TÉCNICO LISBOA

VIENA Electric Vehicle Current Inverter

Programming, Implementation, Integration and Performance

Nuno Rafael Cortiço Cunha

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisors: Prof. Paulo José da Costa Branco
Eng. Francisco Ferreira da Silva

Examination Committee

Chairperson: Prof. Célia Maria Santos Cardoso de Jesus
Supervisor: Prof. Paulo José da Costa Branco
Member of the Committee: Prof. João Filipe Pereira Fernandes

November 2022

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Ao meu avô, Manuel Pereira da Cunha,
por tudo o que tem feito por mim

Acknowledgments

I would like to thank the following people, without whom I would not have been able to complete this research, and without whom I would not have made it through my masters degree.

My two supervisors, Professor Paulo Branco and Eng. Francisco Silva for providing guidance and feedback throughout this project. Professor João Fernandes for all his observations and suggestions in developing a successful project.

To the laboratory technicians Eng. Andrés Zuñiga and Mr. Duarte Batista, who helped me in the installation process of the project.

To my fellow colleagues, whom I shared my workspace with and made sure I wouldn't pull a cable from test driving the car.

Figure 1: A big thank you to my lab colleagues

Abstract

The VIENA project (Veículo Inteligente Elétrico com Navegaçao Autónoma) aims to create a flexible and accessible platform for the academic community that allows the carrying out of projects and tests within the scope of autonomous driving in electric vehicles. The VIENA project began to be developed in 2016 on a Fiat Seicento Elettra donated by Fiat to the Department of Electrical and Computer Engineering at Instituto Superior Técnico. The SEVCON Gen4 inverter was obtained in order to be integrated with the VIENA project where key objectives were defined: to be fully functional and compatible with the current car system, develop base code so that changing drive parameters and readings are easier for the user, testing in order to fully optimize the inverter for the electric motor.

In order to communicate with the inverter, a Raspberry Pi is used with a custom made CANopen board, a MCP2515 CAN controller microchip with a MCP2551 as CAN transceiver. Programming the inverter will also be made from the Raspberry using *python* has the main programming language.

Different tests were done to fine-tune the vehicle stability and response time with the throttle, this is mainly associated with different sets of combinations for the Proportional and Integral (PI). With tests done with different PI values, some issues were solved, such as the stability of the inverter in low speeds. However, the engine breaking problem whenever the throttle is released, specially in downhill, is yet to be solved.

Keywords: Electric Vehicle, VIENA, Current inverter, Field-oriented Control, Induction motor, Raspberry Pi, CANopen, Python.

Resumo

O projeto VIENA (Veículo Inteligente Elétrico de Navegação Autónoma) pretende criar uma plataforma flexível e acessível para a comunidade académica que permita a realização de projetos e testes no âmbito da condução autónoma em veículos elétricos. O projeto VIENA começou a ser desenvolvido em 2016 num Fiat Seicento Elettra doado pela Fiat para o departamento de Engenharia Eletrotécnica e de Computadores do Instituto Superior Técnico. O inversor SEVCON Gen4 foi obtido para ser integrado ao projeto VIENA, onde foram definidos os principais objetivos: ser totalmente funcional e compatível com o sistema atual do carro, desenvolver o código base para que a alteração dos parâmetros e leituras do drive seja mais fácil para o utilizador, testar para otimizar totalmente o inversor para o motor elétrico.

Para se comunicar com o inversor, é usado um Raspberry Pi com uma placa CANopen customizada, é usado um microchip controlador de CAN, MCP2515, com um transceptor de CAN, MCP2551. A programação do inversor também será feita a partir do Raspberry utilizando *python* como linguagem principal de programação.

Diferentes testes foram feitos para ajustar a estabilidade do veículo e o tempo de resposta com o acelerador, isso está associado principalmente a diferentes conjuntos de combinações para o Proporcional e Integral (PI). Com testes feitos com diferentes valores de PI, alguns problemas foram resolvidos, como a estabilidade em baixas velocidades. No entanto, o problema de quebra do motor sempre que o acelerador é liberado, principalmente em descidas, ainda não foi resolvido.

Palavras-chave: Veículo Elétrico, VIENA, Inversor de corrente, Controlo Orientado de Campo, Motor de indução, Raspberry Pi, CANopen, python

Contents

List of Tables

List of Figures

Abbreviations

BEV Battery Electric Vehicle.

CAN Controller Area Network.

COB Communication Objects.

DCF Device Configuration File.

DLC Data Length Code.

DTC Direct Torque Control.

EDS Electronic Data Sheet.

EV Electric Vehicle.

FOC Field Oriented Control.

FS Foot Switch.

IM Induction Motor.

OD Object Dictionary.

OSI Open System for Interconnection.

PDO Process Data Object.

PI Proportional and Integral.

PM Permanent Magnet.

PMSM Permanent Magnet Synchronous Motor.

PTC Positive Temperature Coefficient.

PWM Pulse Width Modulation.

PWM-VSI Pulse Width Modulated Voltage-Source Inverter.

RPDO Receive Process Data Object.

RPI Raspberry Pi.

RTR Remote Transfer Bit.

SDO Service Data Object.

SVM Space Vector Modulation.

TPDO Transmit Process Data Object.

VIENA Veículo Inteligente Elétrico com Navegaçao Autónoma.

VSC Voltage Source Converter.

Chapter 1

Introduction

In this chapter, the motivation for this thesis is firstly portrayed. Secondly, a brief explanation of the subject will be given; introducing the objectives to accomplish and the deliverables available. Lastly, a detailed structure of the thesis outline is provided.

1.1 Motivation

Across the globe, momentum is building to increase adoption of light-duty passenger Electric Vehicle (EV) and reduce the use of internal combustion engine vehicles to reduce the emissions of NO_2. EV adoption has increased in recent years, driven by a confluence of circumstances: from improvements in battery technology, to supply and demand side policies, such as EV purchase incentives or requirements for auto manufactures, as well as increased access to enabling infrastructure. While increasing EV sales cannot be attributed to any single event, these factors may have created positive feedback loops that made EVs more attractive and accessible, such for example, lower cost of batteries making EVs cheaper to produce and purchase [1]. With this in mind, Fiat, donated in 2017 a Fiat Seicento Elettra to Instituto Superior Técnico energy department, and thus, the Veículo Inteligente Elétrico com Navegaçao Autónoma (VIENA) project was born, with the goal to make an electrical vehicle completely autonomous.

1.2 Objectives and Deliverables

This project aims to contribute for the development of the VIENA car. The vehicle is now under renovation of its powertrain system, where new batteries and a new inverter were acquired. This work focus on the correct implementation, testing and characterization of the new power inverter, optimized for the VIENA car. The following objectives are set:

- Set up the new Inverter - SEVCON Gen4 110 V 300 A Size 4 in order to be fully functional and compatible with the car system available.

- Develop base code in order to easily interact with the inverter and access its parameters, either for view or change.
- Integration and testing of the inverter in the VIENA car. Laboratory tests and field tests will be carried out to fully implement and optimize the inverter to the electric motor.

As deliverables of this thesis and result analyses templates, are provided in an online open-access Google Drive folder [1] and all the code developed are provided in a Github repository [2].

1.3 Thesis Outline

This thesis is arranged in 5 chapters. The first chapter, Chapter 1 corresponds to this introduction.

In Chapter 2, a brief description will be made of different types of motors used in electric vehicles and some examples, an introduction will be made to the VIENA project, describing the goal of the project and its history, the motor used along with its parameters. Different speed controlling techniques of the induction motor will be presented and explained in mild detail of their functionality. An explanation of how will the communication between vehicle and user will occur, detailing key aspects.

In Chapter 3 a throughout explanation of the inverter will be presented, detailing important characteristics of its functionality, connections and modes of operation. The use of the communication protocols will also be explained with detail.

In chapter 4, the experimental tests and results will be shown and explained.

In Chapter 5, concludes this thesis, providing the main achievements of the work developed and some topics proposed for future work.

[1] Google Drive folder
[2] Github repository

Chapter 2

Background

In this chapter, firstly, we will describe some different types of motors and the different EVs models that include these motors. Secondly, we will describe the VIENA project, its definition and purpose and detailed examination of the motor used. Thirdly, different controlling techniques of an induction motor will be presented. And lastly, a brief explanation of the communication protocols to use between the vehicle and the user.

2.1 Used types of motors in electric vehicles

In an electric vehicle, the electric motor transforms electrical energy from the battery into mechanical energy, allowing the vehicle to move. During regenerative activity it also serves as a generator, recovering energy back to the batteries. EVs can have varying numbers of motors depending on their requirements; the decision is determined by the type of vehicle and the functions it is expected to supply.

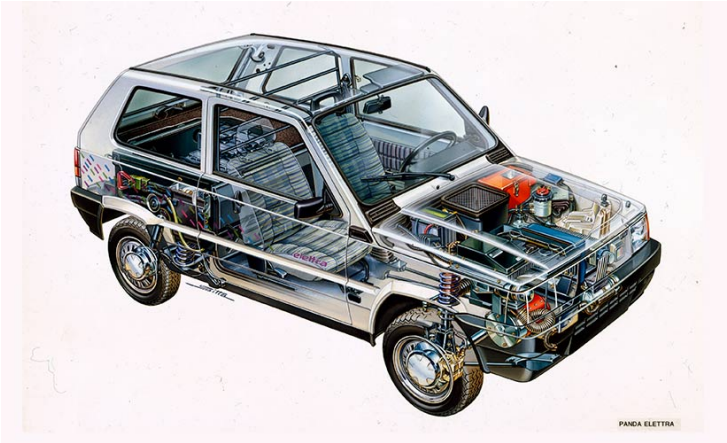
When choosing a certain kind of motor for an electric vehicle, it is critical to assess motors against a set of fundamental criteria, because different motor types display distinct characteristics, such as: power-to-weight ratio (obtained by dividing the peak power output of motor in kW by weight of motor in kg), torque-speed characteristics, motor efficiency, cost of motor controllers and the cost of the motors. A detailed comparison between these motor characteristics can be found in [2].

DC motors drives exhibit some of the requisite features for EV applications, but their lack of efficiency, bulky construction, lack of dependability due to the commutator or brushes included in them, and related maintenance requirements made them less appealing [3]. Different motor types arose to satisfy the demands of the automobile sector as power electronics and advanced control systems , with induction and permanent magnet types being the most widespread [4].

Brushed DC motor

The brushed DC motor can be found in the Fiat Panda Elettra (series DC motor), seen in Figure 2.1 along with its specifications. The downsides of this motor include its large size (lowest score in terms of power-to-weight ratio), low efficiency [2] and overheating problems. Because of these reasons, brushed DC motors are not used in EVs anymore [5]. Electricity was stored by twelve 6V lead-acid batteries.

Two were located in the engine compartment, while the other ten were installed inside a sturdy steel container occupying the base of the trunk, with an average cruising speed of 50 km/h, the range of the batteries were 100 km before depleting. At the end of 35.000 km batteries should be replaced. Charging the Panda Elettra takes approximately eight hours on a 220 V, 16 A domestic socket.



Fiat Panda specifications	
Power	30 kW
Maximum Torque	130 Nm
Max Speed	90 km/h
Total Weight	1240 kg

Figure 2.1: Fiat Panda Elettra and its specifications.

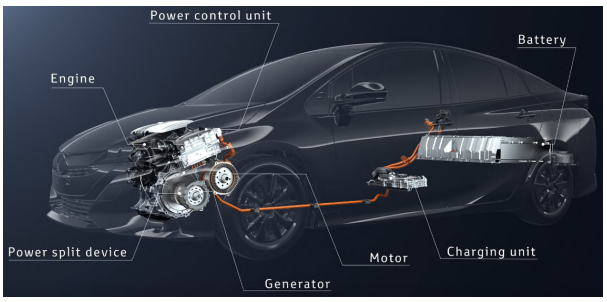
Brushless DC motors

Brushless DC motors, as the name implies, does not use brushes. They can give maximum torque constantly throughout rotation and have high performance and efficiency since they do not have any carbon brushes, which eliminates frequent brush replacement needs and maintenance expenses. Being a very light-weight motor its downside is the price range, being the most expensive motor [2]. Such motor can be found, for example, in the second generation of the Toyota Prius seen in Figure 2.2 and its specifications.

Toyota Prius specifications	
Power	100 kW
Maximum Torque	142 Nm
Max Speed	180 km/h
Total Weight	1425 kg

Figure 2.2: Toyota Prius and its specifications.

(a) Prius brushless DC motor.



(b) Prius powertrain.

Figure 2.3: Prius brushless DC motor and respective powertrain.

Permanent Magnet Synchronous motor

The Permanent Magnet Synchronous Motor (PMSM) is the most used motor in the Battery Electric Vehicle (BEV) available currently, at least 26 vehicle models use this motor technology [6]. PMSM are efficient, have good dissipation of heat and small in size. The downsides to this motor are the high cost and the demagnetization of the magnets, either due to age or severe temperatures, being low or high temperatures. This motor is used in vehicles such as the Nissan Leaf and the Kia Soul EV, these vehicles can be seen in Figure 2.4.

(a) Nissan Leaf.

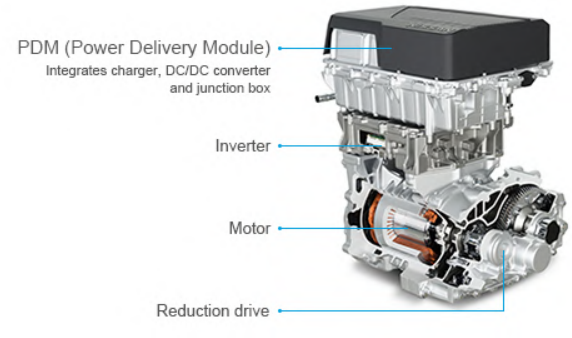
(b) Kia Soul EV.

Figure 2.4: Nissan Leaf and Kia Soul EV.

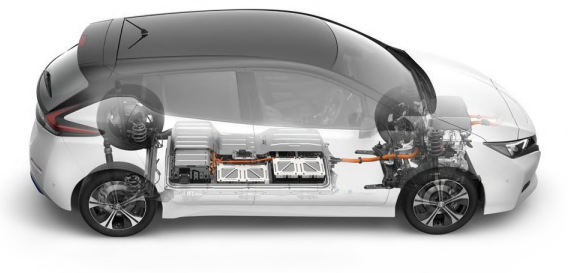
The Nissan Leaf and the Kia Soul EV specifications can be seen in Table 2.1, and respective motor and powertrain can be seen in Figures 2.5 and 2.6.

Table 2.1: Nissan Leaf and Kia Soul EV specifications.

	Nissan Leaf	Kia Soul EV
Power	110 kW	81.4 kW
Maximum Torque	320 Nm	285 Nm
Max Speed	144 km/h	145 km/h
Total Weight	1580 kg	1492 kg

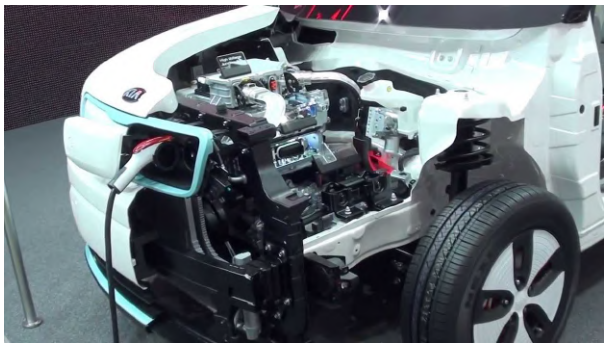


(a) Nissan Leaf PMSM motor.

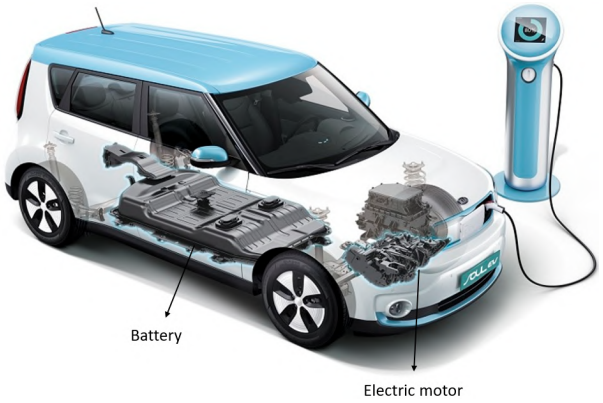


(b) Nissan Leaf powertrain.

Figure 2.5: Nissan Leaf PMSM motor and respective powertrain.



(a) Kia Soul EV PMSM motor.



(b) Kia Soul EV powertrain.

Figure 2.6: Kia Soul EV PMSM motor and respective powertrain.

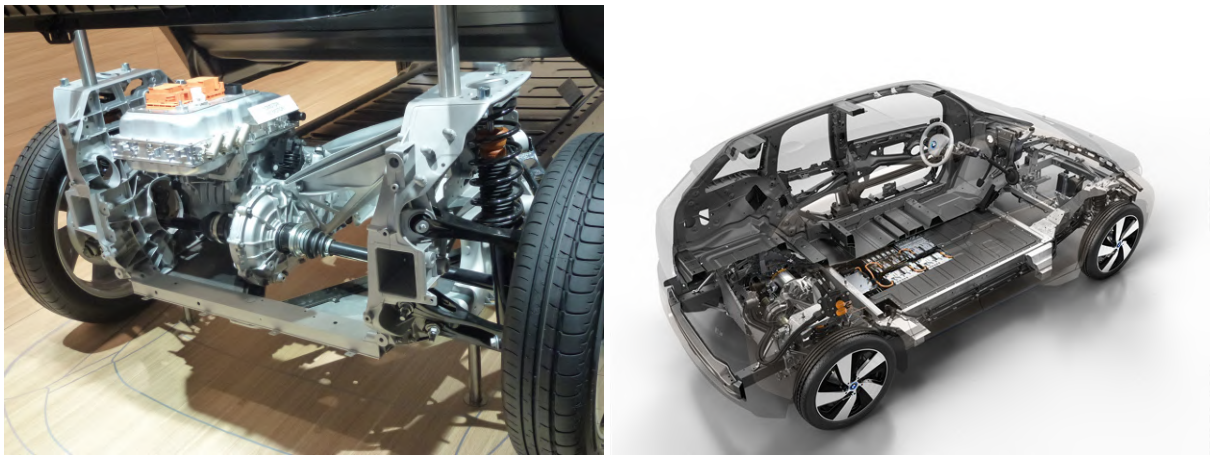
Permanent Magnet Assisted Synchronous Reluctance motor

Permanent Magnet Assisted Synchronous Reluctance Motor is a motor obtained by incorporating permanent magnets inside the rotor, enhancing efficiency with minimum back EMF and minimal stator change [7]. These motors are highly reliable although have high torque ripples. Such application can be found in the BMW i3 models, seen in Figure 2.7 and its specifications.

BMW i3 specifications	
Power	125 kW
Maximum Torque	250 Nm
Max Speed	160 km/h
Total Weight	1486 kg

Figure 2.7: BMW i3 and its specifications

The BMW i3 motor and powertrain can be seen in detail in Figure 2.8.



(a) BMW i3 motor. (b) BMW i3 powertrain.

Figure 2.8: BMW i3 motor and respective powertrain.

Induction motor

The three phase Induction Motor (IM) are widely used in electric vehicles because of high efficiency, good speed regulation and absence of commutator [2]. This type of motor has several drawbacks, such as poor starting torque and difficulty in controlling its speed. Induction motors can be found in Tesla Model S and Tesla Model X, seen in Figure 2.9.

(a) Tesla Model S.

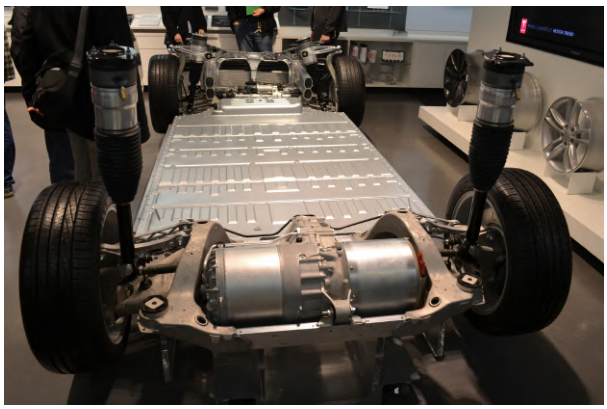
(b) Tesla Model X.

Figure 2.9: Tesla Model S and Tesla Model X.

The Tesla Model S and the Tesla Model X specifications can be seen in Table 2.2, and their respective powertrain can be seen if Figure 2.10.

Table 2.2: Tesla Model S and Tesla Model X specifications

	Tesla Model S	Tesla Model X
Power	615 kW	580 kW
Maximum Torque	1300 Nm	1140 Nm
Max Speed	260 km/h	250 km/h
Total Weight	2215 kg	2352 kg



(a) Tesla Model S powertrain.

(b) Tesla Model X powertrain.

Figure 2.10: Tesla Model S powertrain and Tesla Model X powertrain.

Switched Reluctance motor

These motors have a simple and durable mechanical structure, a cheap cost, a fast speed, a low risk of failure, an inherent extended constant power range, and a high power density that is beneficial for EV applications. When compared to Permanent Magnet (PM) machines, they are more noisy due to

the variable torque nature, have worse efficiency, and are bulkier in size and weight. This motor can be found for example, in the Tesla model 3, seen in Figure 2.11, and its specifications.

Tesla model 3 specifications	
Power	211 kW
Maximum Torque	640 Nm
Max Speed	225 km/h
Total Weight	1612 kg

Figure 2.11: Tesla model 3 and its specifications

A detailed examination of the Tesla model 3 motor can be seen in Figure 2.12.

Figure 2.12: Tesla model 3 motor.

Synchronous Reluctance motor

A Synchronous Reluctance Motor operates at synchronous speed, combining the benefits of both PM and induction motors. They are as strong and fault-tolerant as an IM, as efficient and compact as a PM motor, and do not have the limitations of PM motors [7]. They use a control method similar to that of permanent magnet motors. Controllability, manufacturing, and poor power factor are some of the issues with these motors that prevent them from being used in EVs.

2.2 VIENA Project

VIENA stands for ”Veículo inteligente elétrico de navegação autónoma”, which translate to ”Autonomous Navigation Intelligent Electric Vehicle”. The VIENA is a scientific-pedagogic project which conciliates research and teaching of electric vehicles. It is intended to use VIENA as a learning platform, where students can test and apply their knowledge acquired in electric and computers engineering, and,

at the same time, create awareness of the main challenges related to electric vehicles and autonomous electric vehicles.

This topic is very appealing both for teaching and research subjects because the trend nowadays is, automobile industry investing in self-driving and environmentally friendly vehicles, namely, electric vehicles.

With this in mind, besides the pedagogical objectives, VIENA has also scientific objectives, such as:

- Developing support technology, dedicated to providing a more autonomous, safe and electric urban mobility.
- Developing tools to detect and evaluate undesirable driving conditions.
- To continuously upgrade its power train to achieve the highest efficiency possible.
- Develop autonomous driving techniques and combination of sensors for better pose estimation.

Project VIENA car is driven by a Siemens 1LH5118 motor, a squirrel-cage induction motor coupled with the rear wheels by means of a gearbox with a fixed ratio seen in Figure 2.13. The nameplate data of the motor is illustrated in Figure 2.14 and Table 2.3 lists the values of the parameters of the equivalent circuit of the motor.

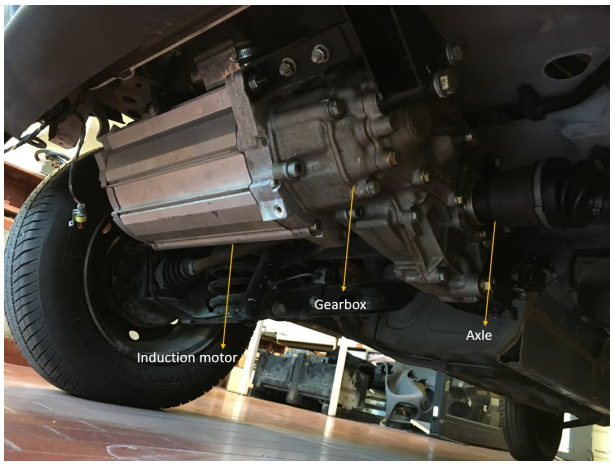


Figure 2.13: Induction motor, gearbox and the axle used in VIENA.

Figure 2.14: Nameplate data of the Induction Motor.

An induction motor was chosen based on the fact that it is more robust than DC motor and can work better in abnormal ambient conditions, can be manufactured for nominal voltage until 25 kV and reaching a rotation of 50000 rpm and being cheaper than a DC motor for the same nominal power [8].

Table 2.3: Parameters of equivalent circuit of the induction machine.

Stator resistance, R_S	8.56	$m\Omega$
Stator leakage inductance, $l_{\sigma s}$	0.063	mH
Mutual inductance, L_M	1.012	mH
Rotor resistance, R_r	10.20	$m\Omega$
Rotor leakage inductance, L_{M_r}	0.067	mH

2.3 Speed controlling techniques of an induction motor

Because VIENA uses an induction motor, it is important to know the different techniques available to control its speed, since it is one of the main drawbacks of this type of motor, because it is a constant speed motor so it's difficult to control its speed. The speed control of an induction motor is done at the cost of a decrease in efficiency and low electrical power factor. The speed of the induction motor may be changed by adjusting the slip, number of poles, or supply frequency. The various ways of induction motor speed control are classed as scalar and vector [9].

2.3.1 Scalar command

In the scalar method of speed control, it can be classified as:

- Stator voltage control
- Frequency control
- Rotor voltage control
- Stator voltage and frequency control

The most common type of scalar control for AC motors is the V/f control, where the ratio of stator voltage to electric frequency is kept constant along the range of operation. Its implementation is very simple as it does not require rotor position feedback or motor parameters. Nevertheless, its practical application at low frequency is still challenging, due to the influence of the stator resistance and the necessary rotor slip to produce torque. In addition, the nonlinear behavior of the modern Pulse Width Modulated Voltage-Source Inverter (PWM-VSI) in the low voltage range makes it difficult to use constant V/f drives at frequencies below 3 Hz [10].

