the vehicle motion.

6.3.5 Wiring, connecting and testing

To verify the functionality of the implemented hardware, the system components were tested both individually and in combination. The initial tests focused on confirming that each component could be controlled separately before integrating them into the complete vehicle system. This included testing the IMU, servo, ESCs, motors, and communication between the microcontrollers.

Several issues were encountered during the hardware integration process. One of the first problems involved communication with the GY-BNO055 IMU. The sensor was intended to communicate with the ESP32 and Teensy 4.1 using I2C, but com-

munication could not initially be established. This was later found to be caused by the IMU module requiring onboard soldering to enable I2C communication. After the required solder connection was completed, the sensor could communicate with the microcontroller as intended.

An additional issue occurred during ESC control. Early testing was performed using the ESP32, which made stable PWM signal generation more difficult compared to the Teensy 4.1. This complicated the control of the ESCs and made the motor response less reliable during initial testing. In addition, the requirement for a shared common ground between the system components was initially overlooked. Once a common ground was implemented, the ESC control became more stable.

After the individual components had been tested, the servo and IMU were connected to evaluate basic sensor-actuator interaction. A simple control program was implemented so that rotation of the IMU around one axis generated a corresponding movement of the servo. This test confirmed that sensor data could be read and used to control an actuator in real time.

Further testing was then performed with the motors and ESCs to evaluate motor response, startup behavior, and speed control. These tests showed that the motors were able to reach high rotational speeds, but startup behavior was not always reliable. In some cases, the motors produced audible electrical activity and attempted to rotate, but did not start properly. This indicated that the motor and ESC setup required further tuning and careful handling during arming and startup.

When the system was later assembled into the vehicle, the amount of wiring and the interaction between components became more complex. The complete setup required reliable power distribution, stable signal connections, and correct communication between the ESP32 and Teensy 4.1. The integration tests therefore showed that the individual components functioned as intended, but also that the complete system was sensitive to wiring, grounding, and timing-related issues.

6.4 3D-printed mechanical components

The 3D-printed components generally performed better than expected. The printed axles and adapters were able to withstand the applied loads during most tests, and the custom rear drivetrain assembly provided the required functionality. One of the rear drivetrain components broke during a crash, but this was considered an acceptable failure mode since it may have prevented damage to more expensive components such as the motors or chassis.

Material selection also introduced limitations. In the initial design, both the axles and surrounding drivetrain components were printed in PLA. During testing, friction between moving parts generated heat, which caused some components to soften and partially melt together. To reduce this problem, the main rear drivetrain component was later printed in PETG. PETG was selected due to its improved heat resistance compared to PLA, while PLA was retained for components where stiffness was more important than thermal resistance. This material combination improved the reliability of the drivetrain but did not fully eliminate friction related losses.

For simplicity, the suspension was made completely rigid by replacing the original springs and dampers with stiff components. This simplified the mechanical modeling and reduced suspension-induced dynamics. However, the rigid suspension also introduced practical limitations during outdoor testing. Small bumps or uneven surfaces caused the entire vehicle to move vertically which reduced the normal force on some wheels and therefore decreased traction. This affected cornering behavior, especially during dynamic maneuvers. The foam inside the tires provided a small amount of compliance, which partially compensated for the lack of suspension, but it was not sufficient to fully absorb surface irregularities.

The lack of suspension also compromises the wheel load distribution. Suspension systems, such as those in regular cars, have static sag which helps keep the normal force on the tires more evenly distributed during rapid lateral movements. Since a stiff suspension was applied, severe maneuvers where the car was prone to tilting, were harder to control.

Overall, the mechanical platform was sufficient for demonstrating the basic function of the TV system, but it also imposed several important limitations. Steering play, drivetrain friction, tire sensitivity, limited suspension compliance, and the durability of 3D-printed parts all affected the repeatability and accuracy of the practical tests. These limitations should be considered when interpreting the experimental results, since they may have contributed to deviations between the simulated and measured vehicle behavior.

6.5 Model assumptions compared to reality

The control system in this thesis is based on simplified vehicle models. These models are mainly used to describe vehicle behavior and to design the controller. The MPC controller uses the linearized dynamic bicycle model where the left and right wheels on each axle are combined into one front wheel and one rear wheel. This makes the model easier to use for real-time control. However, the real test vehicle does not fully match this model. The platform uses two driven motors rather than four independently driven wheels. It also has real steering mechanics and sensor limitations. Because of this, the real vehicle will not behave exactly as the model predicts.