2.3.2 Vector command

Vector control presents a number of options and variations, with different control schemes and levels of complexity, thus making this category of methods generally more suited for IM control. The name vector control originates from the conjugation of space and time vectors, where the quantities on the abc reference frame are projected on a two-axes reference frame in the complex plane. Two of the most

relevant categories of motor control are Field Oriented Control (FOC) and Direct Torque Control (DTC). In a general way, DTC approaches use a stationary reference frame $(\alpha\beta)$ and generate a voltage vector in order to directly control motor flux and torque. On the other hand FOC approaches usually use the position information of the rotor to apply a transformation to the dq_0 reference frame, thus decoupling the flux forming current from the torque forming current and finally controlling the motor torque and flux independently.

Field-oriented control

Field-oriented control is based on the analogy of an alternating current motor to a direct current motor, where current commutation is accomplished physically. The magnetic flux of a DC motor is regulated separately by the excitation current, while the torque is controlled by the armature current. As a result, the two currents are electrically and magnetically separated. In contrast, in AC motors, the armature current of the stator influences both the magnetic field and the torque generated. In a rotor reference rotating frame, flux and torque may be decoupled by dividing the instantaneous current into two components: the field current and the current associated with torque development. FOC consists of controlling the stator currents represented by a vector. This control is based on projections that transform a three phase time and speed dependent system into a two coordinate (d and q frame) time invariant system. These transformations and projections lead to a structure similar to that of a DC machine control. FOC techniques need two constants as input references: the torque component (aligned with the q coordinate) and the flux component (aligned with d coordinate) [11]. As Field Orientated Control is simply based on projections the control structure handles instantaneous electrical quantities. This makes the control accurate in every working operation (steady state and transient) [12].

FOC consists in aligning the direct axis of the dq reference frame with the rotor flux, represented by:

$$\lambda_{QR} = 0 \tag{2.1}$$

which consequently implies the following:

$$i_{d_r} = 0 \tag{2.2}$$

$$\lambda_{d_r} = L_m i_{d_s} \tag{2.3}$$

$$\lambda_{d_s} = L_s i_{d_s} \tag{2.4}$$

$$T_e = \frac{3}{2} n_{pp} \frac{L_m}{L_r} \lambda_{d_r} i_{q_s} \tag{2.5}$$

From (2.3) the direct rotor linkage flux is exclusively commanded by the direct stator current, whereas the electromagnetic torque is commanded by the stator's quadrature current. It is feasible to manage the direct flux and torque separately using a field-oriented controller. This raises the issue of the rotor's quadrature linkage flux being oriented in such a way that its value is null.

The direct and quadrature stator reference currents may be calculated using (2.3) and (2.5). Being $T_{e_{ref}}$ the reference electromagnetic torque and $\lambda_{dr_{ref}}$ the rated peak flux which corresponds to the direct rotor linkage which can be calculated by (2.8), where I_{mb} is the no load current of the motor. The stator

direct and quadrature reference currents are obtained from (2.6) and (2.7).

$$i_{ds_{ref}} = \frac{\lambda_{dr_{ref}}}{L_m} \tag{2.6}$$

$$i_{qs_{ref}} = \frac{T_{e_{ref}}}{\frac{3}{2} n_{pp} \frac{L_m}{L_r} \lambda_{dr_{ref}}} \tag{2.7}$$

$$\lambda_{dr_{ref}} = \sqrt{2} I_{mb} L_m \tag{2.8}$$

2.4 Communication between vehicle and user

In order to enable communication between the user and the vehicle controller, a serial communication method will be used. It is a method to send data between devices, such as computers or devices inside computers one bit at a time sequentially over a communication channel (e.g. wire) or computer bus (wire, software and communication protocol). The serial communication architecture used in this thesis is CAN bus, therefore Controller Area Network (CAN) standard and CANopen protocol are presented.

2.4.1 CAN Bus

CAN is a low-level serial communication protocol that is widely standardized. It was originally intended for the automobile sector in the 1980s, but it has since found widespread use in a variety of industries and applications. It is mostly utilized in embedded systems to offer quick communication across nodes while meeting real-time requirements [13]. Its main characteristics include:

- **Connection** – Multiple devices can be linked to the CAN bus at the same time, and there is no logical limit to the number of connectable units.;
- **Message transmission** - All messages are transmitted in predefined format;
- **Remote data request** - Nodes can request data from other nodes by sending remote frame to the bus;
- **Bus line flexibility** – The nodes connected to the bus do not have specified addresses;
- **Error detection** – Errors can be detected by all nodes. If a defective message is discovered, the message will be sent again.;
- **Connection speed** – CAN supports multiple different connections speeds up to 1 Mbit/s.

2.4.2 Bus line

The CAN bus enables different system nodes (for example, controllers and computers) to communicate with one another without the need for a host computer. Because the CAN bus connects all devices, all transmitted messages are routed to all network nodes. Each message contains a unique identifier

that allows network devices to determine whether the message is relevant or should be filtered. The priority of the message is also included in the message's identifier.

The network's nodes are linked by two wires, CAN High (CAN_H) and CAN Low (CAN_L). Resistors terminate the bus line and are required to suppress electrical reflection on the bus. It is not recommended to connect the resistors to a CAN node, but rather at the bus's ends. This is because if the resistor-equipped node is removed from the line, the bus will lose proper termination. Typically, the resistor's nominal value is 120 Ω. The described description of the CAN bus can be seen in Figure 2.15.

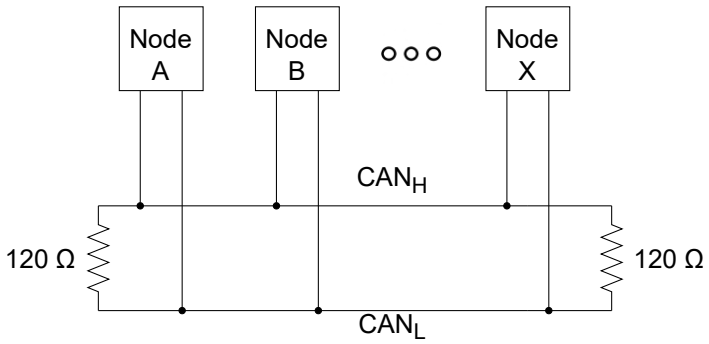


Figure 2.15: CAN bus diagram.

The Open System for Interconnection (OSI) layer reference mode, seen in Figure 2.16, specifies the seven levels from physical layer to the application layer.

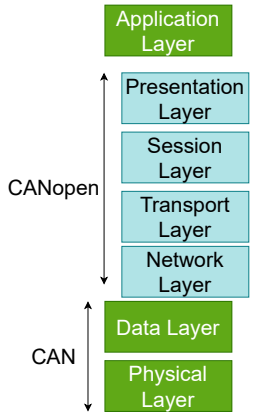


Figure 2.16: CANopen communication reference model.

To save memory, the connection between the data link layer and the application layer is bypassed in the standard CAN implementation. Additional software resources are required to cover the missing layers. Higher layer protocols such as CANopen, DeviceNet, and J1939 offer these resources. Using a higher-level protocol simplifies software development and allows the developer to concentrate only on programming the application layer.

Physical layer

The physical layer is responsible for the physical cable or wireless connection between network nodes. It defines the connector, the electrical cable or wireless technology connecting the devices, and

is responsible for transmission of the raw data, which is simply a series of binary numbers, while taking care of bit rate control.

Data layer

The data layer establishes and terminates a connection between two physically-connected nodes on a network.

Network layer

The network layer serves two primary purposes. One method is to divide segments into network packets and then reassemble the packets on the receiving end. The other method is to route packets by determining the optimum path over a physical network.

Transport layer

The transport layer collects data sent at the session layer and divides it into portions. It is in charge of reassembling the portions on the receiving end and converting them back into data that the session layer may use. The transport layer handles flow control, providing data at a pace that matches the receiving device's connection speed, and error control, determining if data was received erroneously and, if so, requesting it again.

Session layer

The session layer establishes communication channels between devices, known as sessions. It is in charge of starting sessions, keeping them open and functional while data is exchanged, and closing them after communication finishes. During a data transfer, the session layer can also define checkpoints: if the session is stopped, devices can continue data transmission from the latest checkpoint.

Presentation layer

The presentation layer is in charge of preparing data for the application layer. It specifies how two devices should encode, encrypt, and compress data such that it is appropriately received on the other end. Any data transmitted by the application layer is prepared for transmission across the session layer by the session layer.

Application layer

The application layer is the layer that actually interacts with the application of the CAN device or the operating system.

For more in-depth information regarding CAN and CANopen, technical reports and papers can be found in [14].

2.4.3 CANopen

CANopen is a CAN based internationally standardized communication system. It consists of high-layer protocols and profile specifications. CANopen has been developed as a standardized embedded network with highly flexible configuration capabilities. The main benefits of CANopen is that it releases the system developer from dealing with CAN hardware specific details and low-level layers. In addition it provides standardized Communication Objects (COB), configuration and network management data. As mentioned before, shown in Figure 2.16, CANopen covers the higher layers. One of the main features of CANopen is a table called Object Dictionary (OD), situated in the application layer ,in which configuration and process data is stored. A requirement for all CANopen devices is to implement an OD.

The OD can have different formats as such: Electronic Data Sheet (EDS) or Device Configuration File (DCF). In the OD, an entry is defined as such:

- **Index** - 16-bit base address of the object;
- **Subindex** - 8-bit sub address of the object;
- **Object type** - denotes what kind of object is at that particular index within the object dictionary;
- **Data type** - determines a relation between values and encoding for data of that type;
- **Access type** - defines the access rights for a particular object;
- **PDO mapping** - determines if the index is available to map to a PDO

The CANopen standard defines that a certain address and address ranges in the OD must contain specific parameters. For example, it is defined in the standard that index 1008h with sub-index 00h must contain the product/device name (CiA Technical Report 301) [14].

CANopen devices utilize object dictionaries as a method of communication. For example, to initiate an event on a CANopen device, a massage is sent to change a value of an object in the object dictionary. Then the change is interpreted as a signal to start the event. The master node may also need to read the configuration and process data from the object dictionary. The first mechanism to access the OD is Service Data Object (SDO) and the other is Process Data Object (PDO).

The message format for a CANopen frame is based on the CAN frame format. In the CAN protocol, the data is transferred in frames consisting of an 11-bit or 29-bit CAN-ID, control bits such as the Remote Transfer Bit (RTR), start bit and 4-bit data length field, and 0 to 8 bytes of data. The COB-ID, commonly referred to in CANopen, comprises of the ID and the control bits. In CANopen the 11 bit ID is split into two sections, a 4 bit function code and a 7 bit CANopen node ID. In addition, the message has Data Length Code (DLC) and the data field. The structure of data frame in CANopen is shown in Figure 2.17.

The main task of CANopen is to exchange process data. For this purpose most of the CAN identifiers and object dictionary entries are provided. CANopen offers two different ways to transfer data. The first method, service data objects, is based on client-server communication and it allows direct access for the client node on the object dictionary of the server node using the indexes and sub-indexes. The other method, Process Data Object, provides more efficient transmission of data.

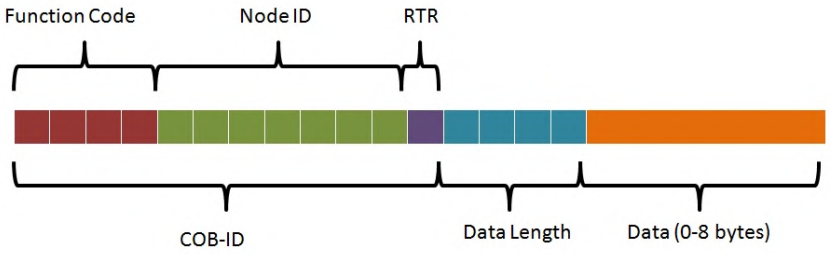


Figure 2.17: CANopen data frame format.

The SDO protocol is used for setting and for reading values from the object dictionary of a remote device. The device whose object dictionary is accessed is the SDO server and the device accessing the remote device is the SDO client.

The PDO protocol is used to process real time data among various nodes. It can transfer up to 8 bytes (64 bits) of data per one PDO either from or to the device. One PDO can contain multiple object dictionary entries and the objects within one PDO are configurable using the mapping and parameter object dictionary entries.

SDOs are not a suitable solution to access the process data, stored in the OD, since CANopen protocol also has the requirement that a node must be able to send data itself without a re-quest from CANopen master. To minimize the data exchange time and to avoid unnecessary traffic in the bus, data is sent with PDO. There are two kinds of PDOs: Transmit Process Data Object (TPDO) and Receive Process Data Object (RPDO). The former is for data coming from the device and the latter is for data going to the device; that is, RPDO can send data to the device and with TPDO, it can read data from the device. In the pre-defined connection set there are identifiers for four TPDOs and four RPDOs available, with a configuration of 512 PDOs possible.

Chapter 3

Current Inverter

In this chapter, a description of the inverter along with its characteristics, connections, parameters programmed and modes of operation, the communication protocols used will also be presented.

3.1 Characteristics

In this project it will be used the Sevcon Gen4 inverter. This device is designed to control the power of 3-phase AC induction motors and PMAC motors, in battery powered traction and pump applications. Aside from motor control, these devices contain a number of Input/Output (I/O) pins that may be used to drive line contactors, serve as inputs for brake signals, throttle action, forward and reverse switches, and so on.

The Gen4 inverter adapts its output current to suit the loading conditions and the ambient in which it is operating, it will also protect itself if incorrectly wired. The controller in question is represented in Figure 3.1, along with its dimensions [15].

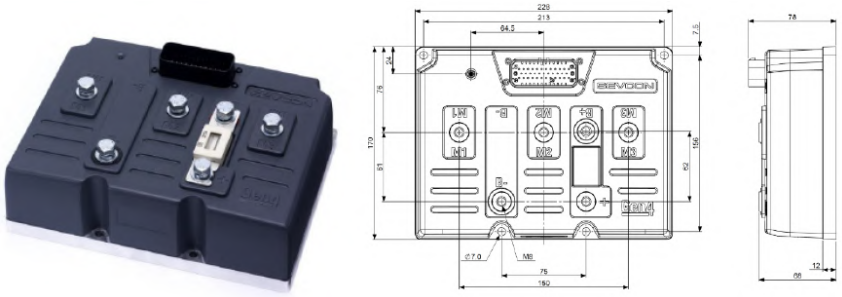


Figure 3.1: Gen4 controller.

The main input protection fuse can be mounted directly onto the controller body, as shown in Figure 3.2, connected to the battery positive cable to the B+ terminal. The B+ terminal is a dummy terminal only and has no internal connection. The fuse rating will be chosen based on our output current, which is represented in Table 3.1. Another protection used in the system, is a 24 V contactor. The voltage threshold of the inverter can be seen in Table 3.2.

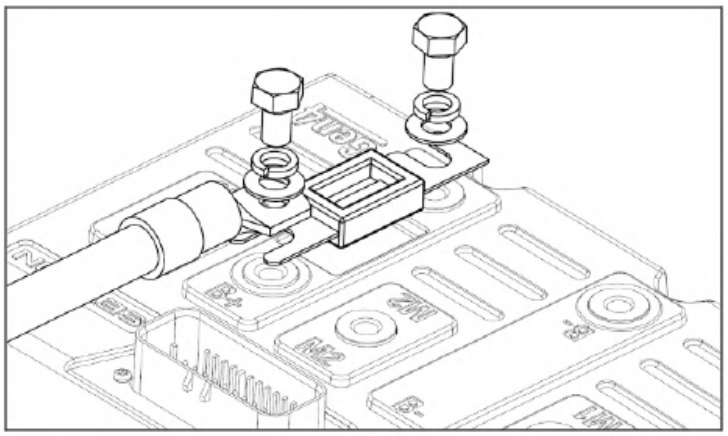


Figure 3.2: On-board fuse mounting [15].

Table 3.1: Fuse rating

Gen4 input voltage	96 to 110 V
Gen4 peak output current	300 A
Fuse rating	355 A

Table 3.2: Working voltage threshold

Conventional working voltage	67 V to 132 V
Working voltage limits	48 V to 150 V
Non-operational overvoltage limits	150 V

The inverter can be divided into two separate subsystems: the power and control systems. The power subsystem is the part of the controller circuitry that is responsible for the the transformation of the DC voltage into AC voltage of variable amplitude and frequency, used to excite the motor. In this case, this system is a Voltage Source Converter (VSC), and can also be called inverter. The DC to AC voltage conversion is usually done using a Pulse Width Modulation (PWM) technique to control the opening and closing of power semiconductors, such as MOSFETs or IGBTs. The modulator, here considered as part of the power system, takes a set of 3-phase stator voltage references and generates the PWM signals required to drive the semiconductor switches.

In general, the purpose of the control subsystem of the motor controller is the generation of the voltage references used by the inverter modulator stage to properly excite the motor. The main control methods were previously discussed in Chapter 2.3.

In the Gen4 product manual [15], Sevcon discloses key information about the type of converter, modulation and control approaches used. Concerning the control system, the controller uses (i_{dq}) current control, which indicates the control is performed in the dq reference frame. It is possible to control various proportional and integral gains, associated with speed and current controls and that a position encoder is used to measure the motor velocity. From these statements it is assumed that the control system is based on the direct FOC approach with Proportional and Integral (PI) regulators. In the power

system, it is stated that the modulation technique is Space Vector Modulation (SVM) and that the power converter is a 6 switch MOSFET bridge - a three phase, two level inverter, seen in Figure 3.3.

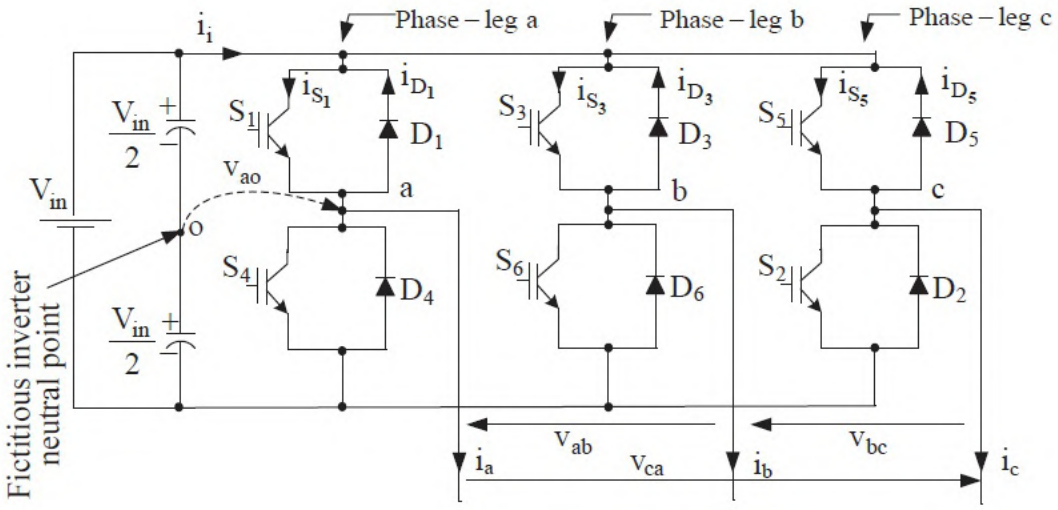


Figure 3.3: General structure of a three-phase 6 switch inverter [16].

Although the switches in Figure 3.3 are represented as IGBTs, and the inverter in question uses MOSFETs, for the sake of simplicity these will be modelled as ideal, thus neglecting the voltage drop across their terminals and respective losses. The opening and closing delay of the switches as well as the gating signals' dead time were also not considered [17].

This three phase inverter is composed by three legs, one for each motor phase, and two switches per leg. The upper and lower switches work in a complementary fashion: they are never turned on simultaneously, as this would short circuit the the DC voltage source. The point "o", which, works as a fictitious neutral point for the leg voltages: V_{ao}, V_{bo} and V_{co}, which is important for the generation of gate signals. Lastly, the voltages V_{aN}, V_{bN} and V_{cN} are defined as the phase to neutral voltages, where the neutral is the load neutral point N, different from the inverter fictitious neutral point "o". As shown in [18], one can define the inverter output voltages as in (3.1).

$$V_{abc_o} = \begin{bmatrix} v_{ao} \\ v_{bo} \\ v_{co} \end{bmatrix} \tag{3.1}$$

From where the line to line voltages of the inverter can be defined as:

$$V_{LL} = \begin{bmatrix} v_{ab} \\ v_{bc} \\ v_{ca} \end{bmatrix} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_{ao} \\ v_{bo} \\ v_{co} \end{bmatrix} \tag{3.2}$$

The phase to neutral voltages of the load can be written as:

$$v_{abc_N} = \begin{bmatrix} v_{aN} \\ v_{bN} \\ v_{cN} \end{bmatrix} \tag{3.3}$$

Assuming that the IM is balanced (all three lines share equivalent loads) and is connected in star configuration, the relation $v_{aN} + v_{bN} + v_{cN} = 0$ holds. However, this may not be the case for the inverter output voltages v_{abc_o}. From [18], these voltages can be written as:

$$v_{abc_o} = \begin{bmatrix} v_{a-h_o} + v_h \\ v_{b-h_o} + v_h \\ v_{c-h_o} + v_h \end{bmatrix} \tag{3.4}$$

Where v_h is the homopolar component of the v_{abc_o} voltages. Then this component can be defined by (3.5).

$$v_h = \frac{v_{a_o} + v_{b_o} + v_{c_o}}{3} \tag{3.5}$$

By relating these equations with the line to line voltages v_{LL}, we obtain:

$$\begin{aligned} v_{a_o} - v_{b_o} &= v_{a_N} - v_{b_N} \\ v_{a_o} - v_{c_o} &= v_{a_N} - v_{c_N} \end{aligned} \tag{3.6}$$

Summing both equations, we have the result:

$$2v_{a_o} - v_{b_o} - v_{c_o} = 2v_{a_N} - v_{b_N} - v_{c_N} = 3v_{a_N} \tag{3.7}$$

As mentioned before, the modulation process implemented by the Sevcon Gen4 motor controller is SVM. This modulation process falls in the category of PWM strategy. This is a ubiquitous technique in electronics, specially in the case of modern electric machine control. As referred in [19], one can differentiate two important categories of PWM signal synthesis approaches: carrier-based and space vector-based PWM.

Carrier-based was the first to be extensively used for its simplicity and possibility of analog implementation. This type of PWM is based on the comparison between a higher frequency carrier wave with a lower frequency modulator wave in order to generate switching signals for an inverter (in the case of motor control). With the increasing processing capacity of digital microelectronics, new techniques such as SVM started becoming prominent in the scope of power electronics. This approach utilizes the concept of space vector representation of the stator three phase voltages to compute the switching periods of each switch of the inverter. SVM technique renders a wide linear modulation range for the inverter while being more burdensome to implement than its carrier-based peers.

In [19], a relationship is established between these two approaches, making it clear that it is possible to synthesize an identical modulator using either technique. In [20] the specific case of using a carrier-

based approach for implementing a conventional (referred to as Symmetrical in [19]) SVM modulator for a three phase inverter driving a balanced star connected load was proposed. Because there is no specification of the SVM strategy used in the Sevcon Gen4 operation, it is assumed that symmetrical SVM is implemented.

Carrier-based PWM methods generate switching signals by comparison of a modulator signal with a carrier signal, seen in Figure 3.4.

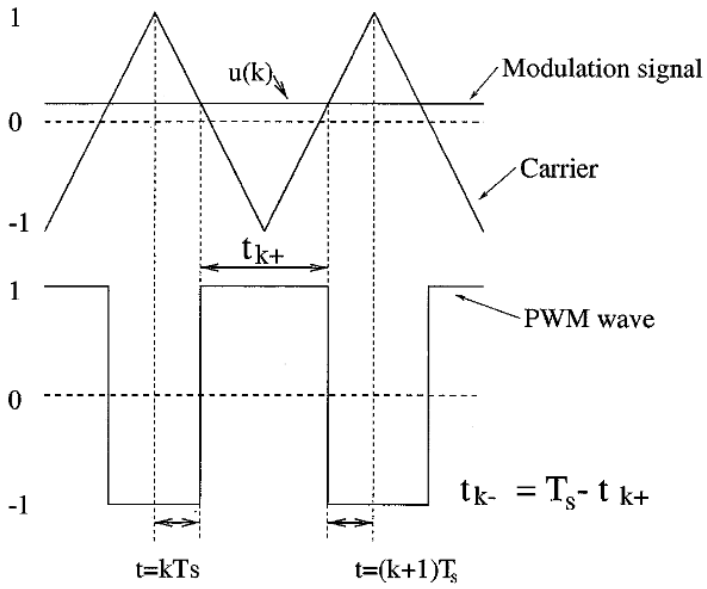


Figure 3.4: General illustration of carrier-based PWM methods - pulsed output voltage waveform generation from triangle carrier wave and modulator wave comparison [19].

In this case $u(k)$ is the modulator signal in period k, t_k+ is the ON time of the upper switch for a leg of the inverter, and t_k- is the ON ime for the lower switch of the same leg. $T_s = \frac{1}{f_s}$ is the switching period of the inverter, equal to the carrier wave frequency. Both the output PWM wave and the carrier have normalized amplitude thus, where the modulation index m is equal to the amplitude of the sinusoidal voltage reference $u(t)$. We then have:

$$\begin{aligned} t_{k+} - t_{k-} &= u_k T_s \\ t_{k+} &= \frac{1}{2}(1 + u_k) \\ t_{k-} &= \frac{1}{2}(1 - u_k) \end{aligned} \tag{3.8}$$

In [19], two main modes of operation are considered for PWM methods in general: $linear$ mode where $max(u(k)) \leq 1$ and $nonlinear$ mode where the $max(u(k)) > 1$. The latter corresponds to the situation of overmodulation. The modulator waves $u_i(t)$, where $i \in a, b, c$, can be defined as (3.9).

$$u_i(t) = u_i^*(t) + e_i(t) \tag{3.9}$$

Where $u_i^*(t)$ are defined as the fundamental signals, which are the ones set as references by the control system of the controller. These are a set of sinusoidal voltages where $u_a^*(t) + u_b^*(t) + u_c^*(t) = 0$. The

signals $e_i(t)$ are the injected harmonics, also called homopolar components or zero-sequence signals. Defining the fundamental component of the modulator signals as (3.10), the phase to inverter neutral output voltages, u_{abco}, of the inverter can then be defined as (3.11).

$$\begin{aligned} u_a^*(t) &= msin(\omega t) \\ u_c^*(t) &= msin(\omega t - \frac{2\pi}{3}) \\ u_b^*(t) &= msin(\omega t + \frac{2\pi}{3}) \end{aligned} \tag{3.10}$$

$$\begin{aligned} v_{ao} &= \frac{V_{DC}}{2}[msin(\omega t) + e_i(t)] \\ v_{bo} &= \frac{V_{DC}}{2}[msin(\omega t - \frac{2\pi}{3}) + e_i(t)] \\ v_{co} &= \frac{V_{DC}}{2}[msin(\omega t + \frac{2\pi}{3}) + e_i(t)] \end{aligned} \tag{3.11}$$

The output line-line output voltages, v_{LL}, can then be written as (3.12).

$$\begin{aligned} v_{ab} &= \frac{V_{DC}}{2}\sqrt{3}mcos(\omega t - \frac{\pi}{3}) \\ v_{bc} &= \frac{V_{DC}}{2}\sqrt{3}mcos(\omega t + \pi) \\ v_{ca} &= \frac{V_{DC}}{2}\sqrt{3}mcos(\omega t - \frac{\pi}{3}) \end{aligned} \tag{3.12}$$

From (3.12) it is once again clear that the homopolar component does not appear in the line-line voltages. From the analysis of the circuit of Figure 3.3 it is known that the maximum line-line voltage amplitude is $\frac{V_{DC}}{2}$. Then, the maximum value of the modulation index, m, is $\frac{2}{\sqrt{3}}$.

By adding different homopolar components to the fundamentals signals, different PWM modulators are obtained. For example, with $e_i(t)$ = 0, a sinusoidal PWM modulator is obtained. If $e_i(t) \neq 0$, nonsinusoidal modulators are achieved.

3.2 CANopen communication

Communication with the inverter will be made by using a Raspberry Pi (RPI) 3 Model B, a single board computer that is affordable and small in size. The RPI can be interacted with wirelessly once connected to a local network. The RPI will use a custom CANopen board that includes a MCP2515 CAN controller microchip with a MCP2551 as CAN transceiver. The MCP2515 is a chip that implements the CAN 2.0B (29-bit identifiers) specifications, has the capability to transmit and receive standard and extended data frames and remote frames. The MCP2551 serves as the interface between a CAN protocol controller (MCP2515) and the physical bus.

The RPI configuration, network setup and the protoboard schematics and connections can be seen in more detail in the following guide written by Bruno Tibério [21]. In order for the RPI to properly communicate with the inverter, both must share a common ground, the B- terminal, else the RPI wont be able to communicate properly via CANopen. Therefore the RPI can not be supplied by the batteries.

To solve this problem, a PiJuice HAT is used, a portable power platform for the RPI that keeps it powered throughout experiments. In Figure 3.5 the components described previously can be seen.

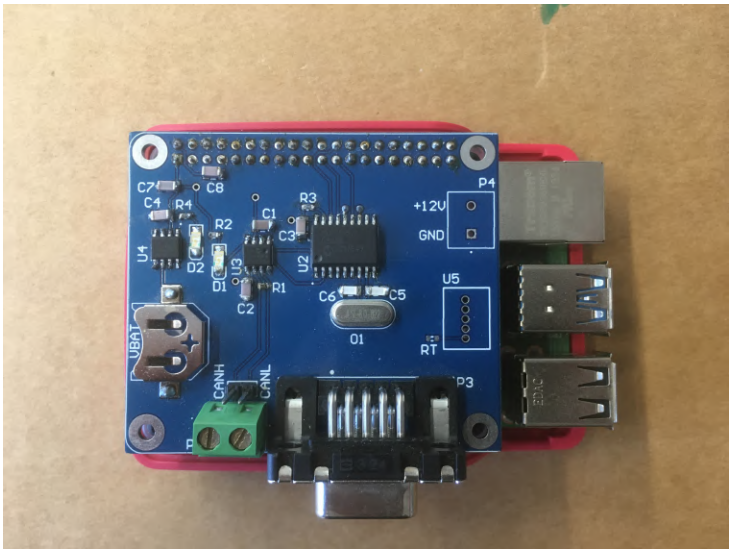


Figure 3.5: CAN communication setup.

With the CAN bus terminals connected, communication with the inverter can be established. Two programs were developed: one to view any errors regarding the operation of the inverter (both inverter unique errors and all CANopen general errors) with a detailed description of the error and a possible solution (can be viewed in the Github repository provided in Chapter 1.2), and one to configure the parameters of the inverter, such as: the proportional gain to set, the integral gain to set, rotor resistance and so on. It is important to note, that in order to change any characteristic inside the DCF file, one must have clearance to do so since there are different types of access level to the inverter. The latter can be viewed in more detail in Annex A

Some inverter parameters are different from each other and require different initial approaches when to be programmed, being by bitmask, different values for scaling and if able to map to a PDO. Therefore, it is important to have an updated version of the DCF file corresponding to the motor in question, to see the base values already written and what needs to be done to change certain aspects. Besides configuration, the code is also able to monitor certain parameters that have been set to a PDO mapping, such as: velocity, battery current and voltage, motor temperature, target torque and so on. These values are always stored in a text file that saves inside the RPI microSD to later be accessed for study and analyses.

In Figure 3.6, a manual change of the parameters can be seen for on the fly changes. It is important to start *python* in the folder where the inverter DCF file is located, so the *canopen* library can access and read the indexes provided, adding these to nodes. Most parameters require the inverter to be in a *pre-operational* state and the access level must be in *Engineering* or level 4, in order for them to be saved if changed. In order for a node to be viewed, it most follow the following structure: *['Index name']['Sub-index name']* OR *['Index number']['Sub-index number']* given in the DCF file, names read from the node command are case-sensitive. All parameters read from the *.raw* command return a hexadecimal value,

which has then to be turned to a decimal value and finally apply (only if applicable) the scaling number. From the example given in Figure 3.6, the *.raw* commands returns the hexadecimal number 310, which is equal to number 784 in decimal, multiplying by the scaling number corresponding to that parameter, K_p, we obtain 0.02.

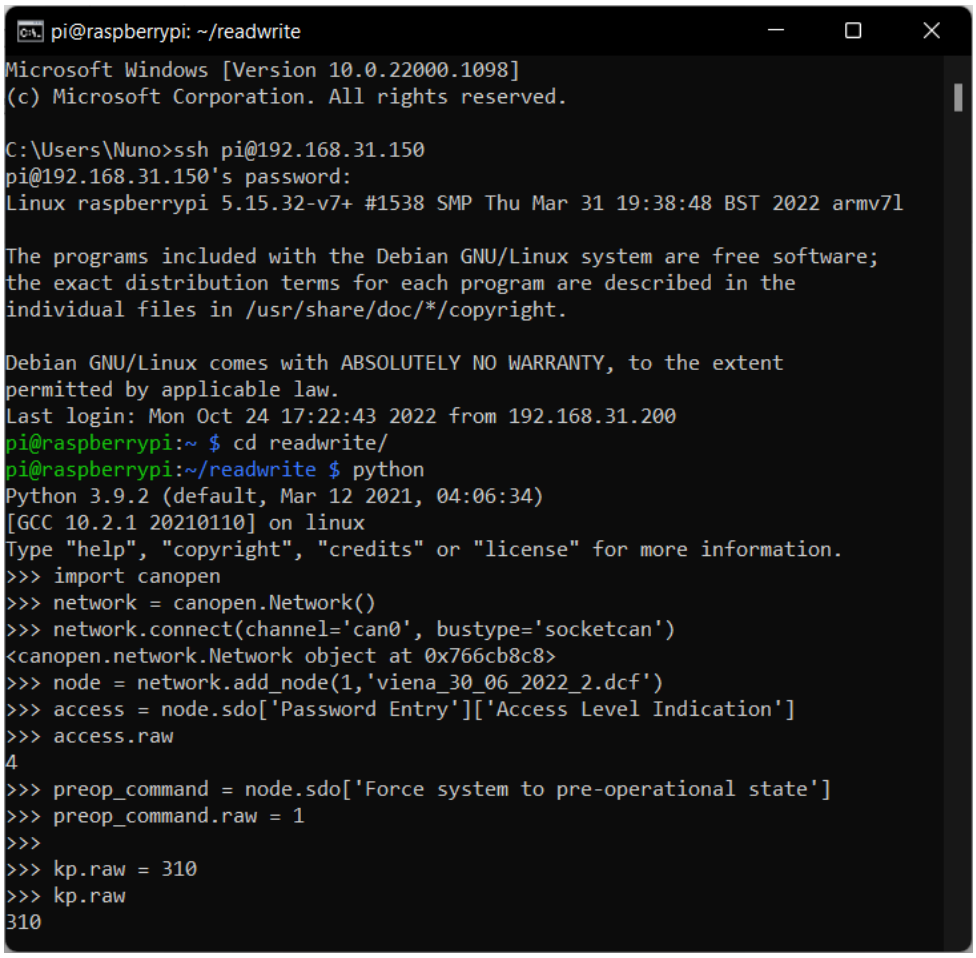


Figure 3.6: Manually parameter changes.

3.3 Peripheral inverter connections

Signal connections are made to the inverter via a 35 way AMPSeal connector. The pins are protected against short-circuits to the battery positive or negative terminals. Signal wire sizes are between , these signal wires are fed through AMPSeal specific contactor.

Each pin on the AMPSeal corresponds to a specific function, each pin function can be found in the Gen4 manual, what type of signal the pin transmits, what to connect, maximum current and voltage rating and important information about small details to take into consideration. The schematic of the connections made can be found in Annex B and the power schematic in Annex C. The inverter is powered by 24 lithium polymer battery cells, a more detailed description of the batteries can be viewed in [22]. In Annex D pictures of the setup used for the VIENA project can be seen.

The type of encoder used by the IM is an AB encoder, supplied with 5 V., has two output signals,

A and B, which give a pulsed signals when the device is moved. The pulses emitted from the A and B outputs are quadrature-encoded, meaning that when the incremental encoder is moving at a constant velocity, the A and B waveforms are square waves and there is a 90 degree phase difference between A and B.

The IM uses a Positive Temperature Coefficient (PTC) thermistor, whose resistance increases with increasing temperature.The model used is a KTY84/130 silicon temperature sensor. A thermistor map has been programmed to the inverter using the reference points seen in Table 3.3. It is important to note that the encoder and thermistor share the same ground, but isolated from the B- terminal of the inverter.

Table 3.3: Thermistor Mapping.

Temperature ($^\circ$ C)	Resistance (Ω)
0	498
20	581
40	672
60	773
80	882
100	1000
120	1127
140	1262

The throttle uses a three-terminal 2000 Ω potentiometer. Two of the terminals are attached to the opposing ends of a resistive element, while the third terminal is connected to the pedal.

VIENA uses a simple forward and reverse switch to choose which direction the vehicle moves. As for the foot brake, VIENA has an on and off integrated switch, in which it applies maximum foot braking when the switch is closed, contrary to the throttle which uses a potentiometer.

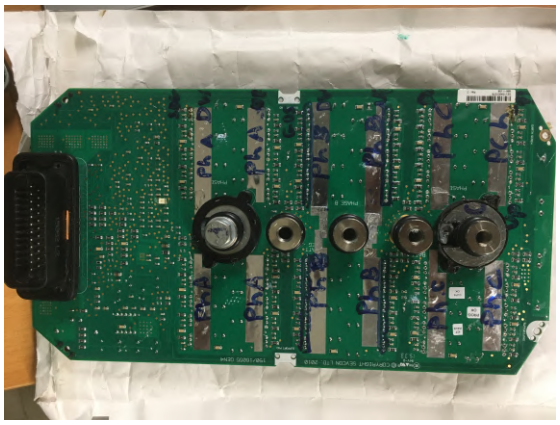
In order to operate with the Gen4 inverter, the safety interlocks must be met (if programmed), such options include: Foot Switch (FS)1, Deadman, Seat, Handbrake and Sequence Fault Masking. The FS1 switch is normally part of the throttle assembly, it closes when the throttle is pressed, however VIENA throttle does not include such a switch, so a normal on and off switch is used instead. The FS inhibits drive until it is active.

To fullfil the cooling requirements for the inverter, to avoid overheating, thermal paste is used between the controller and the mounting surface. The mounting surface is a flat metal plate, the entire assembly is then bolted together at all mounting holes.

3.4 Current Control

Examining the inverter from the inside, seen in Figure 3.7a, there are two current sensors, one placed in phase A and one in phase C. Current sensing is accomplished with hall effect current sensors. These are simply a slotted toroidal magnetic core around the power conductor. The slot concentrates the magnetic field, allowing a hall effect sensor in the slot to sense the magnetic flux which is proportional

to the current. The gap / hall sensor is under the clear silicone. Due to high currents, iron laminations are used. The opposite side of the inverter can be seen in Figure 3.7b.



(a) Current sensors in the inverter

(b) Opposite side of the inverter.

Figure 3.7: Gen 4 size 6

With this information, it is assumed that the Gen4 inverter measures current from phase A and phase C, and phase B is estimated from (3.13).

$$i_B = -(i_A + i_C) \tag{3.13}$$

3.5 Inverter driving options

The Gen4 inverter provides two driving options: torque mode and speed mode. In torque mode, seen in Figure 3.8, the inverter maintains the motor torque output at a constant value for a given throttle position.

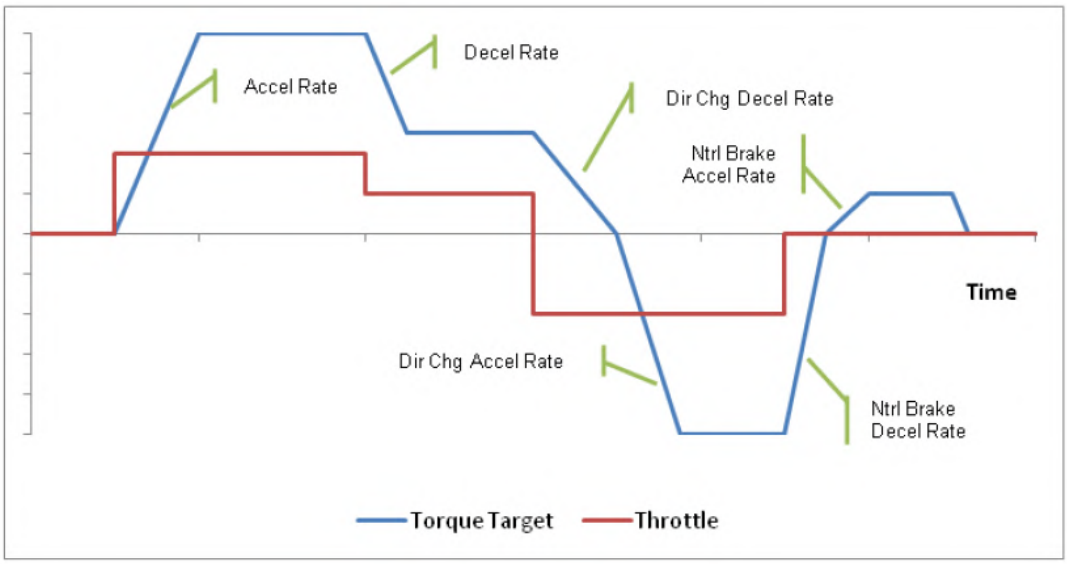


Figure 3.8: Torque mode acceleration/deceleration.

While in speed mode, seen in Figure 3.9 ,the inverter maintains the motor at a constant speed for a

given throttle position as long as sufficient torque is available.

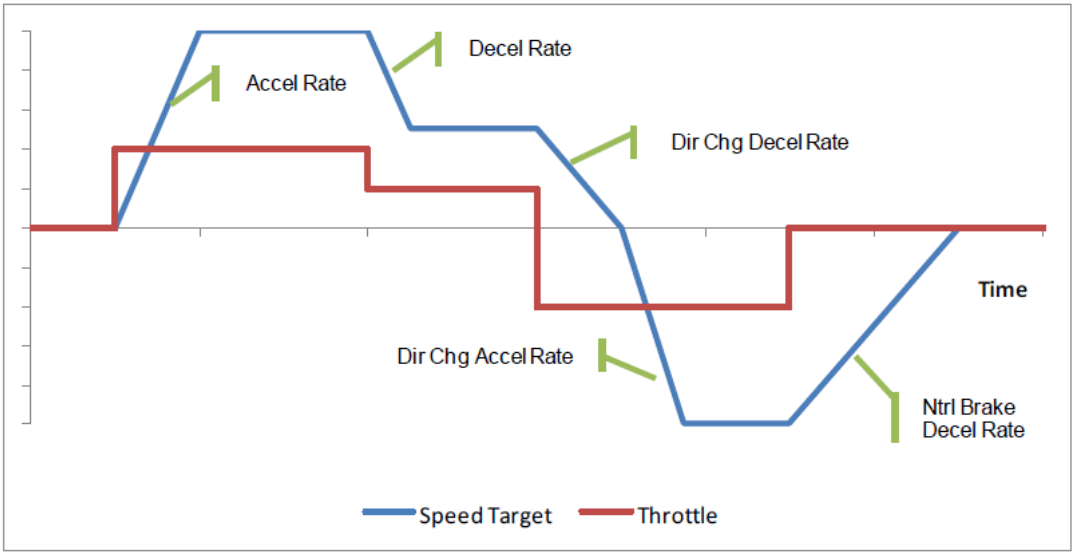


Figure 3.9: Speed mode acceleration/deceleration.

Speed mode differs from torque mode in that the torque value applied to the motor is calculated by the controller based on the operator's requested speed (determined by throttle position) and the vehicle's actual speed. In Torque mode, the acceleration and deceleration rates control the rate of change of torque. In Speed mode, the acceleration and deceleration rates control the rate of change of speed, in both modes the ramp rates can be configurable. The inverter performs measurements of the motor speed and position through the motor encoder and estimates the motor torque. Additionally, it measures the input and output voltages and currents and estimates the i_d and i_q currents.

During tests, torque mode as used, which means the inverter was controlled by a torque reference.

Chapter 4

Experiments and Results

In this chapter,the experiments conducted to test the inverter are detailed. First, the experiments were aimed to address the way the inverter makes use of the FOC and test the relation between the throttle and the torque demand. Secondly the PI value variations, are performed to access the stability of the controller in reference following. Thirdly, the analysis of the efficiency of the motor, inverter and the total system are shown. Lastly, a correction to the best PI parameters are set for testing and analyses.

4.1 Torque mode analyses

In order to verify the relationship of the signals when converted To dq transform in the inverter, the current and voltage values from the three-phases of the IM must be obtained, in this case, using an oscilloscope, seen in Figure 4.1. There is no known access to the values read by the inverter sensors, only after the dq transform which correlate to i_d, i_q, u_d and u_q. With the Clarke and Park transformations, a three-phase system can be transformed to a generic dq reference frame, which can be obtained from the Matrix given in (4.1).

$$\begin{bmatrix} x_d \\ x_q \\ x_0 \end{bmatrix} = \frac{2}{3} \begin{bmatrix} cos(\theta) & cos(\theta - 120^\circ) & cos(\theta + 120^\circ) \\ -sin(\theta) & -sin(\theta - 120^\circ) & -sin(\theta + 120^\circ) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} \tag{4.1}$$

Where *x* can be currents, voltages or fluxes and θ is the electrical angle (equal to number of poles divided by 2 times the spatial angle) between the direct axis and the stator phase-a axis [9].

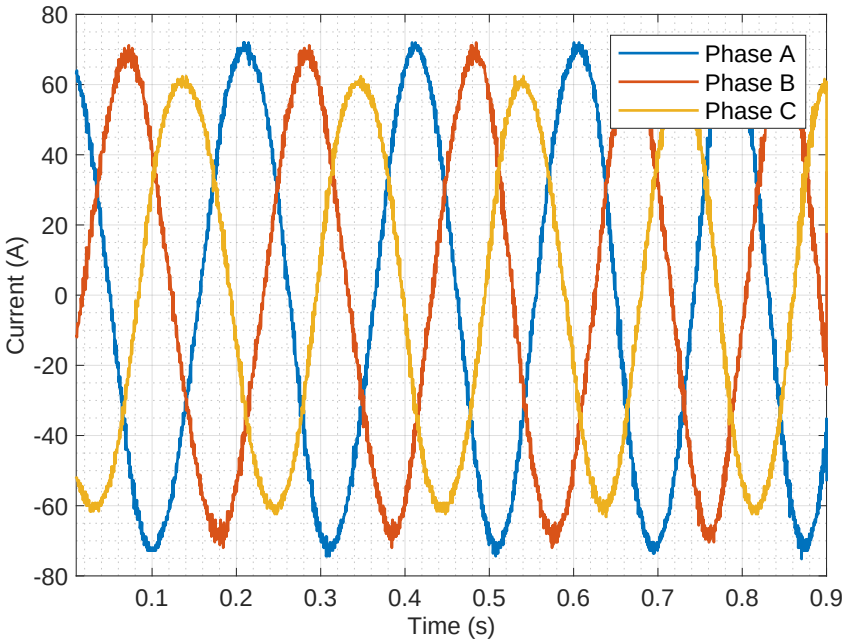


Figure 4.1: Current for each phase obtained in the oscilloscope.

To obtain θ, we integrated the electrical frequency ω over time:

$$\theta = \int \omega dt, \tag{4.2}$$

and with a fixed frequency, it yields:

$$\theta = \omega t + \theta_0 \tag{4.3}$$

where θ_0 is the initial flux electric position, which is unknown and can only be estimated.

Since the frequency, *f*, varies during time, one can obtain a linear regression equation as a function of frequency over time.

$$\omega(t) = 2\pi f(t) \tag{4.4}$$

From the data from Figure 4.1, the frequency can be obtained from:

$$f = \frac{1}{t_2 - t_1} \tag{4.5}$$

Where t_2 and t_1 are the time stamps where the phase reaches 0 A.

Knowing the equation related to the frequency we can finally calculate θ giving in (4.3), using the Clarke and Park transformation given in (4.1). We can replace θ and x_a, x_b and x_c with the currents obtained from each phase of the IM obtained in the oscilloscope. From Figure 4.2 the results can be observed.

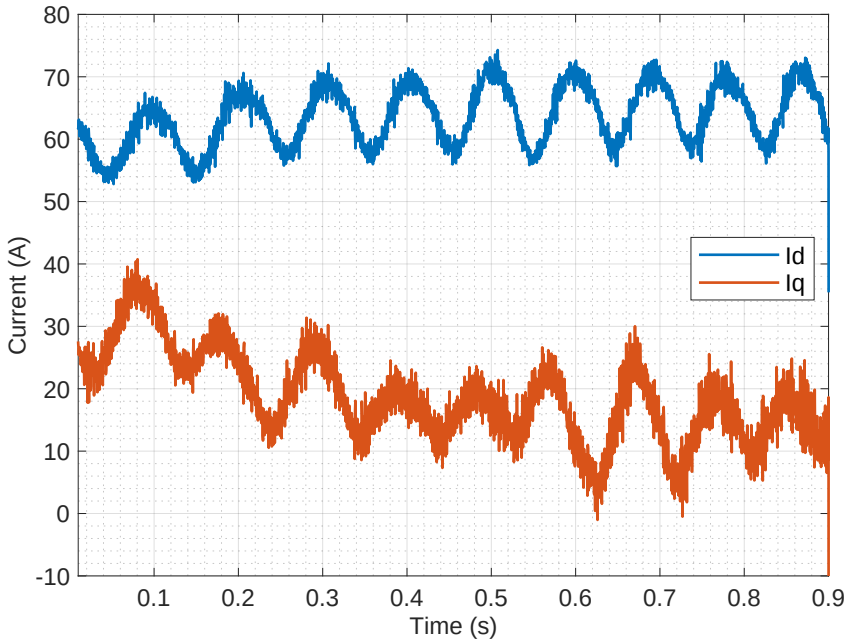


Figure 4.2: Currents obtained after Clarke and Park transformation.

The currents measured by the inverter in the dq transform can be seen in Figure 4.3, which were obtained in Index 4600_h sub-indices 7 and 8, i_d and i_q respectively.