One important assumption is that the tires behave linearly. This means that the lateral tire force is assumed to increase in a predictable way when the slip angle increases. In reality, this is only true for small steering angles and calm maneuvers. During sharp turns or sudden direction changes, the tires may reach their grip limit. This can cause understeer or oversteer. As a result, the controller may not provide the same stabilizing effect in real tests as expected from the theoretical model.

Limited TV capability also plays a factor. In an ideal four-motor vehicle, torque can be controlled independently at each wheel. In this platform, only two driven motors are used. This limits how much yaw moment can be generated through torque distribution. The controller, therefore, has less freedom than assumed in the theoretical model. This can reduce the effect of TV during more demanding maneuvers.

Another simplification is that the vehicle speed is assumed to be constant. In reality, the speed will change during driving. This can happen because of acceleration, braking, motor response, rolling resistance, and battery voltage drops. Since the bicycle model depends on vehicle speed, these changes can reduce the accuracy of the predicted motion. This may lead to path-tracking errors if the controller uses a fixed speed value while the actual vehicle speed changes.

The model also assumes that control inputs are applied instantly and accurately. In the real system there will be delays. The microcontroller needs time to calculate the control signal. The motor controllers need time to process the command. The motors also need time to change torque. The actual torque may differ from the commanded torque because of current limits, motor behavior, battery voltage drops and mechanical losses. These effects can influence the performance of the TV system.

Sensor measurements create another difference between the model and reality. The model can describe values such as position, yaw rate, velocity and steering angle. In physical testing these values must be estimated using available sensors such as an IMU. Since no wheel encoders are used, wheel speed and traveled distance cannot be measured directly. This makes state estimation less accurate and increases the risk of drift over time. Noise, bias and limited sensor resolution can also affect the estimated vehicle motion. These errors can reduce the accuracy of practical testing.

Mechanical and environmental factors can also affect the real vehicle. Steering backlash, wheel alignment errors, chassis flexibility, vibrations, tire wear, uneven ground and changes in surface friction may influence the results. These effects are difficult to include accurately in the mathematical model. For this reason the real vehicle should not be expected to match the model perfectly. Instead, the comparison between the model and real testing can show how well the simplified model represents the most important vehicle behavior. It can also show which assumptions have the greatest effect on performance.

The small-scale platform is useful for testing the basic control concept, but the results cannot be transferred directly to a full-size vehicle. Many of the differences mentioned in Section 6.5 also become scalability issues. Tire behavior, vehicle mass, yaw inertia, actuator response, sensor accuracy and the use of only two driven motors all affect how well the results represent a real vehicle. These factors do not scale in a simple way and would therefore need to be reconsidered for a full-size application.

A full-size vehicle would also place higher demands on the hardware and control system. The motors would need to handle much larger forces and the motor controllers would need to manage higher power levels. The sensors would also need higher accuracy and reliability, especially because small errors can have larger safety consequences at real vehicle speeds. The controller would therefore need to be tested and tuned again with full-scale vehicle parameters. The test environment is another scalability issue. The small-scale platform can be tested under controlled conditions, while a real vehicle must work on different road surfaces, at different speeds and in changing weather conditions. This means that the small-scale results should mainly be seen as a proof of concept.

6.6 Society, ethics and ecological perspective

TV is also societally relevant in relation to Sweden's Vision Zero (Nollvisionen) which aims to completely eliminate fatalities and serious injuries in road traffic[40]. By improving vehicle stability and controllability during critical driving situations TV reduces the risk of loss of control accidents and supports the Vision Zero principle of designing transport systems that mitigate the consequences of human error. As the technology can be applied to different vehicle types including public transport vehicles it has the potential to reduce accident risk in situations involving larger groups of passengers. In dense urban environments improved vehicle control can also help reduce the risk of vehicle to pedestrian accidents where precise and predictable motion is particularly important.

Ethical considerations are a significant aspect of implementing TV due to the system directly affecting vehicle stability and safety through automated control of wheel torque. Design errors or incorrect system behavior may have immediate safety consequences. This creates a strong ethical responsibility to ensure robust performance and predictable behavior under a wide range of operating conditions.

In addition, the largely invisible operation of TV raises concerns related to transparency and accountability. Since drivers may be unaware when the system intervenes, responsibility in the event of a failure can be difficult to determine. This highlights the importance of clear documentation, explainable control strategies, and thorough validation. Ethical significance also arises from the risk of driver over reliance, as improved handling performance may encourage operation closer to vehicle limits if system limitations are not clearly understood. For these reasons, ethical considerations must be integrated into both the design and evaluation of TV systems rather than treated as secondary to technical performance.