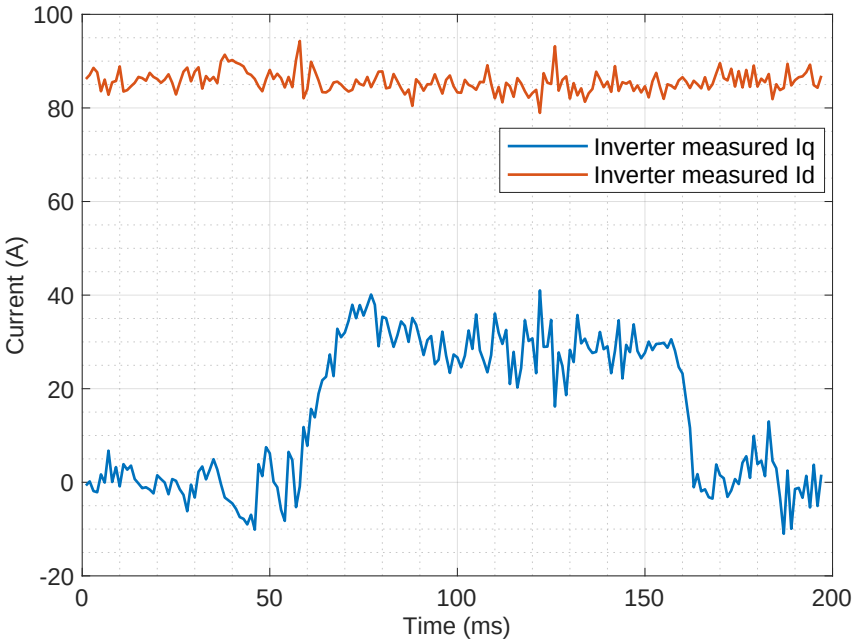
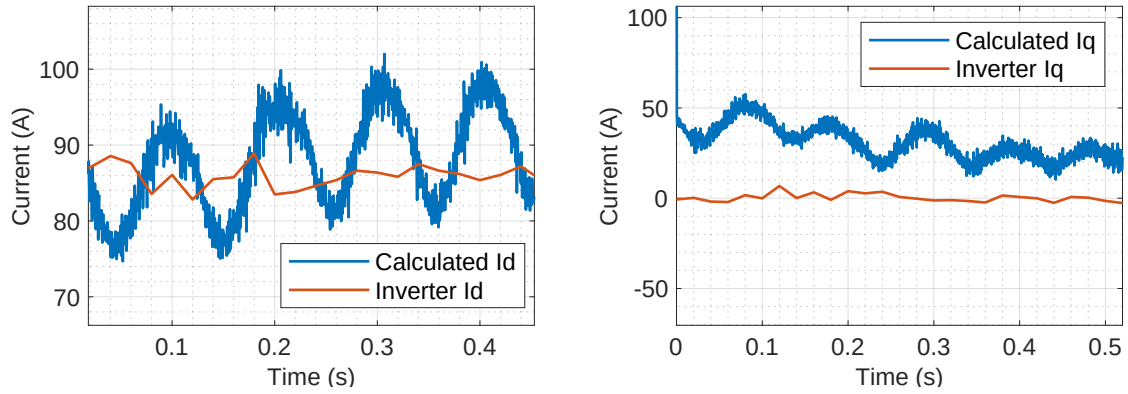


Figure 4.3: Currents obtained from inverter.

Comparing the results, the current i_d in Figure 4.4a, closely resembles the current that the inverter reads, although the i_q, in Figure, 4.4b, there is a slight off-set to it. It is also important to note that the time reference is different from the oscilloscope and inverter readings. This is due to the fact that the inverter reads values every 20 ms, which is not always consistent, while the oscilloscope reads in 100

ms, so the time window has to be adjusted in order to obtain same time stamps.

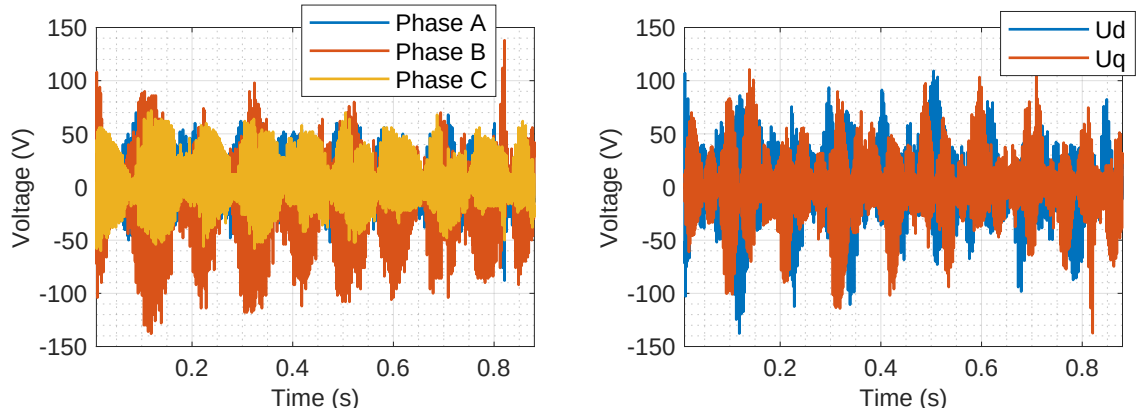


(a) I_d current comparison

(b) I_q current comparison

Figure 4.4: Comparing currents measured and calculated

The same procedure was used for the measurement and calculation of the voltage of the phases. The voltage observed in the oscilloscope can be seen in Figure 4.5a and the voltage obtained using the Clarke and Park transformation in (4.1) can be seen in Figure 4.5b. Observing the results for the voltage, it becomes hard to come to a concrete conclusion, since there is many harmonics in the signal and it is difficult to have a good perception of the situation, therefore voltage values compared with the inverter will not be shown.



(a) Phase voltages from oscilloscope.

(b) Voltage after Clarke and Park transformation.

Figure 4.5: Voltage obtained before and after the dq transform.

Given the dq values from both the current and the voltage, we can calculate the torque being produced by the IM and compare these values with the torque being read by the inverter. The torque can be calculated by

$$T = \frac{3}{2} p_p \frac{L_m}{L_r} (\lambda_{dR} i_q) \tag{4.6}$$

where the rotor inductance is expressed by

$$L_r = L_m + L_{\sigma r}, \tag{4.7}$$

and the flux λ_{dR} is

$$\lambda_{dR} = L_m i_d + L_r i_{dR} \tag{4.8}$$

With the values in Table 2.3, it is possible to estimate the output torque. In Figure 4.6, it is shown the calculated torque using (4.6) and the torque produced by the IM read by the inverter. Regardless of the noise, the calculated torque resembles the output torque read by the inverter from the IM.

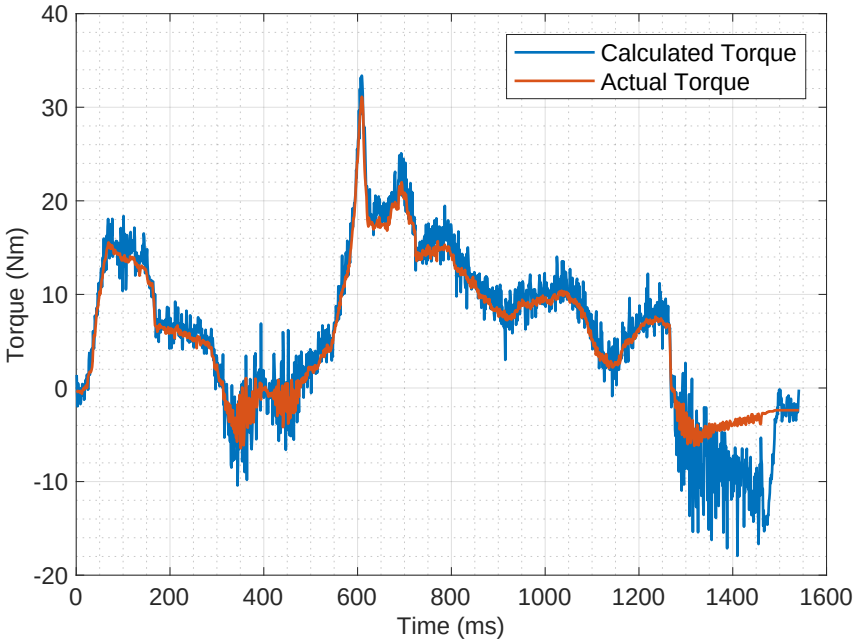


Figure 4.6: Torque value comparison.

With the analysis of the results, it is concluded that the inverter uses (4.6) and uses the motor parameter estimates given in table 2.3 in order to calculate the produced torque. It is also important to note that the IM starts to engine break once no torque reference is given, hence the oscillations at the end of the test.

Since the inverter operates under torque reference, it is possible to find the relation between the reference torque and the voltage applied to the throttle. As mentioned, the throttle used works with a potentiometer, therefore it has a maximum and minimum voltage values. When no throttle is demanded (the throttle pedal is not pressed) the maximum voltage of 5.5 V is present, while in full throttle (pedal pressed all the way down) the minimum voltage of 0.5 V is present. This relation between torque and throttle pedal voltage is expressed by:

$$\frac{0 - T_{el}}{T_{el} - T_N} = \frac{5.5 - V_{thr}}{V_{thr} - 0.5}, \tag{4.9}$$

where T_{el} is the amount of torque being applied at the given instance, T_N is the nominal torque of the IM and V_{thr} is the voltage read at the throttle. Solving the equation, it can be simplified as such:

$$T_{el} = \frac{T_N[5.5 - V_{thr}]}{5} \tag{4.10}$$

Since the inverter reads the voltage being applied to the throttle, T_{thr}, T_{el} can be plotted to view all the torque values and compare to the torque demand readings in the inverter. The throttle analysis can be seen in Figure 4.7. Where $CalculatedTorque$ is obtained from (4.10) and $TorqueDemand$ is the amount of torque demanded by the inverter by the throttle position.

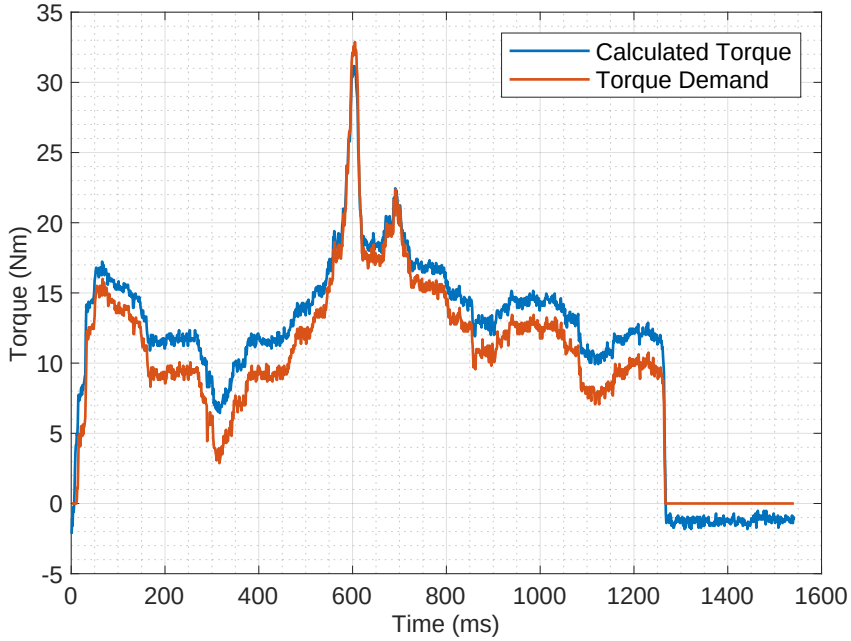
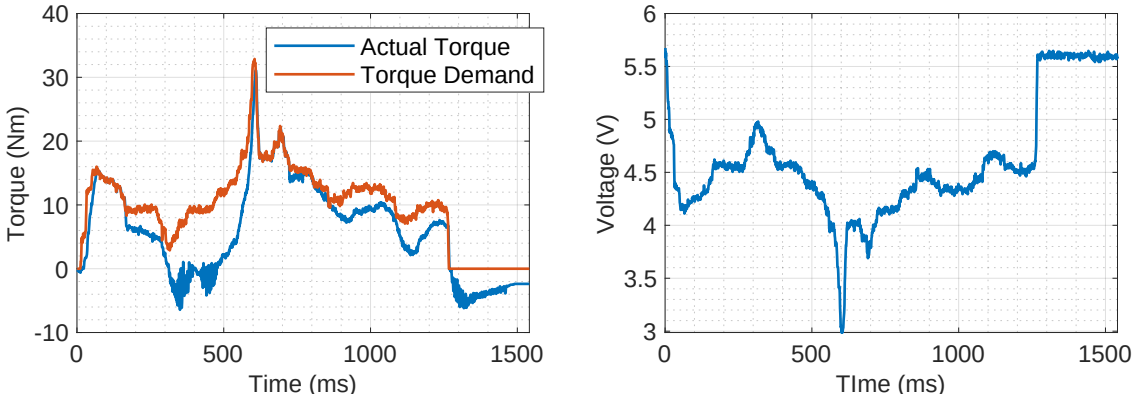


Figure 4.7: Calculated Torque (using (4.10)) and torque demand.

Observing Figure 4.7, the torque calculated by relating the throttle input voltage almost matches the torque being demanded by the inverter, only with a slight offset. This could be to noise in the system, which is notable when the throttle is not being pressed, the values are different than zero while the torque being demanded by the inverter is zero, seen at the end of the plot, which can correlate with different PI settings.

It is important to note, however, that the torque being read by the inverter does not equal the torque being demanded for the amount of throttle given. This can be seen in Figure 4.8a. This case can be seen clearly when the throttle is closed off. A more detailed view of the operation of the throttle can be seen in Figure 4.8b. Whenever the throttle is rolled off the inverter signals the motor to engine break, this causes the vehicle to spasm, since there is no reference torque yet the wheels are still spinning and the vehicle moves.



(a) Actual and Demand Torque comparison.

(b) Throttle Input Voltage.

Figure 4.8: Comparing Actual Torque and Torque Demand to Throttle Input Voltage.

4.2 PI Controller Analysis

An electrical drive based on the FOC needs two constants as control parameters: the torque component reference and the flux component reference. The classic numerical PI regulator is well suited to regulate the torque and flux feedback to the desired values as it is able to reach constant references, by correctly setting both the Proportional term (K_p) and the Integral term (K_i) which are respectively responsible for the error sensibility and for the steady state error. As stated previously in Chapter 4.1, when there is no torque reference, the inverter signals the motor to engine break even when it is moving. This is an issue since it does not provide smooth driving and stability. In Annex E a diagram of the FOC can be found.

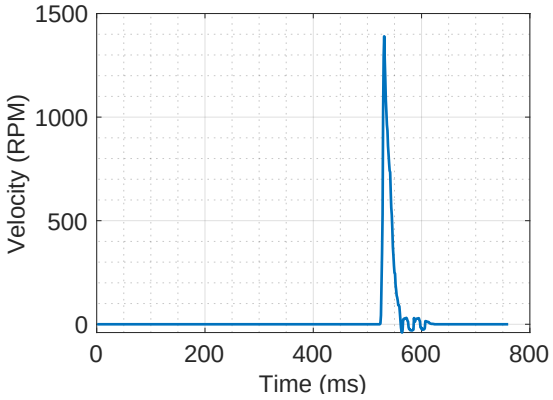
It is important to note certain characteristics of the PI controller when tuning the gains: the higher the proportional gain, the quicker the controller will try to match the actual to the intended value, but the higher the likelihood of overshooting, thus causing oscillation if the proportional gain is exaggerated. While a small gain results in a small output response to a large input error, and a less responsive (or sensitive) controller. If the proportional gain is too low, the control action may be too small when responding to system disturbances. While on the other hand, the integral gain adds up the difference of the intended value minus the actual value over time and increases or decreases the gain accordingly. If the integral gain value is too small it will take too long for the actual value to reach the intended position, if it is too big there will be slow oscillation of actual value around the intended position.

4.2.1 Wheels lifted off the ground

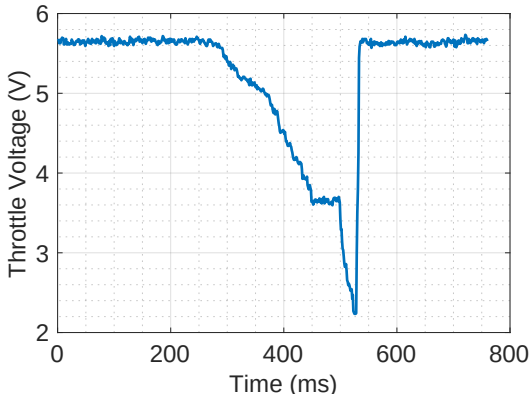
For safety reasons, most of the PI testing was done with the back wheels lifted off the ground due to the motor becoming unstable and have big speed gains in a very short period of time. Several combinations of the PI tuning can be seen below, in order to find the smoothest and safest combination to test with the wheels on the ground.

$K_p = 0\,, \quad K_i = 0$

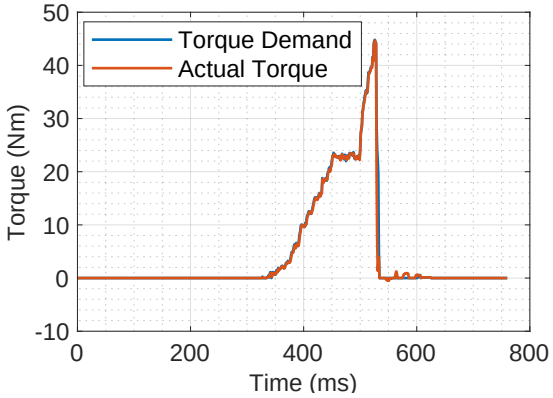
The first set of experiments were setting both regulators to 0. With this set of values, although the inverter turns on and registers no type of errors, it is required close to full throttle for the motor to start to spin, and the acceleration is not smooth but instead a very quick and unstable motor acceleration occurs (jumping from 0 RPM to 1500 RPM) seen in Figure 4.9a. Torque can be registered before the wheels start to spin Figure 4.9c. It is important to notice that the torque is correlated with the throttle input voltage, therefore, theoretically there will always be torque if the throttle is pressed, but that does not mean that the motor spins.



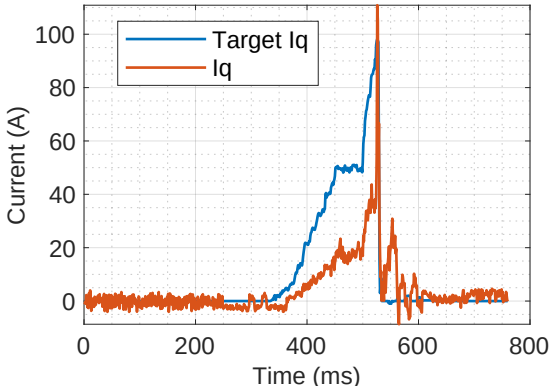
(a) Velocity



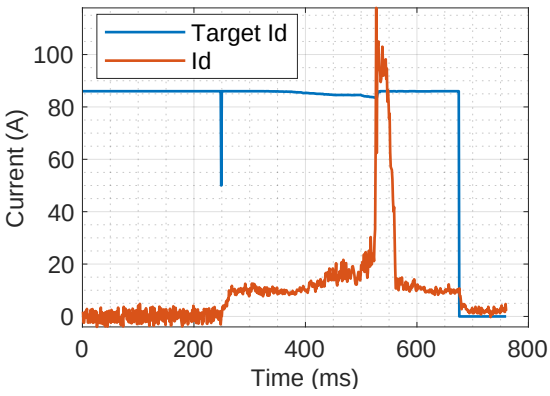
(b) Throttle



(c) Torque



(d) I_q

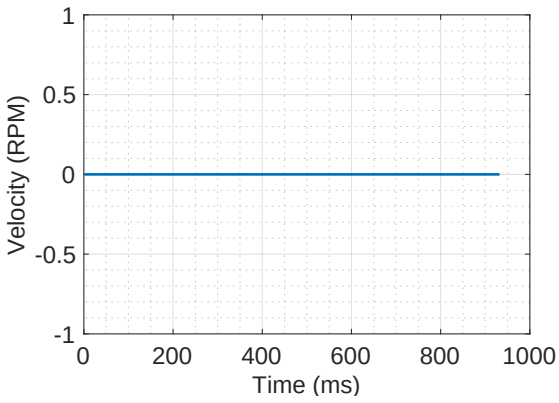


(e) I_d

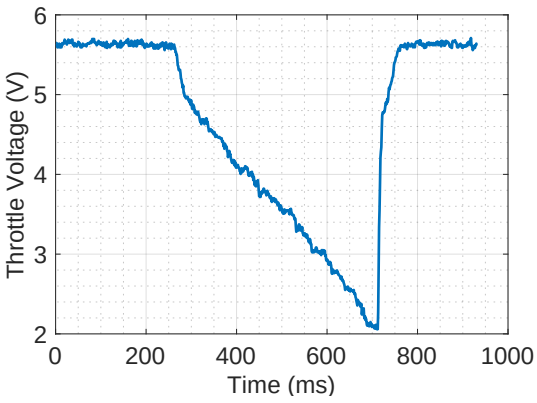
Figure 4.9: K_p 0 K_i 0 - Wheels up testing

$K_p = 1\,, \quad K_i = 0$

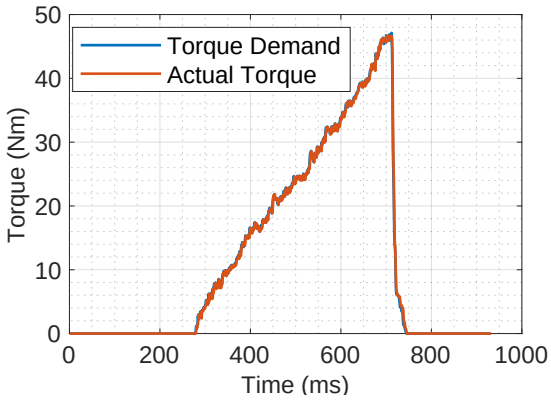
From Figure 4.10a it can be seen that the motor does not spin no matter how much throttle given. Figure 4.10b. From Figures 4.10d and 4.10e, the currents are very oscillatory and have more noise compared to the previous PI setting. This could be due to the fact that the proportional gain is set very high and the integral gain at 0.



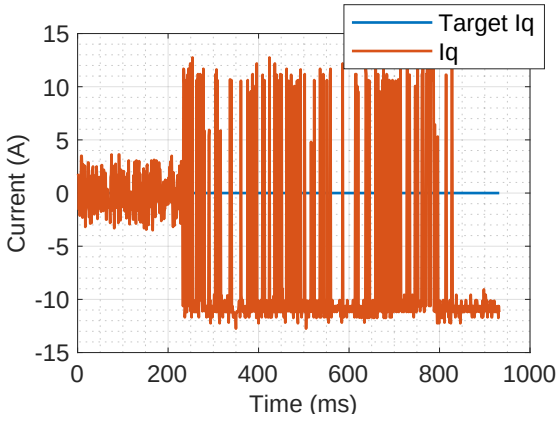
(a) Velocity



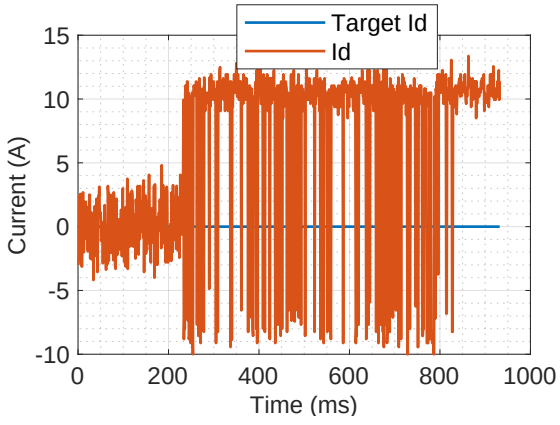
(b) Throttle



(c) Torque



(d) I_q



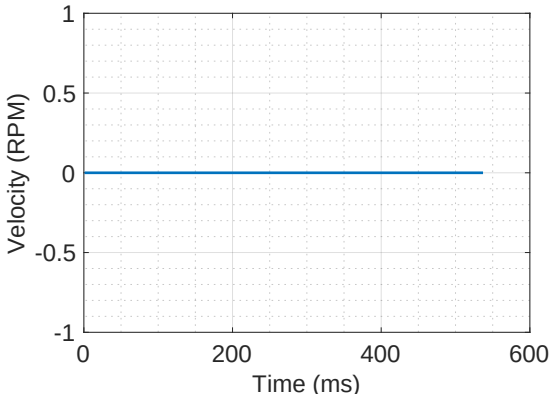
(e) I_d

Figure 4.10: K_p 1 K_i 0 - Wheels up testing

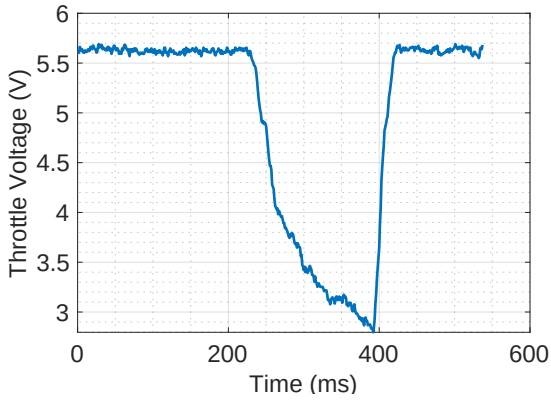
$K_p = 1\,, \quad K_i = 1$

Similar to the combination of $K_p = 1$ $K_i = 0$, the motor does not spin seen in Figure 4.11a no matter how much throttle given Figure 4.11b. From Figures 4.11d and 4.11e, the currents are noisy however

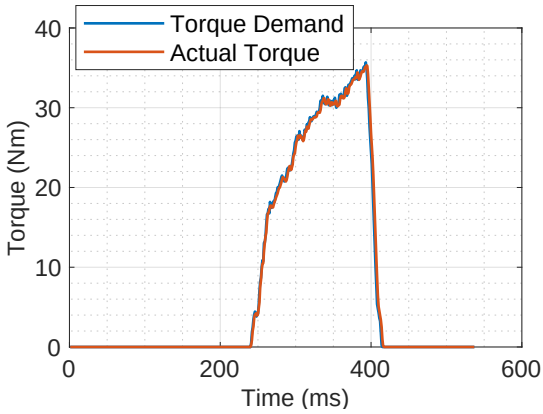
still oscillatory, this is due to the fact that the integral gain is set to a higher value.