From an ecological perspective, TV in electric vehicles can have both benefits and drawbacks. Improved stability and traction can reduce energy losses and unnecessary braking, which may lower overall energy use [41]. By supporting safer and more efficient electric vehicles, the technology can also contribute to reduced greenhouse gas emissions compared to fossil fuel vehicles. At the same time TV increases system complexity. Additional motors, sensors, electronics and batteries require more materials and energy during production. Resource extraction for battery manufacturing and electronic waste also create environmental impacts. The overall ecological effect should therefore be assessed from a life cycle perspective, including production use and end of life management.

7
Conclusion

The aim was to design, implement, and experimentally evaluate a small-scale electric vehicle platform with TV and MPC-based trajectory tracking. The final system should primarily be seen as a proof of concept rather than a fully validated vehicle control platform.

- The first objective was partly met. The final vehicle used two independently driven rear motors and the TV algorithm was able to distribute different PWM commands to the rear wheels. In the modified skidpad test, the vehicle became more drivable when derivative action was added to the TV controller, using PID settings of $10, 0, 4$, compared with proportional-only control at $10, 0, 0$.

  These results objective were strongly dependent on controller tuning. Since the final TV algorithm has no base in physical quantities and the ESCs respond nonlinearly to PWM commands, the implemented TV system did not always improve behavior, and it is uncertain whether optimal tuning would provide adequate stability for a multitude of conditions. In the constant steering angle test, a steering-trim coding issue caused the TV algorithm to calculate the desired yaw rate from an incorrect steering input. The approximation of the cornering stiffness also presents possible faults. Therefore, the TV objective was met in terms of implementation, but only partly met in terms of reliable performance improvement.

- The second was partly met. The MPC was successfully implemented on the Teensy 4.1 using TinyMPC, and no clear computational limitation was observed during testing. In simulation, the MPC followed the modified moose-test reference path with a delay. In physical testing the MPC did not track the intended circular reference accurately. The commanded circular path had a radius of 5 m, but the measured trajectory had a much smaller radius and was not stable. The MPC was therefore successfully implemented, but the physical trajectory tracking objective was only partly achieved.

- The third objective was met. The final platform had a mass of 2.33 kg, used a 3S Li-Po battery with a nominal voltage of 11.1 V and included two brushless motors, two ESCs, an IMU, a ESP32, and a Teensy 4.1. The platform was therefore within the defined mass, voltage, hardware, and budget constraints. The motors had high speed capability but low starting torque, which meant that the vehicle often needed a manual push to begin moving. The gear ratio

of 1:3 improved drivability, but did not fully solve the startup torque limitation. The ESCs also showed nonlinear behavior, especially around PWM values above 1650, where a small increase from 1650 to 1652 caused a large and inconsistent change in vehicle behavior.

- The fourth objective was partly met. The IMU was successfully integrated and used to provide orientation data for the control system. The IMU could not be connected directly to the Teensy as intended and was instead connected through the ESP32, which added communication complexity. The project also lacked a onboard data logging system. As a result, vehicle position and speed were mainly extracted from video recordings using OpenCV, while several internal signals, such as exact PWM distribution, steering input, and state estimation errors, were not logged during the runs. This limited the quantitative evaluation of both TV and MPC performance.

The final result should therefore be described as a partially successful proof of concept, with clear hardware, software, and testing limitations that must be addressed before stronger conclusions can be drawn about the performance benefits of TV and MPC trajectory tracking.

8

For future testing

The next steps would be to correct the steering trim issue and implement closed-loop speed control. This can be done using wheel encoders, GPS, or another reliable velocity measurement tool. This is especially important since the MPC assumes a constant longitudinal velocity, while the practical tests showed that using only a fixed PWM command resulted in unstable speed under load, particularly during turning. A stable speed would therefore improve the reliability and repeatability of future tests.

Further development should also focus on improving the mechanical and control systems to enable drawing of stronger conclusions about the performance benefits of TV and MPC. The gearing ratio could be increased, or motors with higher torque could be selected, so that the vehicle can start from a standstill more reliably and be easier to control at low speeds. The steering system should also be improved to ensure accurate and centered steering, either by using a servo with position feedback or by adapting the control code more precisely to the existing servo. Adding suspension would make the platform more representative of a real vehicle and allow future studies to investigate how load transfer affects TV performance during severe maneuvers and loss of grip. The MPC and TV systems would need to be tested and tuned separately before being combined. This would make it easier to verify the functionality of each controller and identify whether instability originates from the MPC, the TV controller, the hardware, or their interaction. Accurate modeling of the cornering stiffness is needed to guarantee both MPC and TV work accurately. PWM to torque mapping would also need to be implemented for the TV system to be accurate and consistent. Slip-angle regulation should also be implemented and evaluated. The remote controller could also be improved by optimizing the phone interface and adding haptic feedback, which could give the driver better control and improve repeatability during manual testing.

Finally, future testing would benefit from onboard data logging. The system should store IMU data, velocity, steering angle, PWM commands, estimated states, torque distribution commands, and controller outputs during each run. This would make it possible to compare test results more quantitatively and determine whether the observed handling improvements are repeatable and caused by the control algorithms rather than test variation. For a more advanced autonomous driving platform, additional sensors such as cameras or LiDAR could also be added, although this would require further investigation of the computational load on the microcontrollers.

9
Usage of artificial intelligence

AI-based tools, primarily large language models such as ChatGPT and Gemini were utilized according to the Chalmers guidelines regarding usage of AI-tools. These tools were utilized for grammar checking, improving sentence structure, and refining academic language formulations in the final report. AI was also used to assist in understanding portions of existing code, debugging programming errors, generating code for data visualization and graph creation, and exploring possible implementation approaches during development.

Bibliography

[1] Mercury, "Moose test.svg," Wikimedia Commons, 2008, public domain. Accessed: 2026-05-05. [Online]. Available: https://sv.wikipedia.org/wiki/Fil:Moose_test.svg

[2] M. Asperti, M. Vignati, and E. Sabbioni, "On torque vectoring control: Review and comparison of state-of-the-art approaches," *Machines*, vol. 12, no. 3, p. 160, 2024. [Online]. Available: https://www.mdpi.com/2075-1702/12/3/160

[3] J. Nah and S. Yim, "Vehicle dynamic control with 4ws, esc and tvd under constraint on front slip angles," *Energies*, vol. 14, no. 19, p. 6306, 2021. [Online]. Available: https://www.mdpi.com/1996-1073/14/19/6306

[4] P. Falcone, F. Borrelli, J. Asgari, H. P. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 3, pp. 566–580, 2007. [Online]. Available: https://ieeexplore.ieee.org/document/4162483

[5] J. Y. Park, S. Na, H. Cha, and K. Yi, "Direct yaw moment control with 4wd torque-vectoring for vehicle handling stability and agility," *International Journal of Automotive Technology*, vol. 23, no. 3, pp. 555–565, 2022.

[6] K. Jalali, L. Sorniotti, A. D. Novellis, and P. Gruber, "Comparison of feedback control techniques for torque-vectoring control of fully electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 8, pp. 3612–3623, 2014.

[7] L. D. Novellis, A. Sorniotti, and P. Gruber, "Wheel torque distribution criteria for electric vehicles with torque-vectoring differentials," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 4, pp. 1593–1602, 2014.

[8] Z. Zhou and Y. Bao, "Path tracking control of autonomous vehicle based on MPC," in *2024 3rd International Conference on Energy and Power Engineering, Control Engineering (EPECE)*, Feb. 2024, pp. 211–216, accessed: 2026-05-13. [Online]. Available: https://ieeexplore.ieee.org/document/10624728/authors#authors

[9] L. Schulze, D. W. Bertol, and R. Sebem, "Conventional and Explicit MPC Applied to Robotic Systems: a Computational Cost Evaluation," in *2021 29th Mediterranean Conference on Control and Automation (MED)*, Jun. 2021, pp. 861–866, iSSN: 2473-3504. [Online]. Available: https://ieeexplore.ieee.org/document/9480185

[10] Infineon Technologies. (2026) What is a microcontroller? Accessed: 2026-03-06. [Online]. Available: https://www.infineon.com/technology/what-is-microcontroller

[11] IBM. (2026) What is flash memory? Accessed: 2026-03-06. [Online]. Available: https://www.ibm.com/think/topics/flash-memory

[12] J. Zhang, Q. Fan, M. Wang, B. Zhang, and Y. Chen, "Robust speed tracking control for future electric vehicles under network-induced delay and road slope variation," *Sensors (Basel)*, vol. 22, no. 5, p. 1787, Feb 2022.

[13] R. Rajamani, *Vehicle Dynamics and Control*, 2nd ed. New York: Springer, 2012.

[14] M. Belrzaeg, A. A. Ahmed, A. Q. Almabrouk, M. M. Khaleel, A. A. Ahmed, and M. Almukhtar, "Vehicle dynamics and tire models: An overview," *World Journal of Advanced Research and Reviews*, vol. 12, no. 1, pp. 331–348, 2021. [Online]. Available: https://wjarr.com/content/vehicle-dynamics-and-tire-models-overview

[15] J. Gomez Fernandez, "A vehicle dynamics model for driving simulators," Master's thesis, Chalmers University, https://publications.lib.chalmers.se/records/fulltext/160346.pdf.