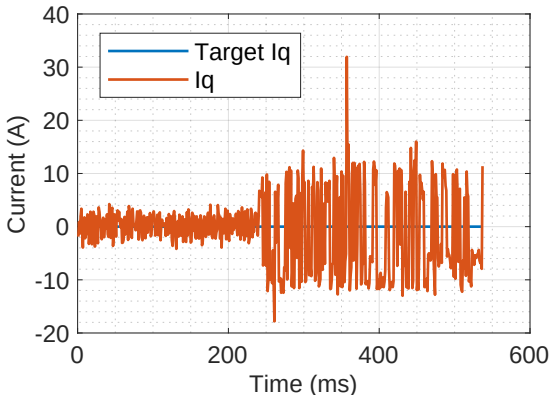
(a) Velocity



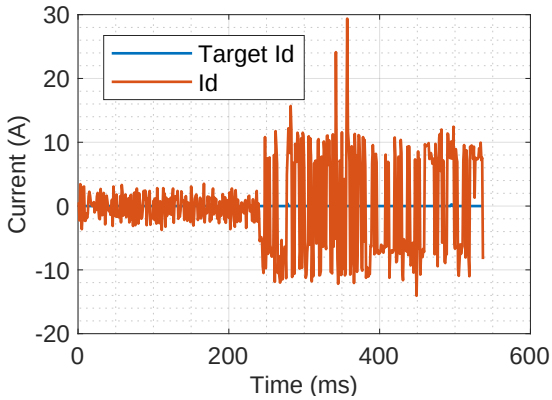
(b) Throttle



(c) Torque



(d) I_q



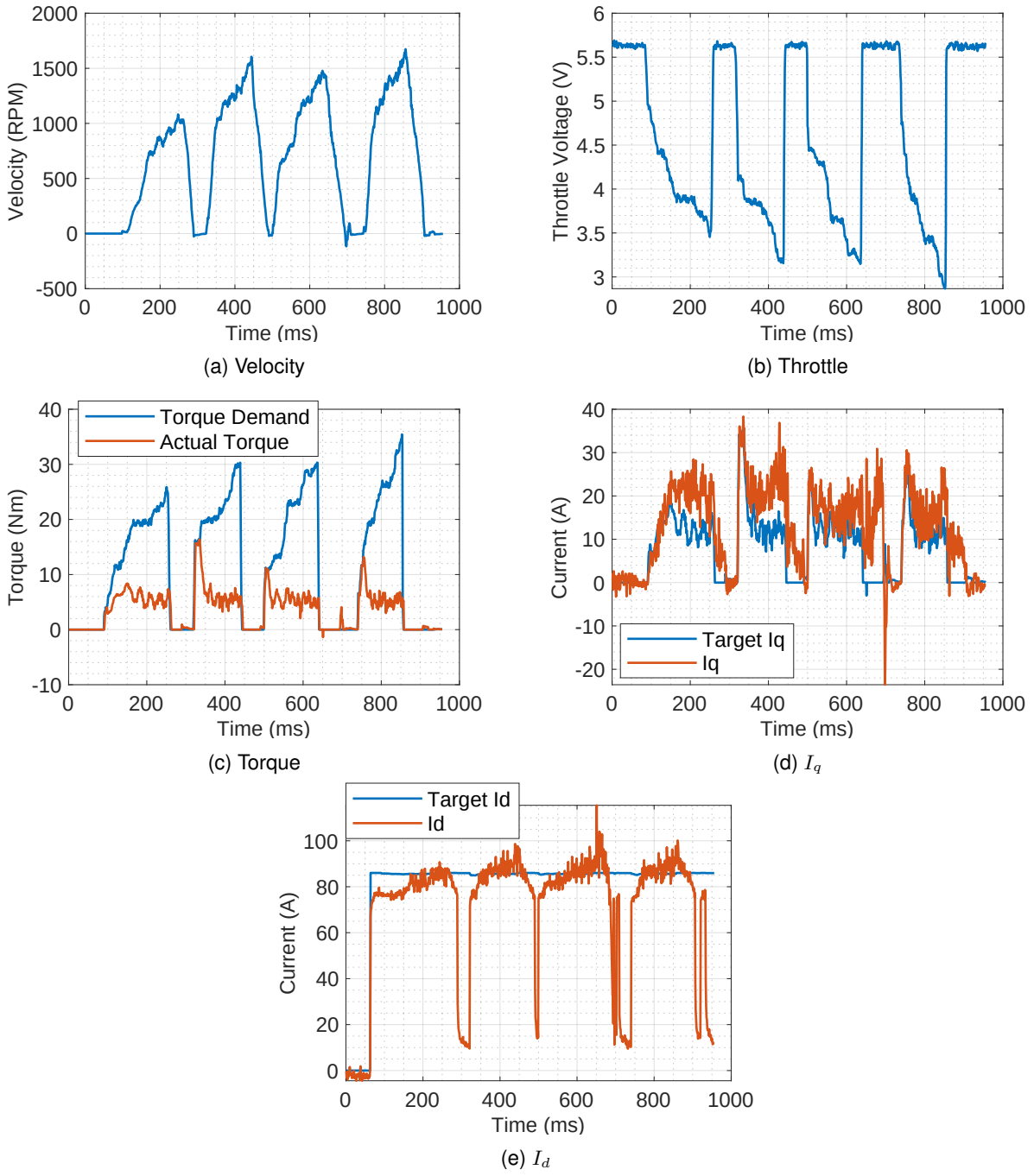
(e) I_d

Figure 4.11: K_p 1 K_i 1 - Wheels up testing

$K_p = 0.01\,, \quad K_i = 0.0001$

With a low K_p value, the small gain results in a small output response to a large input error, this can be seen in Figure 4.12d where the I_q current overshoots the target value, whereas the I_d current, seen in Figure 4.12e has a mixture of overshooting and undershooting. With this proportional gain, the system seems unresponsive to throttle input, comparing the input in Figure 4.12b to the torque values in Figure 4.12c, for the same throttle applied the amount of torque produced is not the same as well with the speed, seen in Figure 4.12a. A very small oscillation can be seen when the throttle is closed off,

specially in the I_q current and torque plots.



(a) Velocity

(b) Throttle

(c) Torque

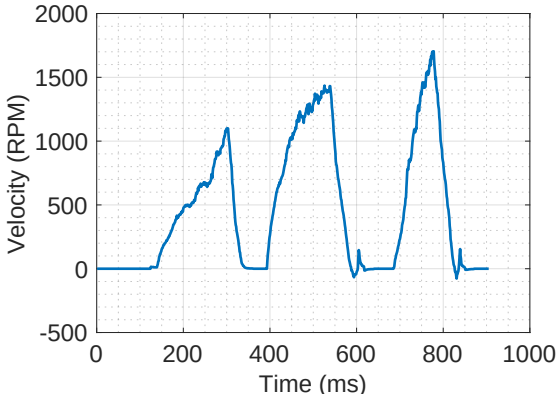
(d) I_q

(e) I_d

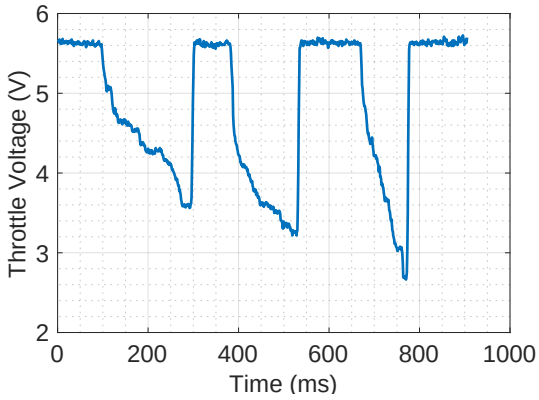
Figure 4.12: K_p 0.01 K_i 0.0001 - Wheels up testing

$$K_p = 0.01\,, \quad K_i = 0.001$$

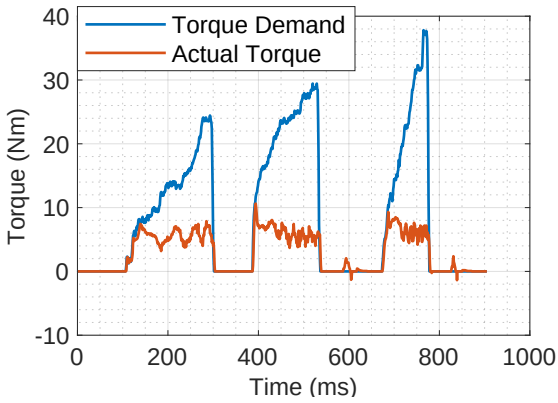
In this test with an increase in the integral gain, it can be see from Figures 4.13d and 4.13e that both respectively, I_q and I_d currents, match the target values, however big oscillations are noticeable when the throttle is closed Figure 4.13b. Regardless of these big offsets, no major torque or velocity are produced when the throttle is closed, seen in respectively Figures 4.13c and 4.13a.



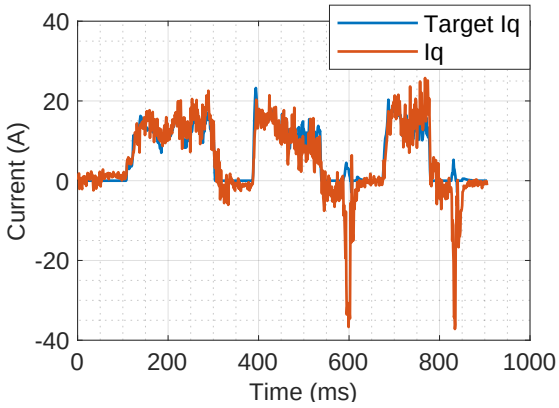
(a) Velocity



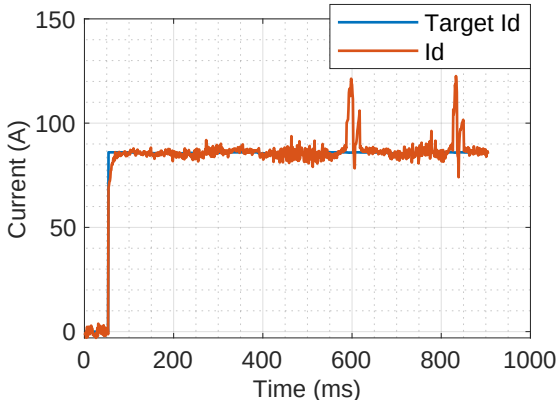
(b) Throttle



(c) Torque



(d) I_q



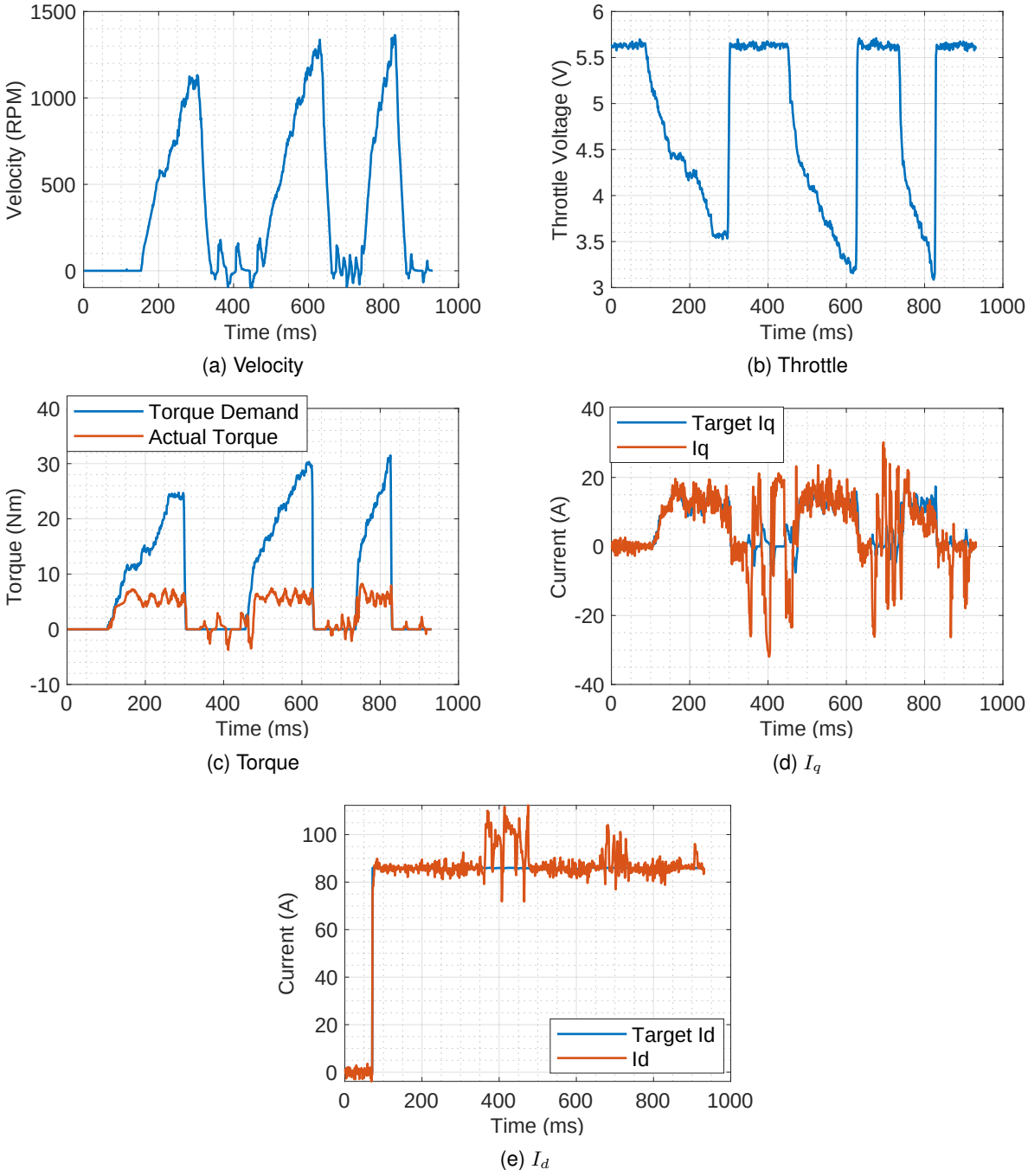
(e) I_d

Figure 4.13: K_p 0.01 K_i 0.001 - Wheels up testing

$K_p = 0.01\,, \quad K_i = 0.004$

From Figure 4.14a it can be seen that there is oscillations in the measured speed whenever the throttle is closed off, seen in Figure 4.14b. These oscillations are bigger and are in fact noticeable during the experiments since the motor tries to engine break, the same issue can be seen in the torque produced in Figure 4.14c. However both I_q and I_d currents, seen in Figures 4.14d and 4.14e respectively, match their target values with some oscillations that occurred similar with the $K_p = 0.01 K_i = 0.001$ setting. These current oscillations in I_d and I_q are responsible for the oscillations and instability of the IM. This is more noticeable in the I_q, whenever the system closes off the throttle, the I_q goes to negative values,

which, is an indicative sign of engine break.



(a) Velocity

(b) Throttle

(c) Torque

(d) I_q

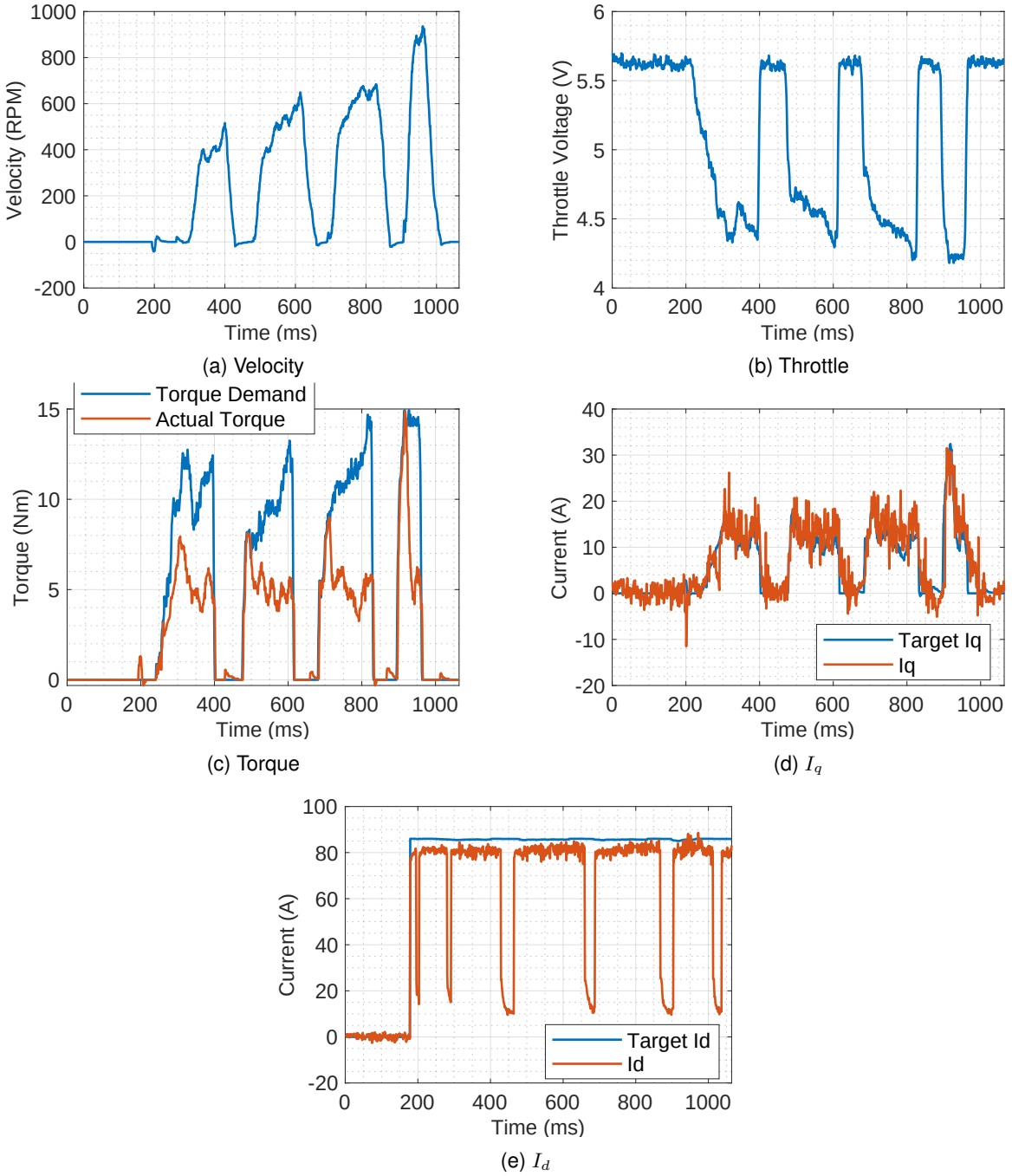
(e) I_d

Figure 4.14: K_p 0.01 K_i 0.004 - Wheels up testing

$K_p = 0.02\,, \quad K_i = 0$

From Figure 4.15a and 4.15b it can be seen that the motor accelerates smoothly with the decrease of the throttle input voltage, the ramp up and ramp down rates of acceleration and deceleration, accordingly, are observed. From Figure 4.15c, some torque can be observed when no throttle is given, however the engine braking was not noticeable when the test was performed. From Figure 4.15e the I_d current does not match the target and a slight offset can be seen, it is important to notice that the I_d current measured by the inverter does not reach 0 when no throttle input is given. From Figure 4.15d some noise is still

present, and it is oscillatory especially in the zones were no throttle input is given.



(a) Velocity

(b) Throttle

(c) Torque

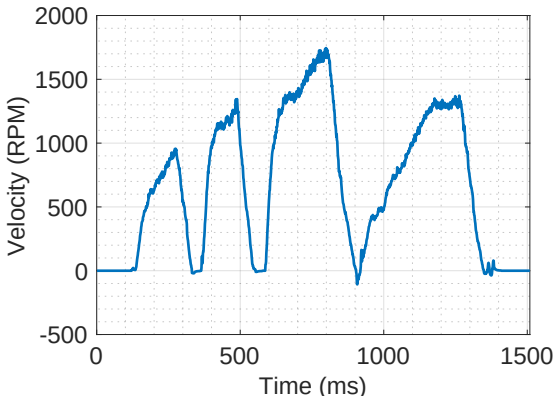
(d) I_q

(e) I_d

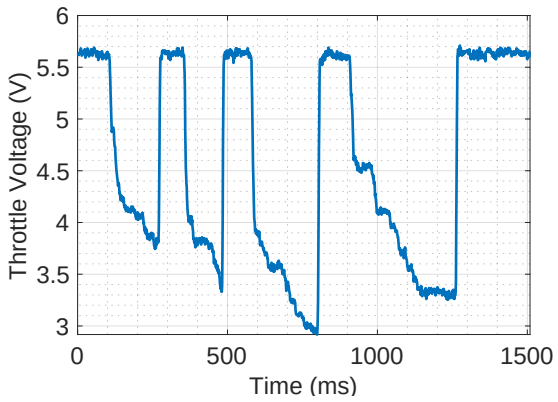
Figure 4.15: K_p 0.02 K_i 0 - Wheels up testing

$$K_p = 0.02\,, \quad K_i = 0.0001$$

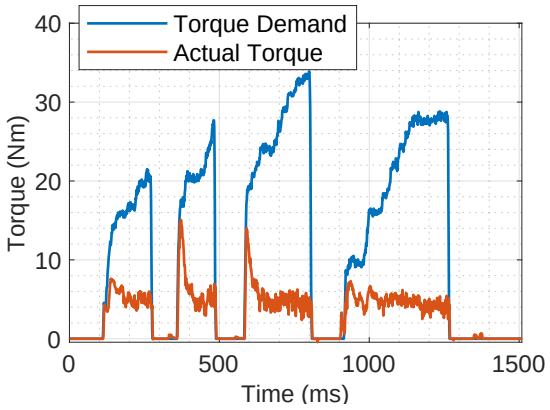
A similar result compared to the $K_p = 0.02$ and $K_i = 0$. From Figure 4.16a it can be noted some speed detection when the throttle is completely shut off. Since the integral gain is no longer 0, the affect can be noticed in Figure 4.16c, where the actual torque does not oscillate as much as previous experiments when integral gain was set to 0. However, in Figure 4.16d a considerable amount of noise can be seen at the end of the plot, although no engine brake was observed during the test. Similar to previous test, Figure 4.16e shows that the I_d current measured still sits below the target.



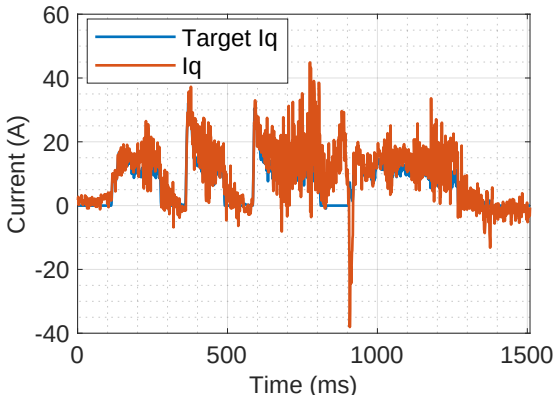
(a) Velocity



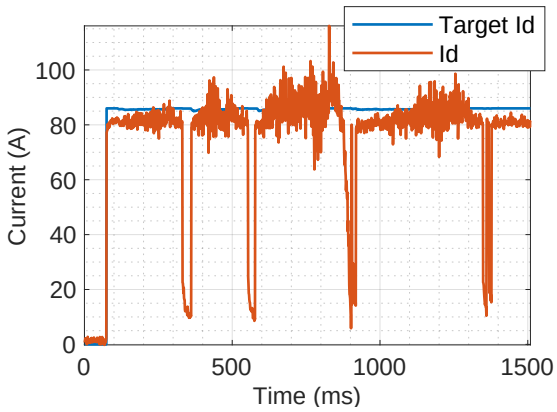
(b) Throttle



(c) Torque



(d) I_q



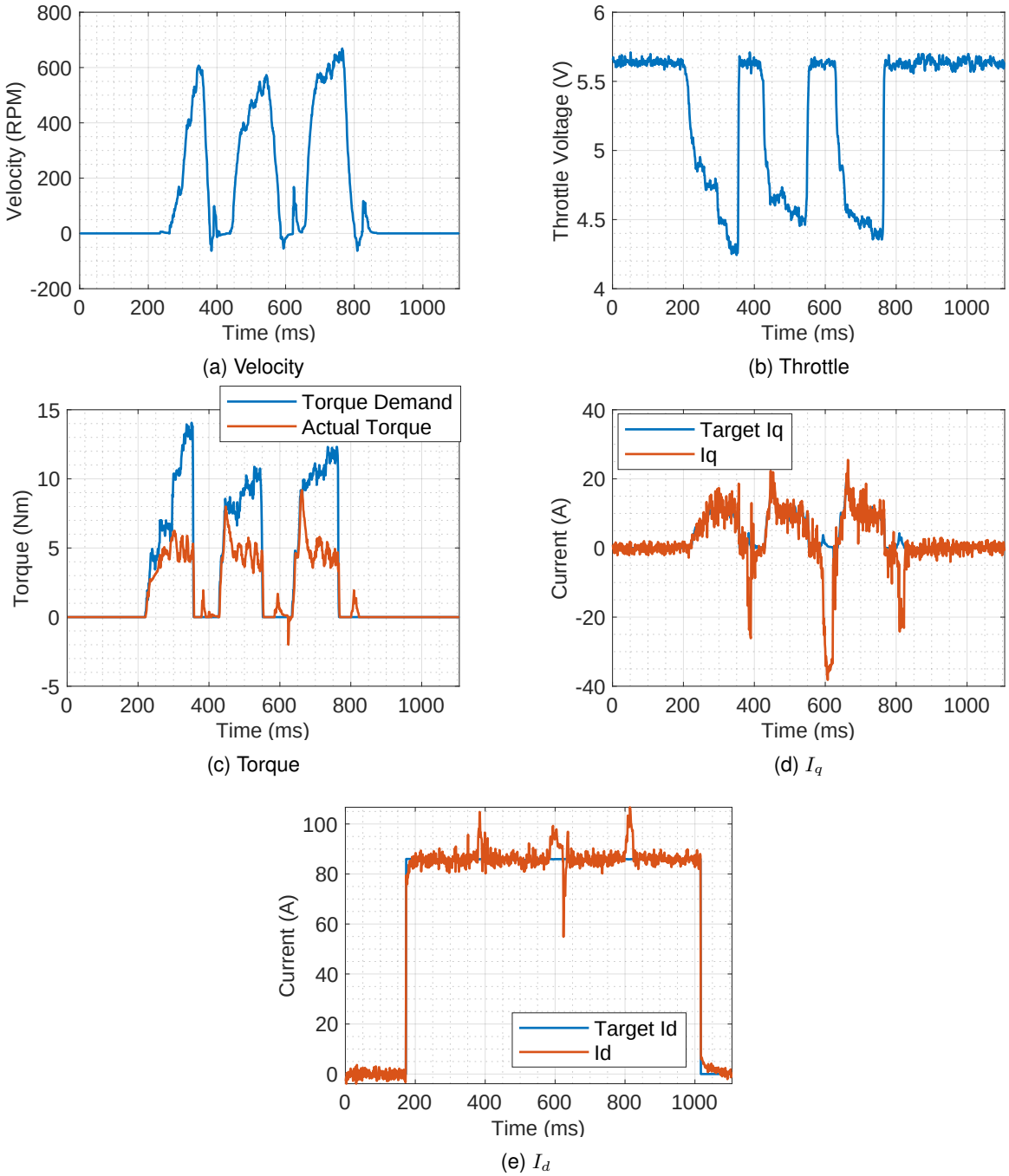
(e) I_d

Figure 4.16: K_p 0.02 K_i 0.0001 - Wheels up testing

$$K_p = 0.02\,, \quad K_i = 0.001$$

With an increase in the integral gain, considerable changes can be seen. From Figure 4.17a, it can be observed that the motor is spinning even when no throttle is being used. Figure 4.17b. Bigger oscillations can be seen in the actual torque as well as a small but significant torque being produced when no throttle is used Figure 4.17c. From Figure 4.17e it can be seen that the current I_d matches the target and less noise and harmonics are generated compared to previous experiments. The same applies to the I_q current. From the data shown in Figure Figure 4.17d it can be seen that there is less noise and harmonics compared to, for example Figure 4.16d. Also important to note the negative value

reading from the inverter in i_q current, compromising the motor behavior. With this PI set up some engine break is felt on the motor and the wheels go in reverse direction.