[16] A. Mistri, "Planar vehicle dynamics using bicycle model," in *Advances in Engineering Design*, ser. Lecture Notes in Mechanical Engineering, A. Prasad, S. Gupta, and R. Tyagi, Eds. Singapore: Springer, 2019.

[17] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Madison, WI: Nob Hill Publishing, 2017.

[18] MathWorks. (2026) Explicit mpc controller. Accessed: 2026-03-06. [Online]. Available: https://www.mathworks.com/help/mpc/ref/explicitmpccontroller.html

[19] Espressif Systems, "Esp32-wroom-32e esp32-wroom-32ue datasheet version 2.0," 2025, accessed: 2026-05-11. [Online]. Available: https://documentation.espressif.com/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf

[20] NXP Semiconductors, "i.mx rt1060 crossover processors for consumer products," 2025, accessed: 2026-05-11. [Online]. Available: https://www.nxp.com/docs/en/nxp/data-sheets/IMXRT1060CEC.pdf

[21] Hobbex. (2026) Modr borstlöst fartreglage 45a. Accessed: 2026-05-01. [Online]. Available: https://www.hobbex.se/modr-borstlost-fartreglage-45a-4-0mm.html

[22] ——. (2026) Modr borstlös motor 3650 3900kv sensorlös. Accessed: 2026-05-01. [Online]. Available: https://www.hobbex.se/modr-brushless-motor-3650-3900kv-sensorless-4-0mm.html

[23] ——. (2026) Modr li-po 11,1v (3s) 3000mah 30c – xt60-kontakt. Accessed: 2026-05-01. [Online]. Available: https://www.hobbex.se/modr-li-po-11-1v-3s-3000mah.html

[24] Mathworks. (2026) Matlab. Accessed: 2026-05-06. [Online]. Available: https://se.mathworks.com/products/matlab.html

[25] PlatformIO. (2026) Platformio. Accessed: 2026-03-29. [Online]. Available: https://platformio.org/

[26] J. P. Prusa Research, "Prusaslicer," accessed: 2026-05-11. [Online]. Available: https://www.prusa3d.com/p/prusaslicer/

[27] OpenCV. (2025) cv::VideoCapture Class Reference. 2026-05-05. [Online]. Available: https://docs.opencv.org/3.4/d8/dfe/classcv_1_1VideoCapture.html

[28] ——. (2025) Geometric Transformations of Images. Accessed: 2026-05-

05. [Online]. Available: https://docs.opencv.org/3.4/da/d6e/tutorial_py_geometric_transformations.html

[29] ——. (2026) cv::TrackerCSRT Class Reference. 2026-05-05. [Online]. Available: https://docs.opencv.org/4.x/d2/da2/classcv_1_1TrackerCSRT.html

[30] K. Nguyen, S. Schoedel, A. Alavilli, B. Plancher, and Z. Manchester, "Tinympc: Model-predictive control on resource-constrained microcontrollers," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10610987

[31] Adafruit Industries. (2026) adafruit/adafruit unified sensor. Accessed: 2026-04-29. [Online]. Available: https://registry.platformio.org/libraries/adafruit/Adafruit%20Unified%20Sensor

[32] Arduino, "Servo," 2023, accessed: 2026-05-12. [Online]. Available: https://github.com/arduino-libraries/Servo

[33] S. Monk, *Programming Arduino Next Steps: Going Further with Sketches*, 2nd ed. McGraw-Hill Education, 2019.

[34] M. A. P. Martin and S. Herman, "Two-wire bus system with a clock line wire and a data line wire for interconnecting a number of stations," Patent EP0 051 332A1, 1982, accessed: 2026-04-29. [Online]. Available: https://worldwide.espacenet.com/patent/search/family/019836092/publication/EP0051332A1?q=EP0051332A1

[35] M. Grahovic and M. Rosicki, "Development and evaluation of a torque-vectoring algorithm on rwd racing cars using a dual clutch," Master's thesis, Lunds university, Department of Automatic Control, 2019. [Online]. Available: https://lup.lub.lu.se/luur/download?func=downloadFile&recordOId=8987522&fileOId=8987524

[36] Reely, "1:10 rc modellbil elektrisk gatumodell onroad chassis fyrhjulsdrift 4wd arr tc-04," 2026, accessed: 2026-04-26. [Online]. Available: https://rb.gy/3f6u9d

[37] H. Gonabadi, A. Yadav, and S. J. Bull, "The effect of processing parameters on the mechanical characteristics of pla produced by a 3d fff printer," *The International Journal of Advanced Manufacturing Technology*, vol. 111, no. 3, pp. 695–709. [Online]. Available: https://doi.org/10.1007/s00170-020-06138-4

[38] International Organization for Standardization, *Passenger cars – Test track for a severe lane-change manoeuvre – Part 2: Obstacle avoidance*, Std. ISO 3888-2:2025, 2011.

[39] M. Balena, G. Mantriota, and G. Reina, "Dynamic handling characterization and set-up optimization for a formula sae race car via multi-body simulation," *Machines*, vol. 9, no. 6, p. 126, 2021. [Online]. Available: https://www.mdpi.com/2075-1702/9/6/126

[40] Trafikverket, "Nollvisionen – tillsammans räddar vi liv," https://bransch.trafikverket.se/for-dig-i-branschen/samarbete-med-branschen/Samarbeten-for-trafiksakerhet/tillsammans-for-nollvisionen/, 2024.

[41] R. Jafari, S. S. Refaat, A. Paykani, P. Asef, and P. Sarhadi, "Reinforcement learning for torque vectoring in electric vehicles: A review of stability and energy optimization methods," *IEEE Open Journal of Vehicular Technology*, vol. 7, pp. 354–380, 2025.

A

TinyMPC

MATLAB code to export the desired MPC to TinyMPC.

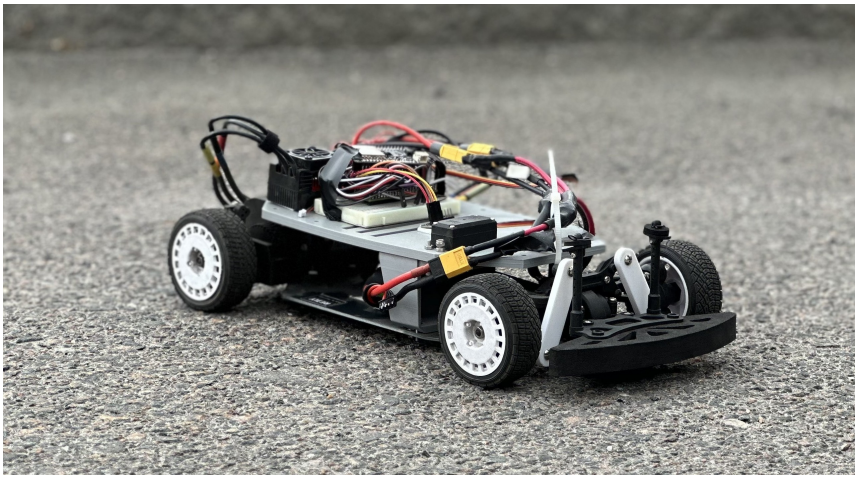
```
addpath("src");
addpath("build");

% physical parameters of the RC car
m = 2.33;
lf = 0.14728;
lr = 0.11572;
Caf = 5000;
Car = 5000;
Vx = 5;
L = lf + lr;
W = 0.195;

% Yaw moment of inertia
Iz = (m / 12) * (L^2 + W^2);



% Vehicle state matrixes
A = [0, 1, 0, 0;
     0, -(2*Caf + 2*Car)/(m*Vx), 0, -Vx-(2*Caf*lf - 2*Car*lr)/(m*Vx);
     0, 0, 0, 1;
     0, -(2*lf*Caf - 2*lr*Car)/(Iz*Vx), 0, -(2*lf^2*Caf + 2*lr^2*Car)/(Iz*Vx)];

% Input matrices
B_1 = [0; 2*Caf/m; 0; 2*Caf*lf/Iz];
B_2 = [0; 0; 0; 1/Iz];

% Combined input matrix
B_ny = [B_1, B_2];

Ts = 0.1;

% Continuous state-space convert discrete time
sys_c = ss(A, B_1, eye(4), zeros(4,1));
sys_d = c2d(sys_c, Ts);
```