(a) Velocity

(b) Throttle

(c) Torque

(d) I_q

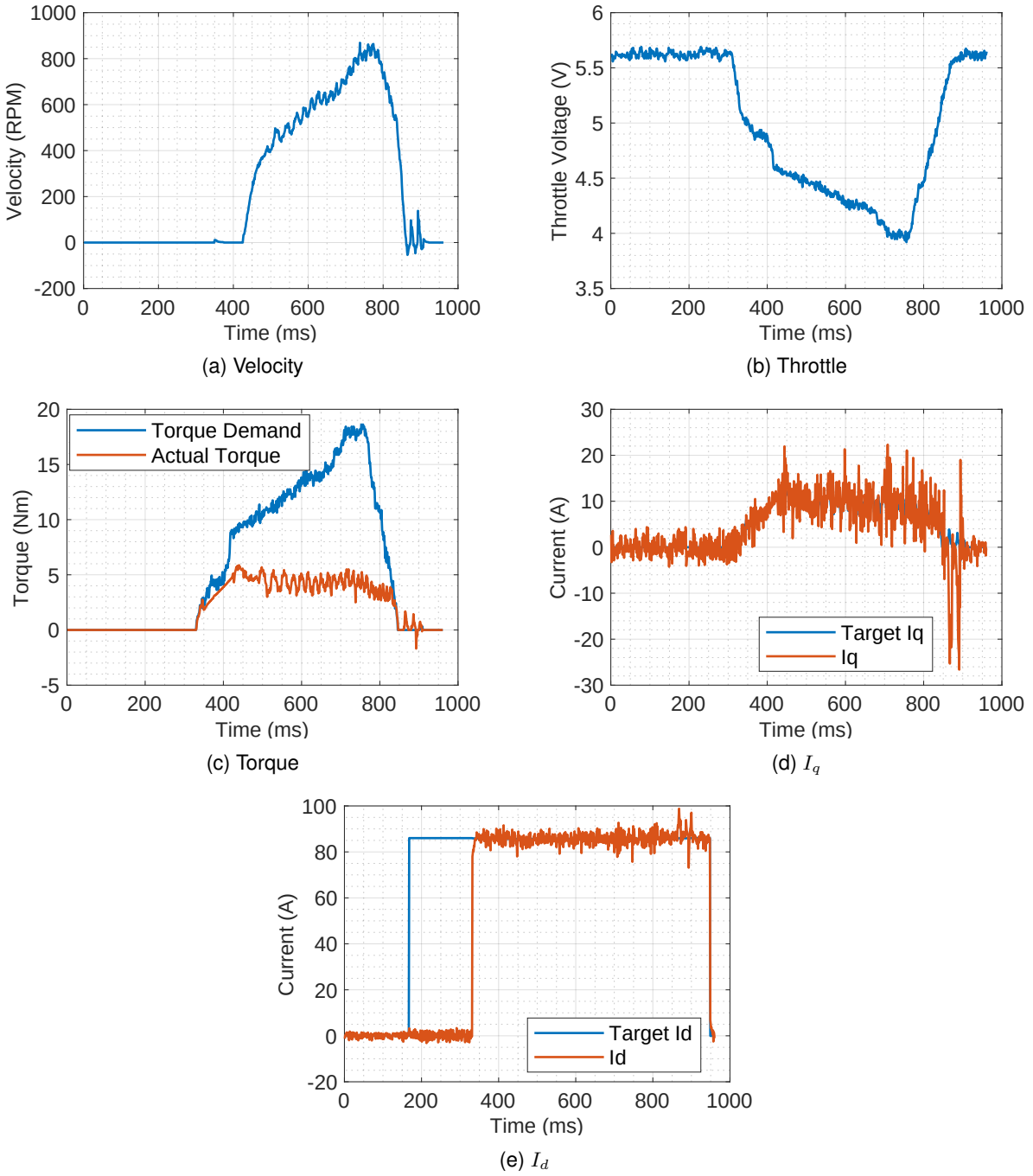
(e) I_d

Figure 4.17: K_p 0.02 K_i 0.001 - Wheels up testing

$K_p = 0.02\,, \quad K_i = 0.003$

From Figure 4.18a it can be seen that the acceleration was smooth and no speed spikes were observed. From Figure 4.18c it can be seen that there is a offset in the torque demand and actual torque values, this is due to the fact that the wheels are elevated and not on the ground, so the torque produced equivalent to the throttle input does not match the actual torque in the wheels. From Figures 4.18d and 4.18e, noise in the values is still noticeable. It is important to note that at the end once no

throttle reference was given, I_q spiked and oscillated, this made the motor engine break and make the wheels go on the reverse direction in order to break, since some torque can be observed at the end of the experiment as well as motor speed. Therefore, another approach when it comes to PI experiments will be conducted, sudden drops in velocity will be performed, closing the throttle abruptly, to observe how the inverter will react, if the engine break when idle is still present.



(a) Velocity

(b) Throttle

(c) Torque

(d) I_q

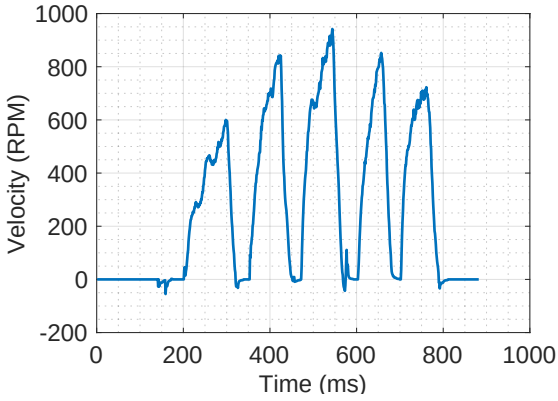
(e) I_d

Figure 4.18: K_p 0.02 K_i 0.003 - Wheels up testing

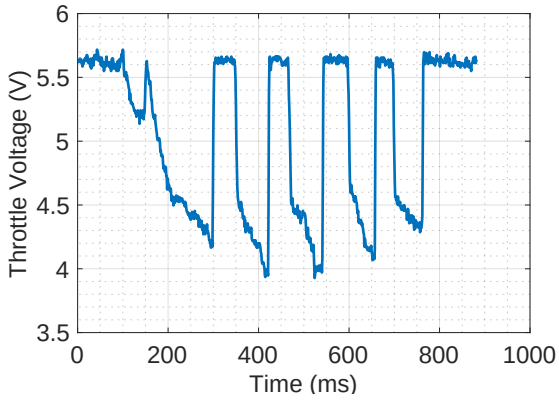
$$K_p = 0.02\,, \quad K_i = 0.004$$

A significant increase in the integral gain is performed however, analyzing Figure 4.19c less actual torque overshoot is read compared to Figure 4.17c which also correlates to less overshoot in velocity

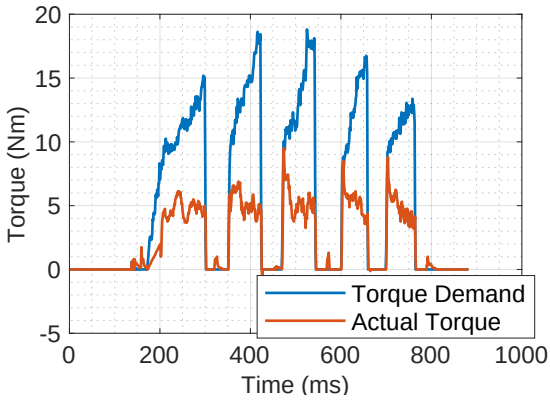
readings, seen in Figure 4.19a. As expected, with the increase on the integral gain, I_d in Figure 4.19e responds quicker to reach the target value compared to Figure 4.17e. However, more oscillations are observed in Figure 4.17d, where current I_q oscillates. With the increase in the K_i, engine braking is more prone to happen.



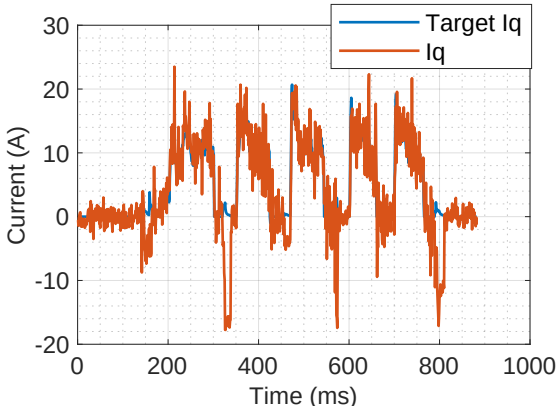
(a) Velocity



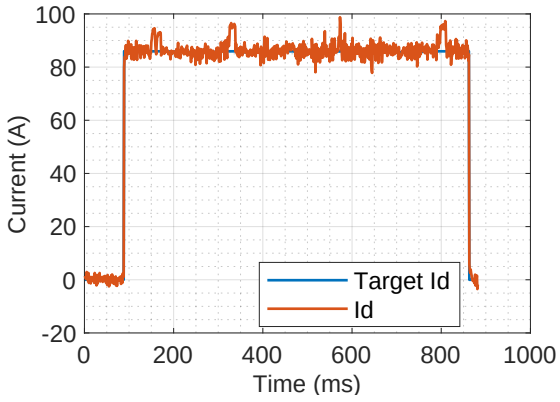
(b) Throttle



(c) Torque



(d) I_q



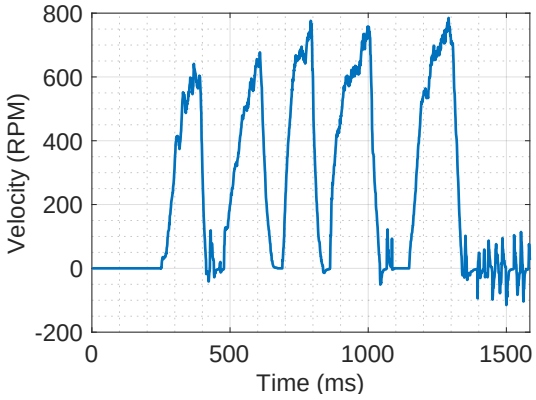
(e) I_d

Figure 4.19: K_p 0.02 K_i 0.004 - Wheels up testing

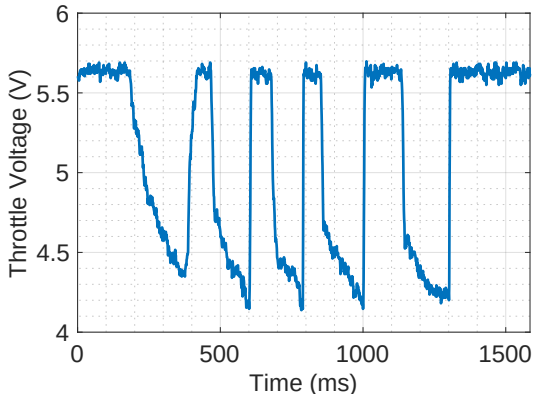
$K_p = 0.02\,, \quad K_i = 0.006$

With this set of combinations, it becomes a turning point where the motor becomes unstable and unresponsive. Big oscillations are observed in current I_q Figure 4.20d, current I_d well above it's target Figure 4.20e, and a significant torque and motor speed being produced seen in Figures 4.20c and 4.20a,

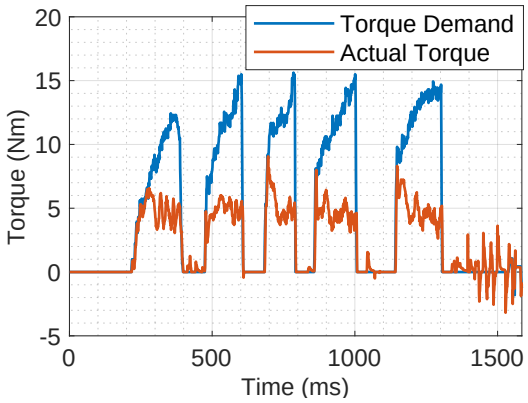
respectively.



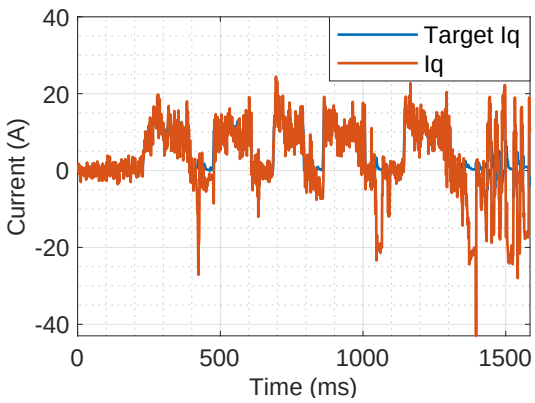
(a) Velocity



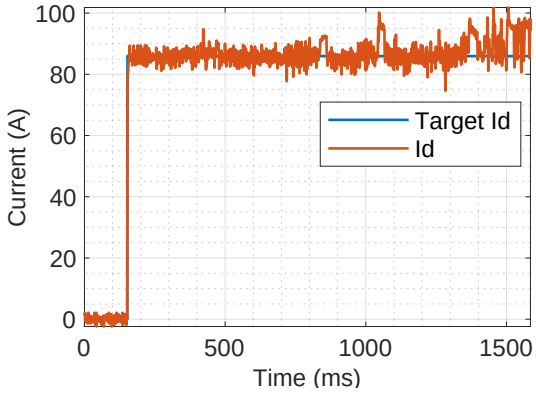
(b) Throttle



(c) Torque



(d) I_q

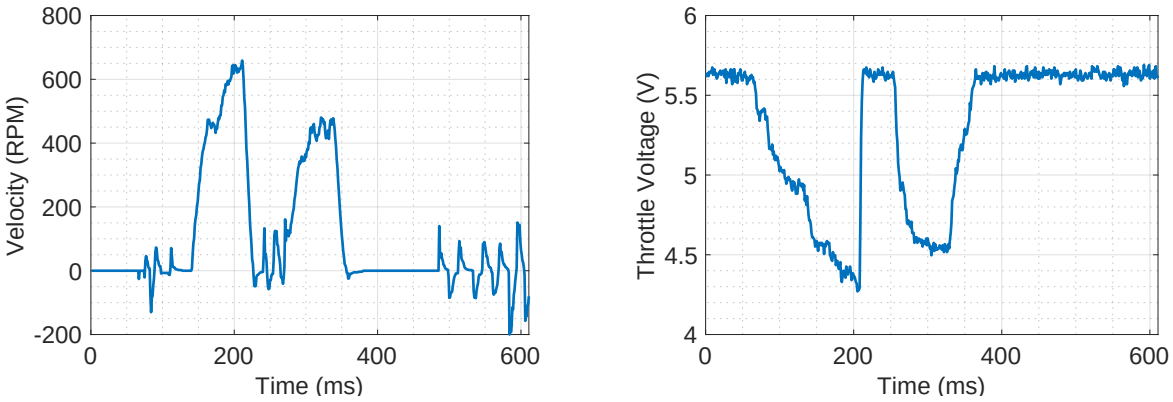


(e) I_d

Figure 4.20: K_p 0.02 K_i 0.006 - Wheels up testing

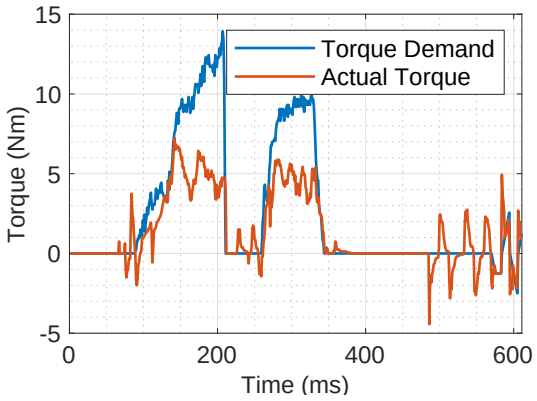
$K_p = 0.02\,, \quad K_i = 0.01$

Since the previous test was somewhat unsuccessful, the motor become unresponsive and unstable, an increase of the integral gain will increase the previous affect. As expected, without pressing the throttle, the motor becomes responsive and torque is already being produced, seen in Figure 4.21c as well as some motor speed is detected Figure 4.21a. It is observed that the motor loses complete stability and the amount of engine braking enhances, destabilizing the car in the process. Greater oscillations can be observed in the I_q and I_d currents, Figures 4.21d and 4.21e respectively.

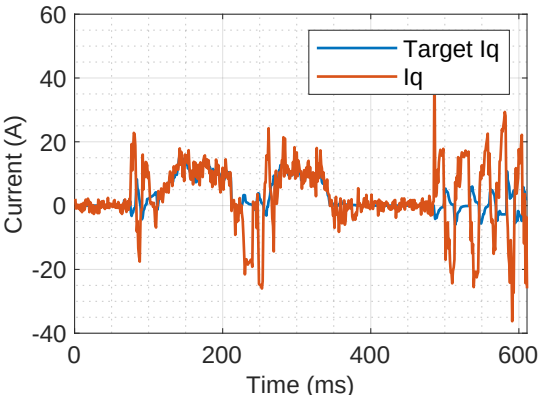


(a) Velocity

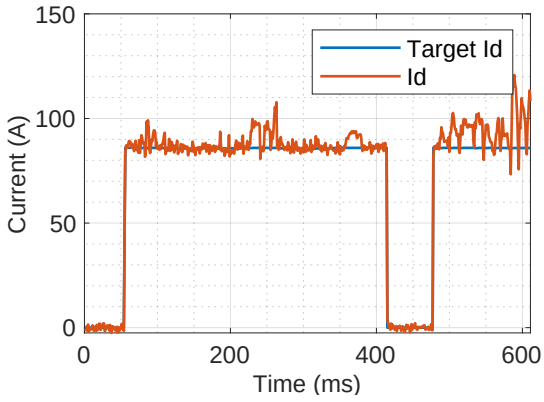
(b) Throttle



(c) Torque



(d) I_q



(e) I_d

Figure 4.21: K_p 0.02 K_i 0.01 - Wheels up testing

The inverter working with a high integral gain, results in big oscillations in the values, at a point where the motor becomes unstable even without a torque reference (touching the throttle). It can be concluded that the inverter works well when the integral gain is set to approximately 10% of what the base proportional gain is set.

4.2.2 Road testing

From the best combinations obtained in Chapter 4.2.1, experimental tests were done outside in a parking lot. The following trajectory of the tests can be seen in Figure 4.22, where the yellow dot represents the starting point. Which can also be viewed in a video for better reference [1].

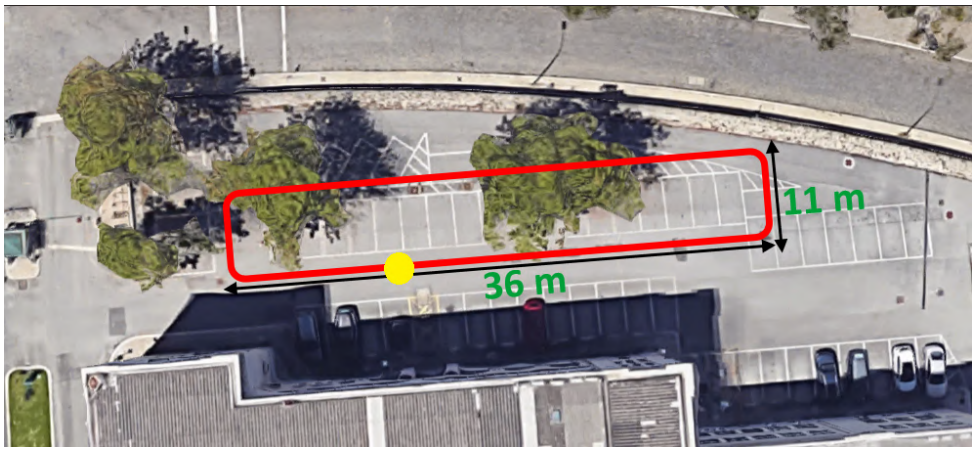


Figure 4.22: Car trajectory.

With the velocity registered by the motor, the total distance traveled can be estimated by integrating the speed of the vehicle. First, the value of the angular velocity of the must be known, which can be obtained from (4.11):

$$\omega_m = \frac{2\pi}{60} * motor_{speed} \tag{4.11}$$

knowing the gearbox relation, which is 8 we can use the following (4.12):

$$\omega_{wheel} = \frac{\omega_m}{g_r} \tag{4.12}$$

From (4.11) and (4.12) and the value of the wheel radius, r_w, the vehicle speed in Km/h can be obtained by (4.13):

$$v = \frac{\omega_{wheel}}{r_w} \tag{4.13}$$

integrating (4.13) the total distance traveled can be obtained as such:

$$distance = \int_0^t v\, dt \tag{4.14}$$

$K_p = 0.02\,, \quad K_i = 0.002$

With a K_i of 10% of the K_p, the first set of experiments are conducted. From Figure 4.23, the velocity is given in Km/h and the average speed is 25 Km/h.

Important to notice that the total distance traveled does not match the expected value of the sum of the distances measured from the trajectory map given in Figure 4.22. For this reason distance measured will not be compared from the other tests. Several issues could be causing this such as: noise in the encoder readings giving wrong measurements in the velocity registered, different and unpredictable

[1] Reference video

CANopen readings from the PDO and also the fact that whenever the throttle is closed off, the inverter signals improper readings.

The actual torque is the amount of torque being produced by the motor while the torque demand is the amount of torque being requested by the throttle.

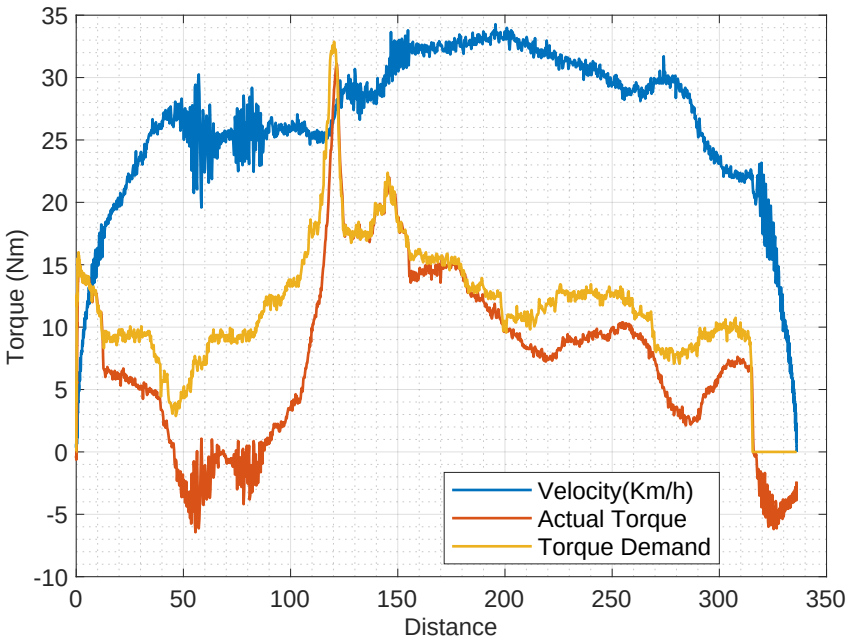
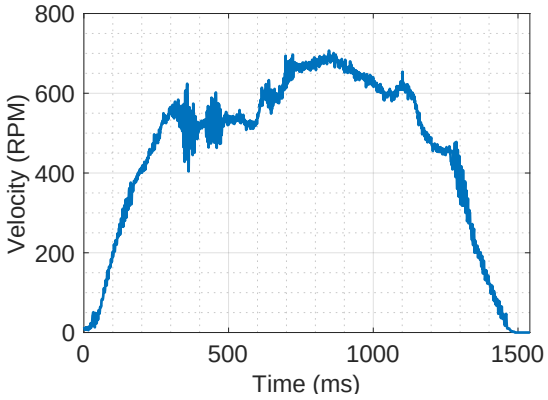
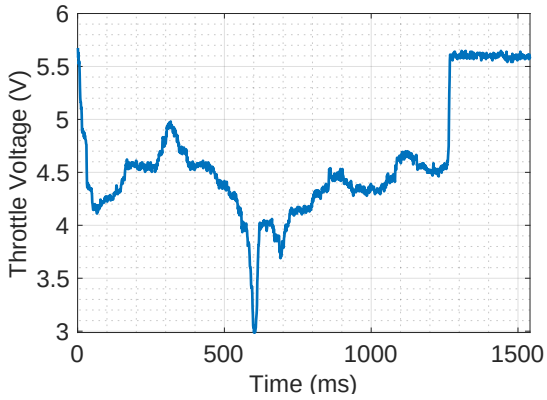


Figure 4.23: Driving test with K_p 0.02 K_i 0.002.

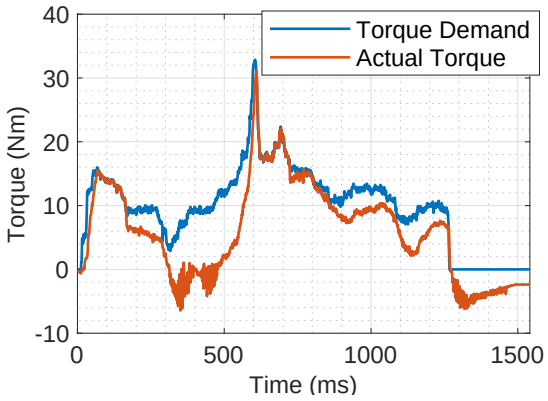
Observing this experiment in more detail, whenever the throttle, Figure 4.24b, is slightly reduced, speed Figure 4.24a and torque Figure 4.24c oscillations can be observed, which in the case of the actual torque, it registers negative values, this increase of oscillation could be caused by the natural decline that there is in the road of the parking lot. From Figure 4.24d, current I_q follows the target value, however when the vehicle comes to a stop, big oscillations occur, which give out negative values. Big oscillations can also be observed in Figure 4.24e where the I_d current does not oscillate near the target value, and contrary to what happens to the I_q current, the current measured overshoots to over 150 A.



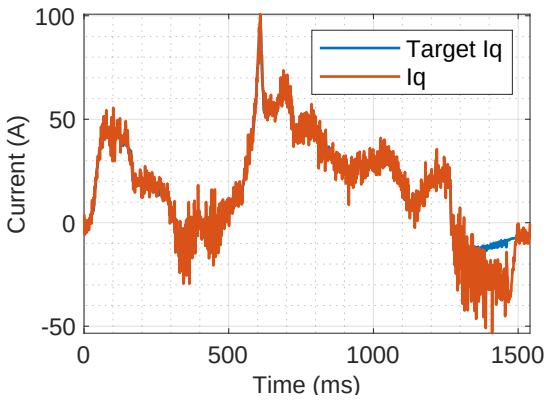
(a) Velocity



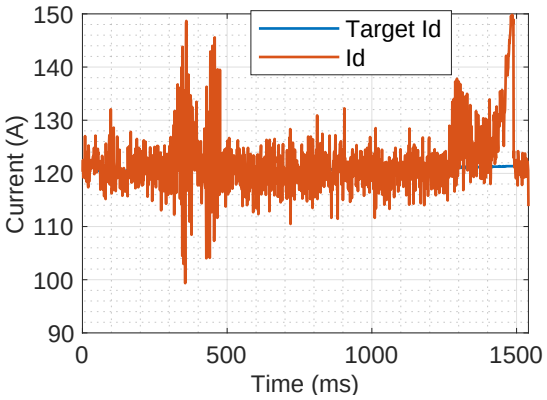
(b) Throttle



(c) Torque



(d) I_q

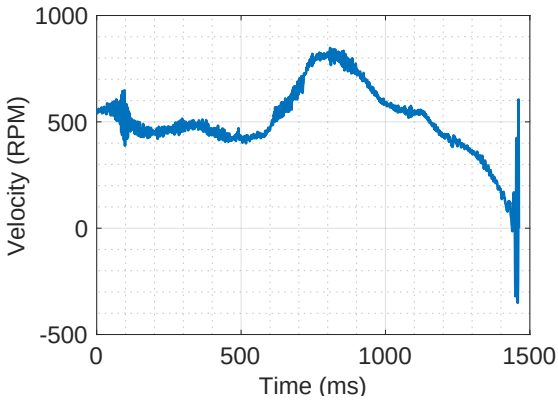


(e) I_d

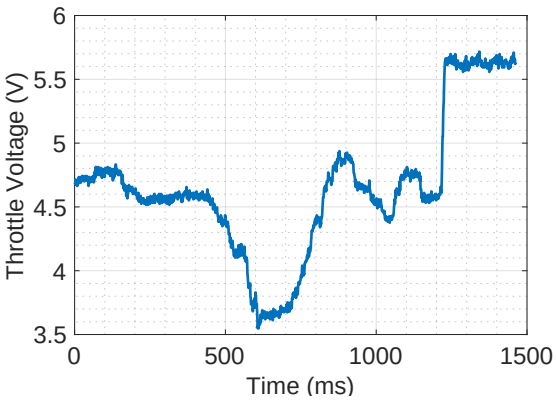
Figure 4.24: K_p 0.02 K_i 0.002 - Road testing

$K_p = 0.02\,, \quad K_i = 0.02$

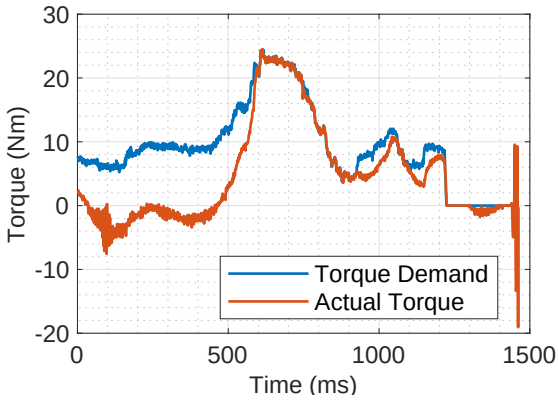
With a significant increase in the integral gain, it is noticeable that accelerations and decelerations are smoother and less harmonics and noise are produced, seen in Figure 4.25a. However, there is a big torque gap observed in figure 4.25c, the actual torque takes longer time to reach the target torque. Compared to Figure 4.25e, the I_d current matches the target current with less harmonics Figure 4.25e, the same can be observed from the I_q current in Figure 4.25d. Only at the end of the experiment when the throttle was closed off, Figure 4.25b the inverted entered in a infinite loop because of the readings, this could be due to the noise coming out of the throttle when idle and thus producing torque.



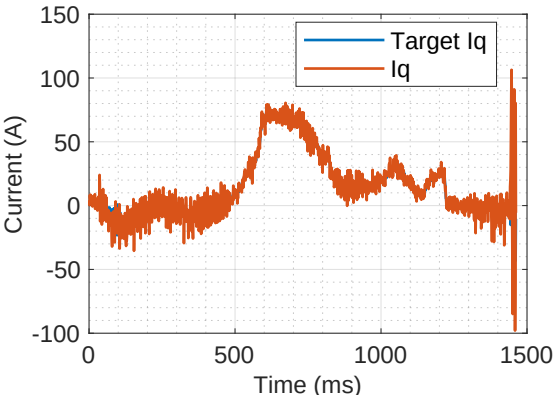
(a) Velocity



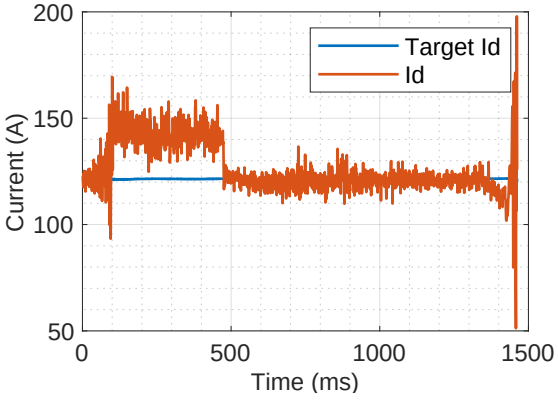
(b) Throttle



(c) Torque



(d) I_q



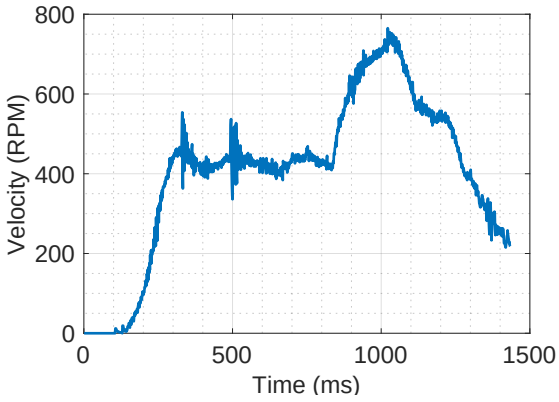
(e) I_d

Figure 4.25: K_p 0.02 K_i 0.02 - Road testing

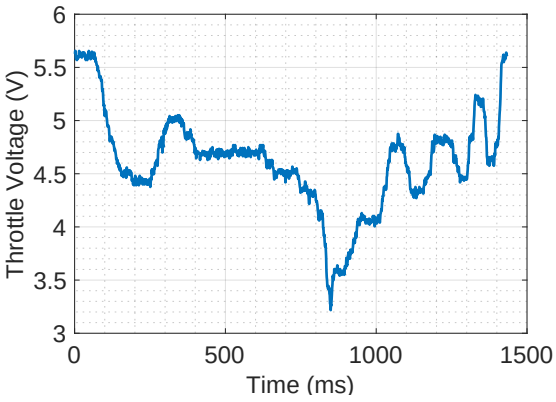
$$K_p = 0.02\,, \quad K_i = 0.003$$

With the integral gain with a value of 15% of the proportional gain, it is seen that from Figures 4.26b and 4.26a, throttle and velocity respectively, that the inverter follows with no major oscillations occurring. It is noticeable that when the throttle is closed off, the inverter takes his time to match the torque demand as seen in Figure 4.26c, when the torque reaches its peak. The current I_d now undershoots the target value seen in Figure 4.26e, not oscillating as much as seen in Figure 4.24e, meanwhile current I_q, in Figure 4.26d overshoots in a small part of the experiment when the inverter tries to catch up when the throttle was closed off but follows the target value without much oscillations, compared to Figure 4.24d

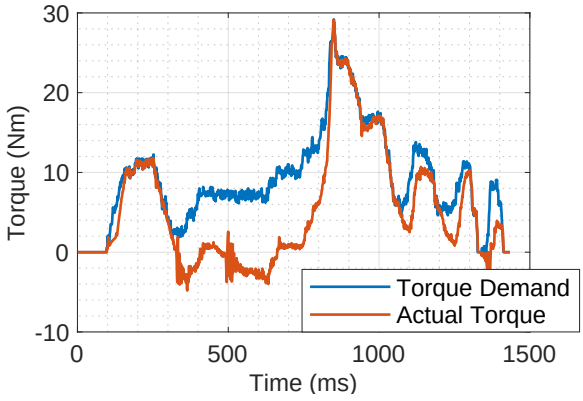
when big oscillations occurred. Comparing these two different situations, a difference from 10% to 15% makes a big impact, especially in the I_{dq} currents, to the point where a lot of noise and oscillations are produced to a steady and precise readings.



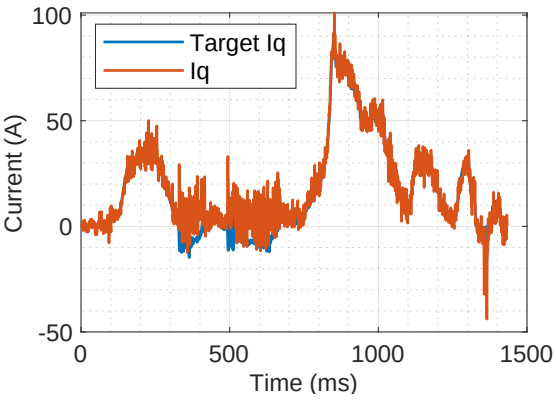
(a) Velocity



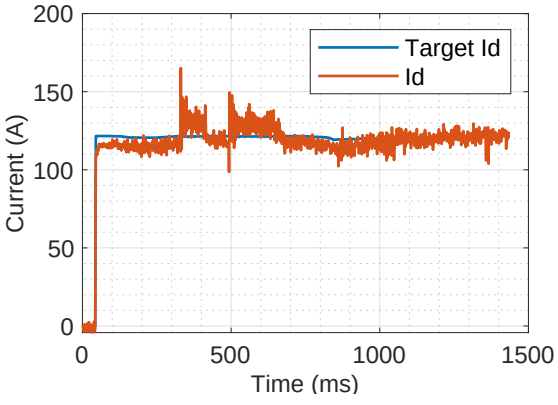
(b) Throttle



(c) Torque



(d) I_q



(e) I_d

Figure 4.26: K_p 0.02 K_i 0.003 - Road testing

4.3 Efficiency and losses analyses

Experiments were conducted to determine the efficiency and losses of the IM. These set of experiments were done with a K_p value of 0.02 and a K_i equal to 0.002 using the same trajectory from Figure 4.22. The rotor resistance, R_r, given in Table 2.3 was changed, an increase of 10% and a decrease of 10%. Knowing from the motor specifications that the standard value is 10.20 $m\Omega$ in order to study how the affect of varying the rotor resistance would affect the efficiency.

Examining the power first, it can be divided in three categories, seen in Figure 4.27, the power straight from the batteries, P_{Bat}, this is the battery voltage times the battery current measured by the inverter. The inverter power, P_{In}, which is the power obtained after the dq transform, which can be calculated in (4.15). The motor power, P_{out}, which can be calculated by (4.16), where v is the motor velocity measured in the encoder, T is the measured torque by the inverter.

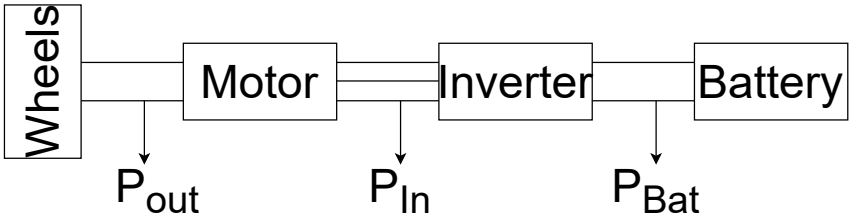
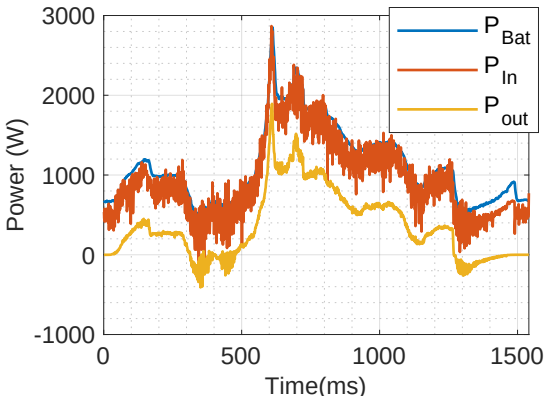


Figure 4.27: Power diagram of the system.

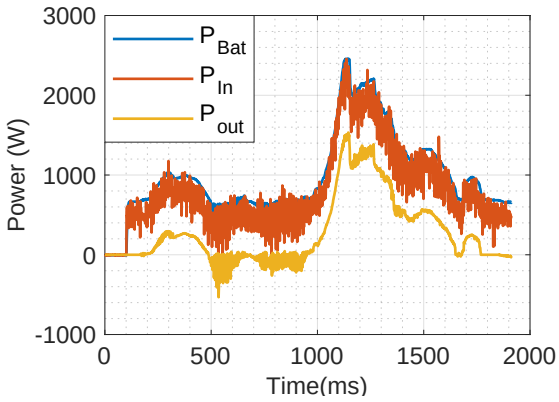
$$P_{in} = U_d I_d + U_q Iq \tag{4.15}$$

$$P_{out} = \omega_m T \frac{2\pi}{60} \tag{4.16}$$

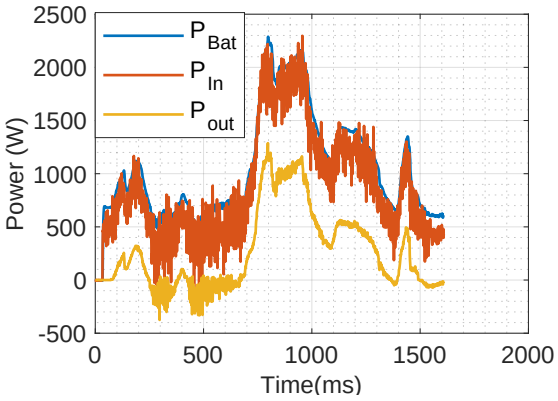
From Figures 4.28a, 4.28b and 4.28c the power obtained from different R_r values can be viewed. It is important to note that the power measured straight from the batteries closely matches the calculated power from equation 4.15. This is important because the power registered from the batteries is the source of power, and with the dq transform reassembling the battery power although some oscillations and noise can be viewed. This is expected, since signals read from the inverter have oscillations and are noisy signals. It is assumed that whenever the power is negative, the engine is breaking, although some power is regenerated from the batteries point of view.



(a) R_r 10 $m\Omega$



(b) R_r 12 $m\Omega$



(c) R_r 8 $m\Omega$

Figure 4.28: K_p 0.02 K_i 0.003 Power testing

Knowing the power, the efficiencies can be obtained. The efficiency of the whole system can be calculated from (4.17), the inverter efficiency can be calculated from (4.18) and the motor efficiency can be calculated from (4.19).

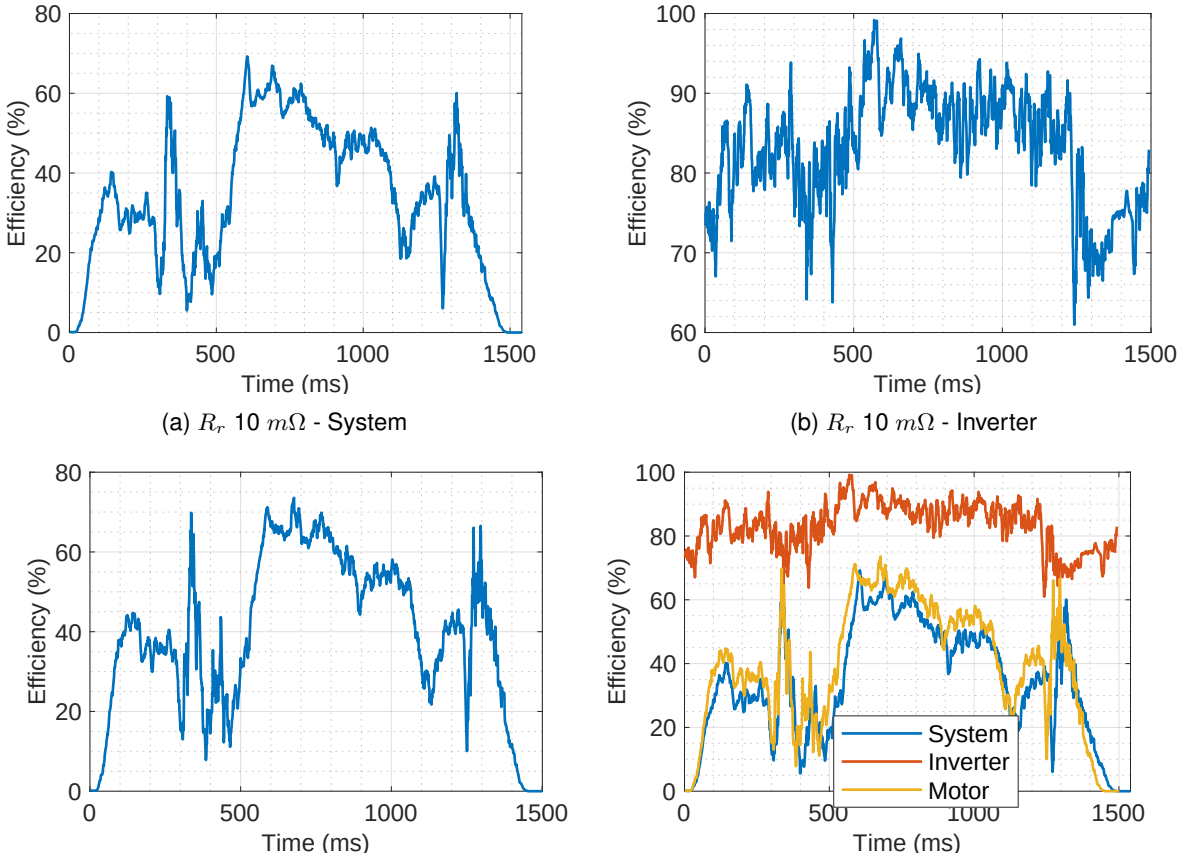
$$\eta_{system} = \frac{P_{out}}{P_{Bat}}.100 \tag{4.17}$$

$$\eta_{inv} = \frac{P_{In}}{P_{Bat}}.100 \tag{4.18}$$

$$\eta_{motor} = \frac{P_{out}}{P_{In}}.100 \tag{4.19}$$

For the initial test, an experiment with the normal R_r value was done, observed in Figure 4.29.

In Figure 4.29a, the system efficiency can be seen. In Figure 4.29b the inverter efficiency can be seen. In Figure 4.29c, the motor efficiency can be seen. In Figure 4.29d the previous plots can be seen for better comparison. It is also important to mention, that the plots were applied a *moving average* during data cleaning (to avoid efficiencies above 100% and below 0%), for a less noisy and more easily perceptible plots.



(a) R_r 10 $m\Omega$ - System

(b) R_r 10 $m\Omega$ - Inverter

(c) R_r 10 $m\Omega$ - Motor

(d) R_r 10 $m\Omega$ - Combination

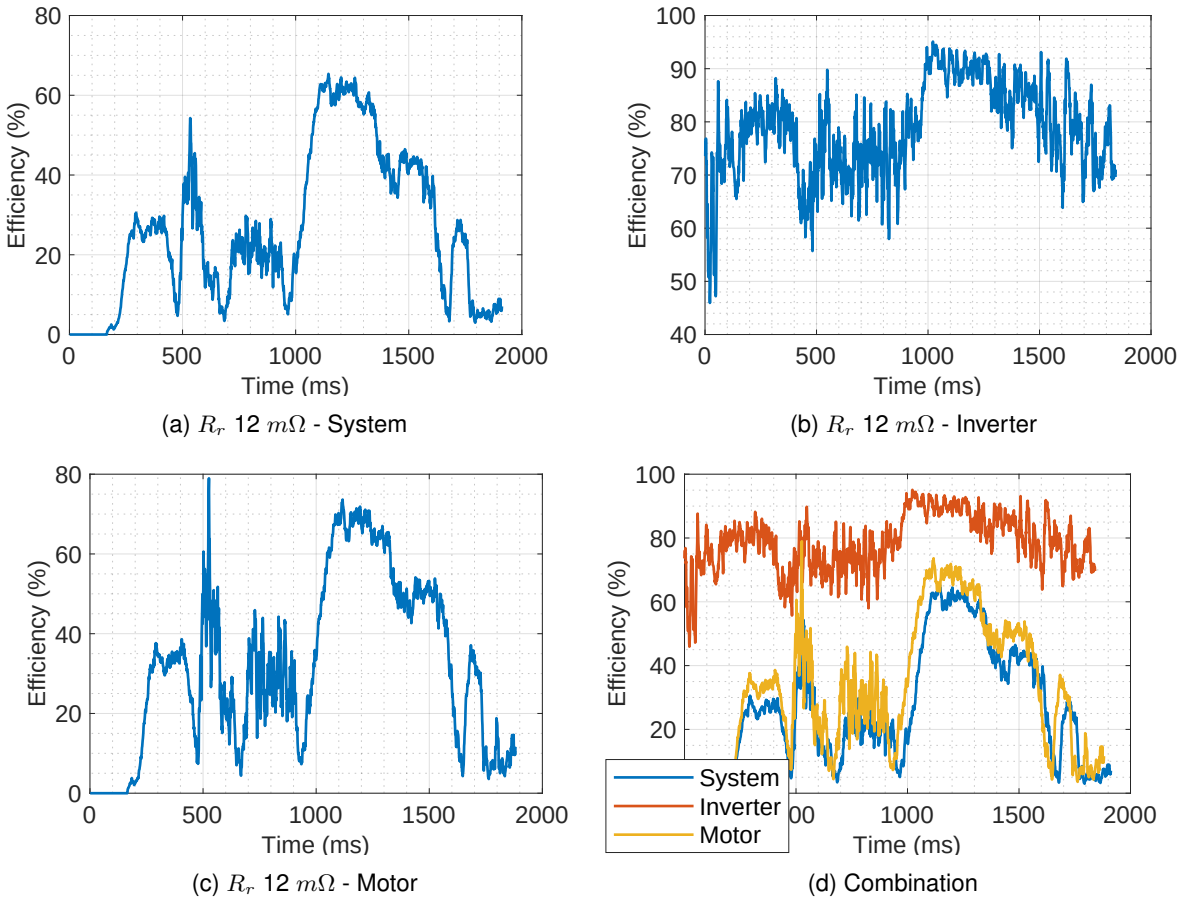
Figure 4.29: K_p 0.02 K_i 0.003 R_r 10 $m\Omega$ efficiency testing

For inverter behavior, this experiment can be viewed in more detail in figure 4.24. Big oscillations occur at the same time when the I_d current oscillates, seen in figure 4.24d as well with the negative torque values seen in 4.24c. Overall with an R_r of 10 $m\Omega$, the mean values for each efficiency can be seen in Table 4.1.

Table 4.1: Efficiency for 10 $m\Omega$.

System	33.91 %
Inverter	83.17 %
Motor	39.41 %

With an increase of 10% of the rotor resistance, Figure 4.30, whenever the throttle is closed off, energy spikes happen and thus, making the system oscillatory. In Figure 4.30a the efficiency of the system can be seen, big oscillations happen when the throttle is closed off. It is noticeable, in Figure 4.30c that the efficiency slightly improved whenever the vehicle moved in higher speeds, however, it's value is lower than to the normal R_r results. In Figure 4.30d the three efficiencies of the system can be seen altogether.



(a) R_r 12 $m\Omega$ - System

(b) R_r 12 $m\Omega$ - Inverter

(c) R_r 12 $m\Omega$ - Motor

(d) Combination

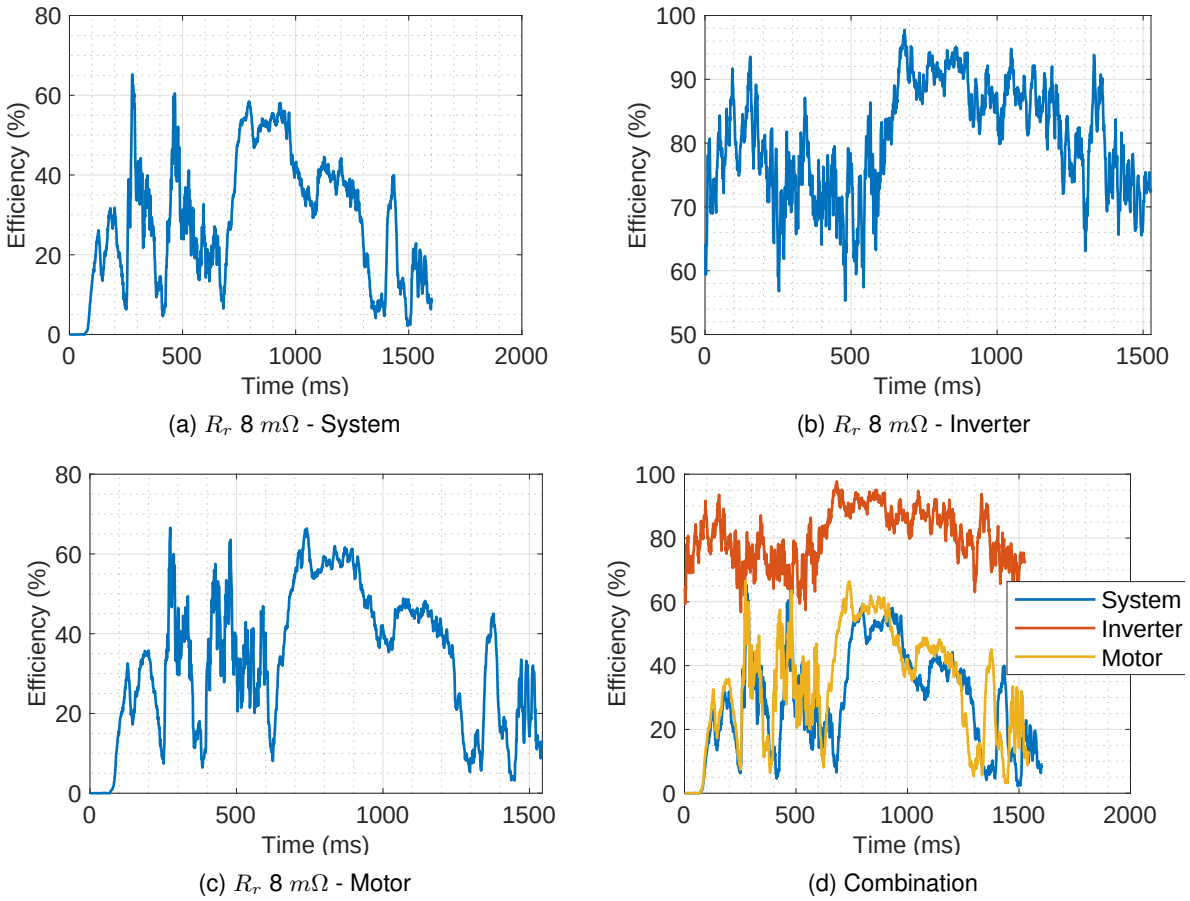
Figure 4.30: K_p 0.02 K_i 0.003 R_r 12 $m\Omega$ efficiency testing

From Table 4.2, the mean value of the efficiencies, observed in Figure 4.30, are shown. It is to note that the system, when compared to normal stated R_r value, the efficiency decreased of all three sectors, the system efficiency decreased 7.92%, the inverter efficiency decreased 3.59% and the motor efficiency decreased 7.53%.

Table 4.2: Efficiency for 12 $m\Omega$.

System	25.99 %
Inverter	79.58 %
Motor	31.88 %

With a decrease of 10% of R_r, the new value becomes 8 $m\Omega$. The plots of this experiment can be seen in Figure 4.31. Compared to the previous test where R_r was equal to 12 $m\Omega$, in the system efficiency in Figure 4.31a, there are more spikes in the oscillations, skyrocketing the efficiency, from Figure 4.31b, there are less oscillations in the signal in the efficiency measured in the inverter and in the motor efficiency, in Figure 4.31c, there are more oscillations in the system. In Figure 4.31d the three efficiencies of the system can be seen altogether.



(a) R_r 8 $m\Omega$ - System

(b) R_r 8 $m\Omega$ - Inverter

(c) R_r 8 $m\Omega$ - Motor

(d) Combination

Figure 4.31: K_p 0.02 K_i 0.003 R_r 8 $m\Omega$ efficiency testing

From Table 4.3, the mean value of the efficiencies, observed in Figure 4.31, are shown. It is to note that the system, when compared to normal stated R_r value, the efficiency decreased of all three sectors, the system efficiency decreased 5.37%, the inverter efficiency decreased 2.62% and the motor efficiency decreased 5.76%.

Table 4.3: Efficiency for 8 $m\Omega$.

System	28.54 %
Inverter	80.55 %
Motor	33.65 %

Comparing the values given in Tables 4.1, 4.2 and 4.3, as expected the value from the nameplate data, $R_r = 10m\Omega$ gives the best efficiency, the decreased 10% value, $R_r = 8m\Omega$ gives the second best efficiency, while the increased 10%, $R_r = 12m\Omega$ gives the worst result.

As for the the analysis of the losses, varying the rotor resistance in the equivalent model of the IM in the inverter, this can make the inverter ask for more or less current, depending on the change of the rotor resistance compared to the actual IM rotor resistance. With this change of current, we can study

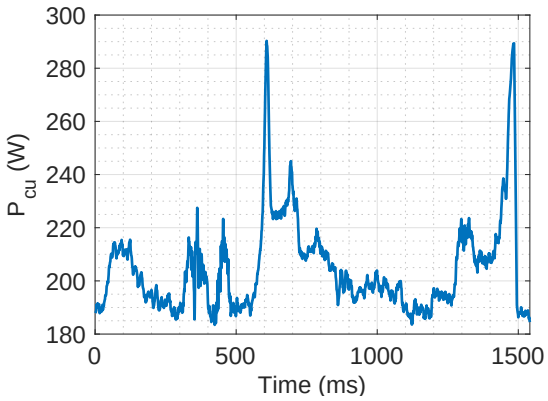
and analyze the copper losses, P_cu, which can be calculated by (4.20)

$$P_{cu} = 3 * r_s * I_s^2 \tag{4.20}$$

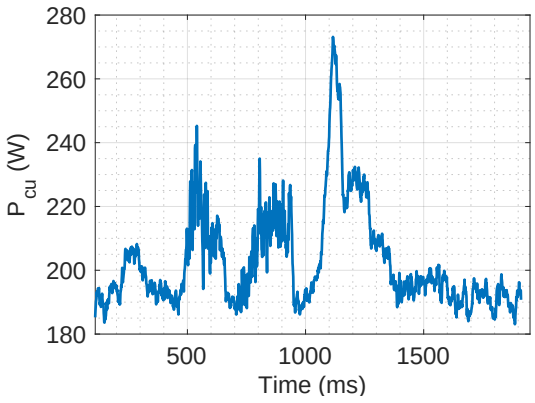
where r_s is the stator resistance and I_s can be calculated by (4.21)

$$I_s = \frac{\sqrt{I_d^2 + I_q^2}}{\sqrt{2}} \tag{4.21}$$

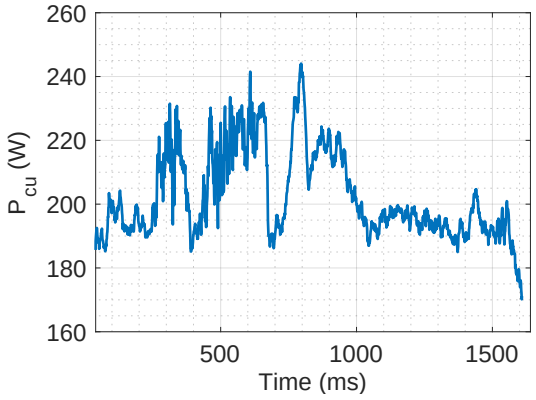
from (4.20) we can obtain three different sets of parameters, corresponding to the changes of the rotor resistance, which can be seen in Figure 4.32. From Figure 4.32a the ”normal” copper losses can be seen, with an average value of 204.59 W. In Figure 4.32b an average value is obtained of 192.10 W and from Figure 4.32c an average of 198.21 W. It is hard to take conclusions from this experiment since a difference of 10 W is minimal.



(a) R_r 10 $m\Omega$ - P_{cu}



(b) R_r 12 $m\Omega$ - P_{cu}



(c) R_r 8 $m\Omega$ - P_{cu}

Figure 4.32: P_{cu} analyses with varying R_r

4.4 PI Correction

In order to correct the oscillations observed in experiments in Chapter 4.2, a trial and error method was implemented. Different sets of values were set until the system managed to stabilize itself.

Also to help stabilize the system, hill hold was disabled, all ramp downs of velocity were set to a minimum, low speed integral gain and low speed proportional gain were set to 0. In Table 4.4, the best combination of set of values can be found.

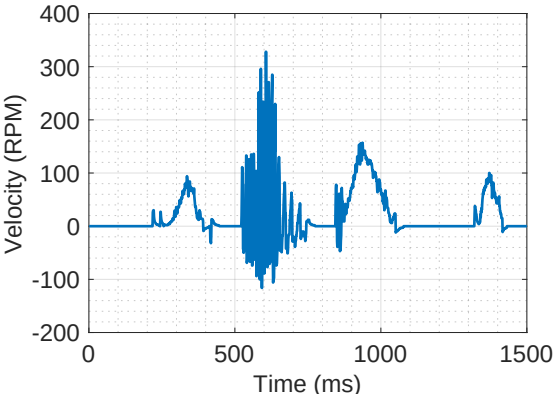
Table 4.4: Set of parameters that result in stabilization.

Proportional gain k_p	0.0097
Integral gain k_i	0.0078
Speed calculation filter pole	0.9499
Current control k_p	0.0094
Current control k_i	0.0004
Voltage control k_p	0
Voltage control k_i	0
Frequency/Mod index control k_p	0.0399
Frequency/Mod index control k_i	0.0199

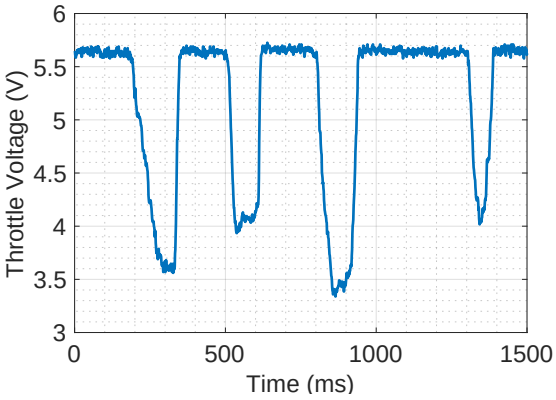
Speed calculation filter pole is used to filter the speed measurement passed to the speed limit and speed control loop. When enabled, it will bypass any existing filtering within the encoder to allow full control of the speed feedback and find a compromise between noise and lag. It is set to the maximum value available.

With the values given in Table 4.4, a small test indoors are performed to test if the inverter responded well to sudden stops of reference torque. The test in question can be seen with detail in Figure 4.33.

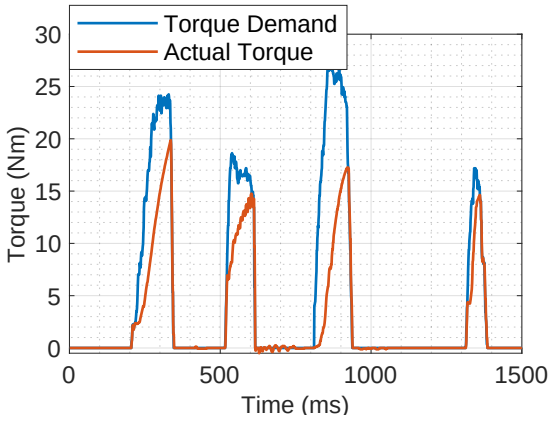
In Figure 4.33a it is observed that only in the second time where the throttle, Figure 4.33b was closed off, the readings oscillated, however after a few seconds it managed to stabilize itself without user input. Some noise is noticeable in Figure 4.33c, where some torque is produced after the second attempt at closing off the throttle. In Figure 4.33d the same issue happens, where more oscillations happen in the same region, else the measured I_q follows the target I_q without much error margin. In Figure 4.33e the I_d current shows some oscillations whenever the throttle is closed off, however the most noticeable one was after the second attempt at closing off the throttle.



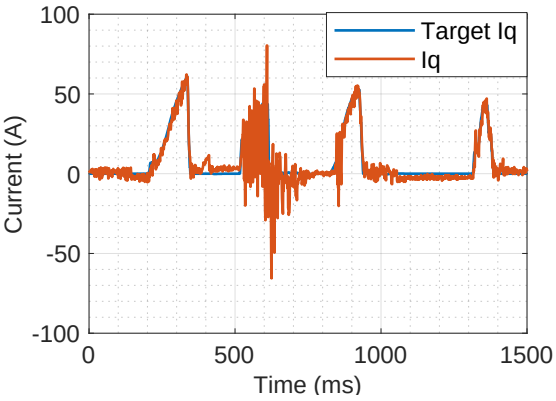
(a) Velocity



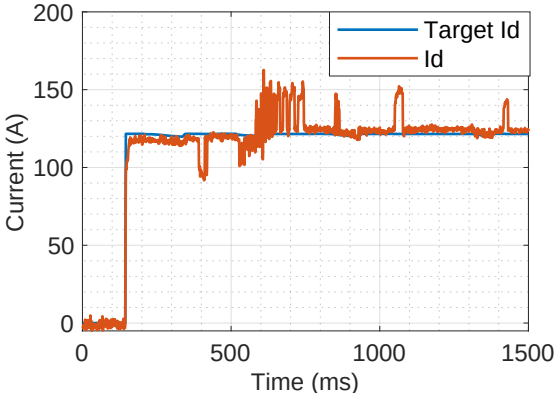
(b) Throttle



(c) Torque



(d) I_q



(e) I_d

Figure 4.33: Stable PI testing

Chapter 5

Conclusions

In this chapter, a summary of the achievements of this thesis is provided, as well as some topics to be developed in future work, following the work conducted and the obstacles encountered.

5.1 Achievements

The objectives set for this thesis were successfully met, however an early objective that was to simulate the VIENA power system in a *Matlab* environment was not concluded due to time constraints.

The new power system acquired for the VIENA project was successfully implemented, with all its required connections and proper switches.

The communication between vehicle and user was using a RPI as a computer with a custom CANopen board that includes a MCP2515 CAN controller microchip with a MCP2551 as CAN transceiver in order to perform CAN communication. All inverter parameters found in the DCF file could be changed and read through *python* programming.

The inverter was somewhat successful in road testing exploring different combinations and settings of the PI gains. Several issues occurred during experiments, the two most common being the fact that when driving in low speed or when the throttle is almost null the inverter enters in a confused state between idle state and driving state, making the motor spasm, producing negative torque. The other most common issue is the fact that, whenever the vehicle goes downhill, the inverter tries to engine break. The first issue by the time this thesis was written was successfully fixed, however the latter has not been fixed yet.

5.2 Future Work

With a good baseline set for the new VIENA power train system, more diverse research topics can be done in the VIENA project. The first issue to solve would be the downhill engine break so that the vehicle would be totally stable and safe for driving when conducting all sorts of tests.

As mentioned, the way the inverter was programmed, it lacks an interface for easy access and visualization. Most of the back-end coding was done, so only an interface would remain. This would prove useful to VIENA project, besides used in research topics, also be used for pedagogical topics for group of students to use. An interface would also be mandatory for an autonomous driving state, as it is one of the main goals of the VIENA project.

Bibliography

[1] International Energy Agency. Global EV Outlook 2021. [Online]. Available: https://www.iea.org/reports/global-ev-outlook-2021, 2021. [Accessed: Feb. 1, 2022].

[2] S. Jape and A. Thosar. Comparison of electric for electric vehicle application. *International Journal of Research in Engineering and Technology*, 6:12–17, sep 2017. DOI 10.15623/ijret.2017.0609004.

[3] Y. Gao, M. Ehsani, and J. Miller. Hybrid Electric Vehicle: Overview and State of the Art. *IEEE Access*, 1:307–316, 2005. DOI 10.1109/ISIE.2005.1528929.

[4] C. Chan and K. Chau. An overview of power electronics in electric vehicles. *IEEE Transactions on Industrial Electronics*, 44(1):3–13, 1997. DOI 10.1109/41.557493.

[5] T. Zarma, A. Galadima, and M. Aminu. Review of Motors for Electric Vehicles. *Journal of Scientific Research and Reports*, 24(6):1–6, oct 2019. DOI 10.9734/jsrr/2019/v24i630170.

[6] E. Grunditz and T. Thiringer. Performance Analysis of Current BEVs Based on a Comprehensive Review of Specifications. *IEEE Transactions on Transportation Electrification*, 2(3):270–289, 2016. DOI 10.1109/TTE.2016.2571783.

[7] K. Rajashekara. Present Status and Future Trends in Electric Vehicle Propulsion Technologies. *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 1(1):3–10, 2013. DOI 10.1109/JESTPE.2013.2259614.

[8] R. Cuenca, L. Gaines, and A. Vyas. Evaluation of electric vehicle production and operating costs. may 2000. DOI 10.2172/764206.

[9] A. E. Fitzgerald, J. C. Kingsley, and S. D. Umans. *Electric Machinery*. McGraw Hill Higher Education, 2003. ISBN 0-07-112193-5.

[10] A. R. Munoz, T. A. Lipo, and D. Novotny. A new induction motor V/f control method capable of high-performance regulation at low speeds. *IEEE Transactions on Industry Applications*, 34:813–821, aug 1998. DOI 10.1109/28.703982.

[11] L. Alberti, N. Bianchi, and S. Bolognani. Field oriented control of induction motor: A direct analysis using finite element. *2008 34th Annual Conference of IEEE Industrial Electronics*, pages 1206–1209, 2008. DOI 10.1109/IECON.2008.4758126.

[12] A. Yousef and S. A. Maksoud. Review on Field Oriented Control of Induction Motor. [Online]. Available: https://www.researchgate.net/publication/282859207, 2015. [Accessed: Sep. 29, 2022].

[13] CiA. History of CAN technology. [Online]. Available: https://www.can-cia.org/can-knowledge/can/can-history/, 2021. [Accessed: Sep. 14, 2022].

[14] CiA. CANopen Technical Documents. [Online]. Available: https://www.can-cia.org/groups/specifications/, 2022. [Accessed: Sep. 14, 2022].

[15] Sevcon. Gen4 Applications Reference Manual. [Online]. Available: https://www.manualslib.com/products/Sevcon-Gen4-10340478.html, 2011. [Accessed: Aug. 13, 2021].

[16] S. N. Manias. *Power Electronics and Motor Drive Systems*. Academic Press, 2017. ISBN 978-0-12-811798-9.

[17] F. R. L. Jesus. Design of a Supercapacitor/Battery Hybrid Energy Storage System for the TLMoto Prototype. Master's thesis, Instituto Superior Técnico, 2019.

[18] W. W. Sons. *Power Electronics Converters*. Wiley & Sons Ltd, 2011. ISBN 978-1-84821-195-7.

[19] K. Zhou and D. Wang. Relationship between space-vector modulation and three-phase carrier-based PWM: a comprehensive analysis [three-phase inverters. *IEEE Transactions on Industrial Electronics*, 49(1):186–196, 2002. DOI 10.1109/41.982262.

[20] P. Varma and G. Narayanan. Space Vector PWM as a Modified Form of Sine-Triangle PWM for Simple Analog or Digital Implementation. *IETE Journal of Research*, 52(6):435–449, mar 2006. DOI 10.1080/03772063.2006.11416484.

[21] B. Tibério. VIENA Documentation Guide. [Online]. Available: https://github.com/VIENA-IST/VIENA, 2018. [Accessed: Apr. 19, 2022].

[22] R. Sebastião. Battery management system integration and performance in the VIENA eletric Vehicle. Master's thesis, Instituto Superior Técnico, 2022.

Appendix A

Code developed

In this appendix, the code developed can be seen: for reading and setting inverter parameters.

```python
import time
import canopen

#Create network
network = canopen.Network()
network.connect(channel='can0', bustype='socketcan')

#Add node to network
node = network.add_node(1,'viena_30_06_2022_2.dcf')

#Shows what nodes are connected in network
for node_id in network:
    print(network[node_id])

""" #Prints DCF file with all the indexes and subindexes
for obj in node.object_dictionary.values():
    print('0x%X: %s' % (obj.index, obj.name))
    if isinstance(obj, canopen.objectdictionary.Record):
        for subobj in obj.values():
            print('  %d: %s' % (subobj.subindex, subobj.name)) """

access = node.sdo['Password Entry']['Access Level Indication']
if access.raw !=4:
    print('User not logged in, logging in as Engineering Access Level')
    password = node.sdo['Password Entry']['Password (16-bit)']
    password.raw = 0x4BDF
    if access.raw == 4:
        print('User log in successfully')
node.tpdo.read()

preop_command = node.sdo['Force system to pre-operational state']
while True:
    inverter_state = input('Set inverter to: "OPERATIONAL:0" or "PRE-OPERATIONAL:1" ?\n')
    if inverter_state.lower() not in ('0', '1'):
        print('Not a valid answer')

    if inverter_state == '1':
        preop_command.raw= 1
        print('Setting inverter to PRE-OPERATIONAL')
```

Code developed - Part 1

```
        #Change Kp and Ki
        kp = node.sdo['AC Motor data (manufacturer specific)']['Current control proportional gain (Kp)']
        kp_set = input('Kp value: ')
        kp_change = 0.000030517578125
        kp_true = float(kp_set)/float(kp_change)
        kp.raw = kp_true
        ki = node.sdo['AC Motor data (manufacturer specific)']['Current control integral gain (Ki)']
        ki_set = input('Ki value: ')
        ki_change = 0.000030517578125
        ki_true = float(ki_set)/float(ki_change)
        ki.raw = ki_true
        node.tpdo[1].event_timer = 10
        node.tpdo[2].event_timer = 10
        node.tpdo[3].event_timer = 10
        node.tpdo[4].event_timer = 10
        node.tpdo[5].event_timer = 10
        #If map is valid
        node.tpdo[1].enabled = True
        node.tpdo[2].enabled = True
        node.tpdo[3].enabled = True
        node.tpdo[4].enabled = True
        node.tpdo[5].enabled = True
        #Save PDO config to node
        node.tpdo[1].save()
        node.tpdo[2].save()
        node.tpdo[3].save()
        node.tpdo[4].save()
        node.tpdo[5].save()
        network.sync.start(1)

    if inverter_state =='0':
        preop_command.raw = 0
        print('Setting inverter to OPERATIONAL')
        read_values = input('\nStart reading values? yes or no\n')
        if read_values ==  'yes' and 'y':
            break

        if read_values == 'no' and 'n':
            print('Exiting program')
```

Code developed - Part 2

```
            exit()
        break

fileobj = open('output.txt', 'w')
fileobj.write('Target Id\tTarget Iq\tId\tIq\tADrive State\tTorque demand\tActual Torque\tMax Torque\tTarget Torque\tThrottle Value\tThrottle Input Voltage\tMotor Temperature\tUd\tUq\tBattery Current\tBattery Voltage\tMax MotorSpeed\tVelocity\n')
try:
    aux_index = 0;
    while True:
        node.tpdo[1].wait_for_reception()
        node.tpdo[2].wait_for_reception()
        node.tpdo[3].wait_for_reception()
        node.tpdo[4].wait_for_reception()
        node.tpdo[5].wait_for_reception()

        #TPDO 1
        target_id = node.tpdo['Additional Motor Measurements.Target Id (If)'].phys*0.0625 #Bits:16
        target_iq = node.tpdo['Additional Motor Measurements.Target Iq (Ia)'].phys*0.0625 #Bits:16
        actual_id = node.tpdo['Additional Motor Measurements.Id (If)'].phys*0.0625 #Bits:16
        actual_iq = node.tpdo['Additional Motor Measurements.Iq (Ia)'].phys*0.0625 #Bits:16

        #TPDO 2
        drive_state = node.tpdo['Traction Application Status.Traction Drive State'].phys #Bits:16
        torque_demand = node.tpdo['AC Motor Debug Information.Torque demand value (U.T_d)'].phys*0.0625 #Bits:16
        torque_actual = node.tpdo['AC Motor Debug Information.Torque actual value (DWork.Td)'].phys*0.0625 #Bits:16
        torque_max = node.tpdo['AC Motor Debug Information.Maximum torque (DWork.Td_max)'].phys*0.0625 #Bits:16

        #TPDO 3
        target_torque = node.tpdo['Target torque'].phys*0.1 #Bits:16
        throttle_value = node.tpdo['Throttle Value'].phys*0.000030 #Bits:16
        throttle_input_voltage = node.tpdo['Throttle Input Voltage'].phys*0.0039 #Bits:16
        motor_temperature = node.tpdo['Additional Motor Measurements.Motor Temperature 1 (Measured - T1)'].phys #Bits:16

        #TPDO 4
        actual_ud = node.tpdo['Additional Motor Measurements.Ud (Uf)'].phys*0.0625 #Bits:8
        actual_uq = node.tpdo['Additional Motor Measurements.Uq (Ua)'].phys*0.0625 #Bits:8
        battery_current = node.tpdo['Device Measurements.Battery Current'].phys*0.0625 #Bits:16
```

Code developed - Part 3