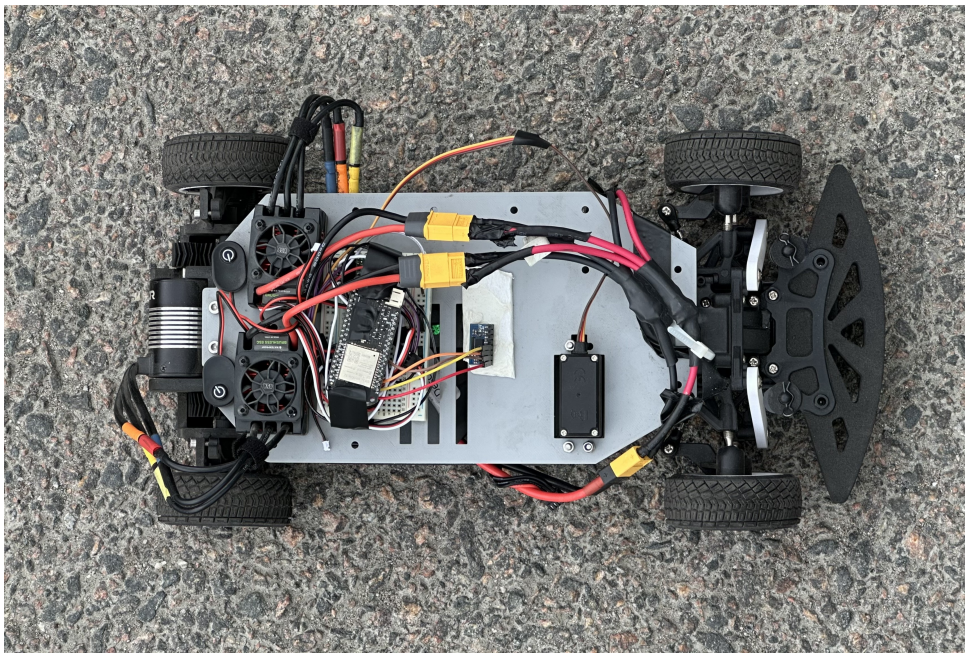
```
Ad = sys_d.A;
Bd = sys_d.B;

% MPC tuning and constraints
Q = diag([100, 1, 10, 1]);

% Input weight matrix
R = 0.1;

% Prediction horizon
N = 20;


% Input constraints
u_min = [-30*(pi/180)];
u_max = [ 30*(pi/180)];

% Generate code
solver = TinyMPC();


solver.setup(Ad, Bd, Q, R, N, "u_min", u_min, "u_max", u_max);

% Export code
solver.codegen("tinympc_esp32_code");
disp("Code exported");
}
```

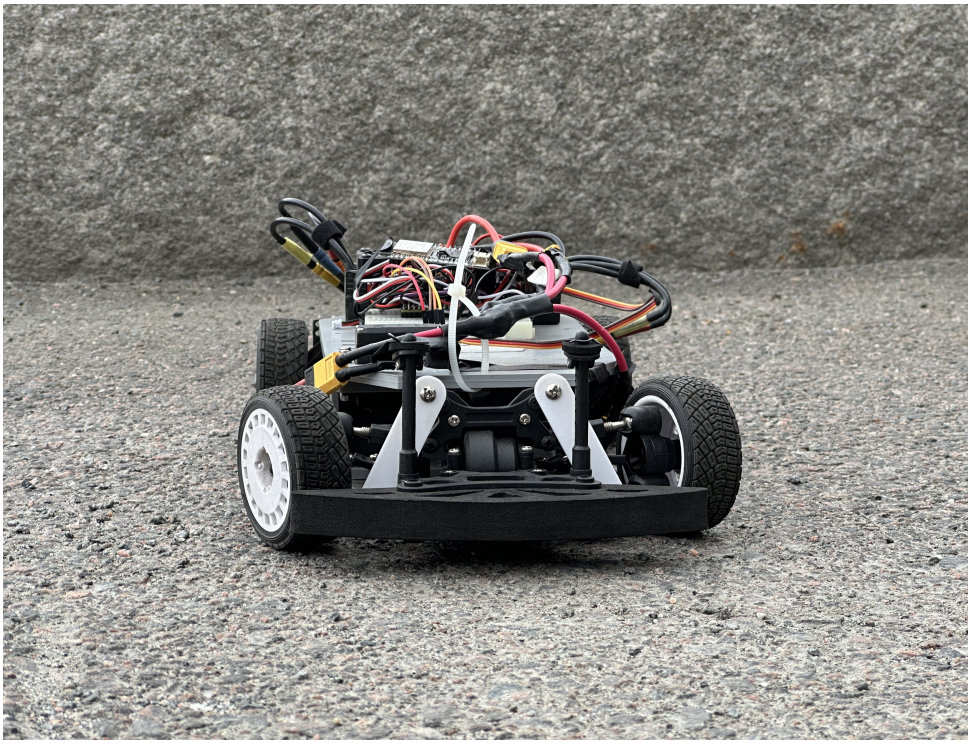
B

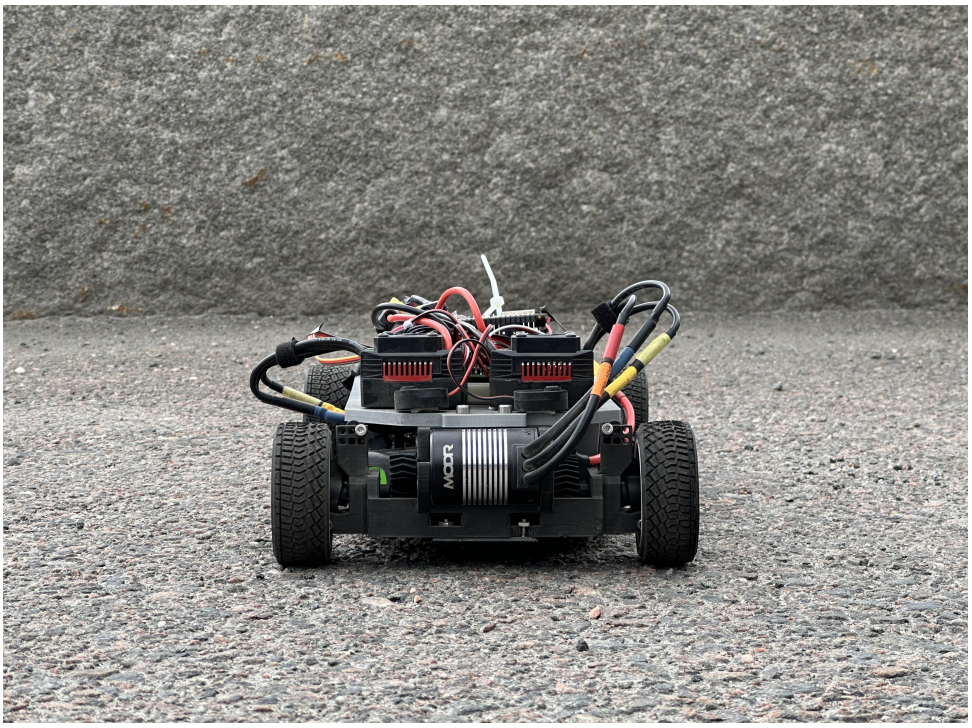
Vehicle platform

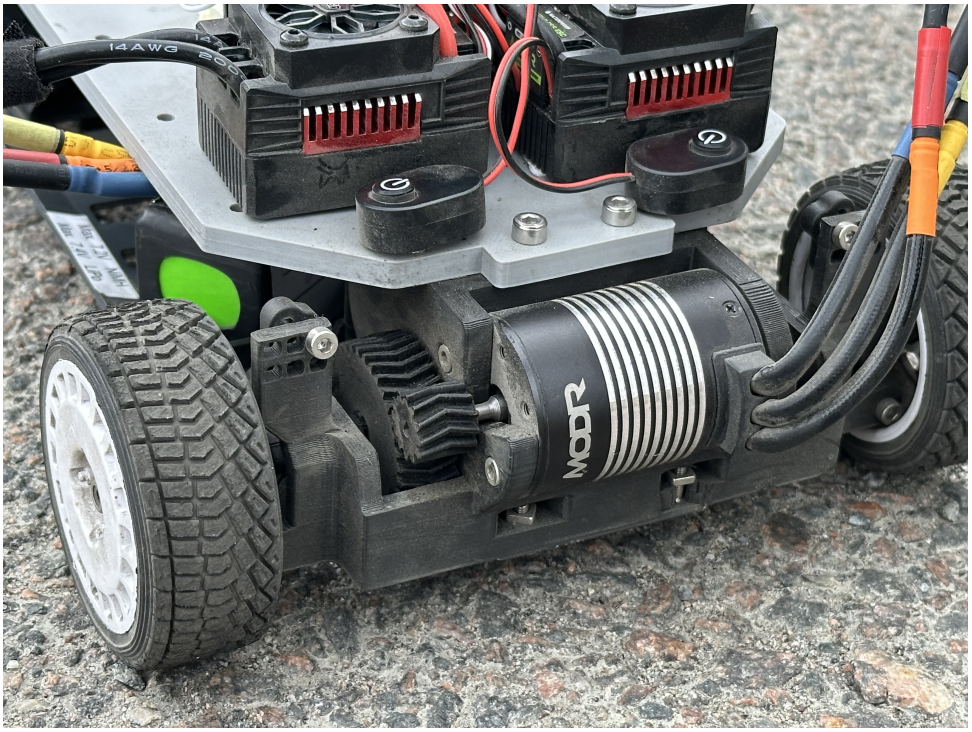
Images of the vehicle platform from different angles are included.








MODR


14AWG
MODR

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS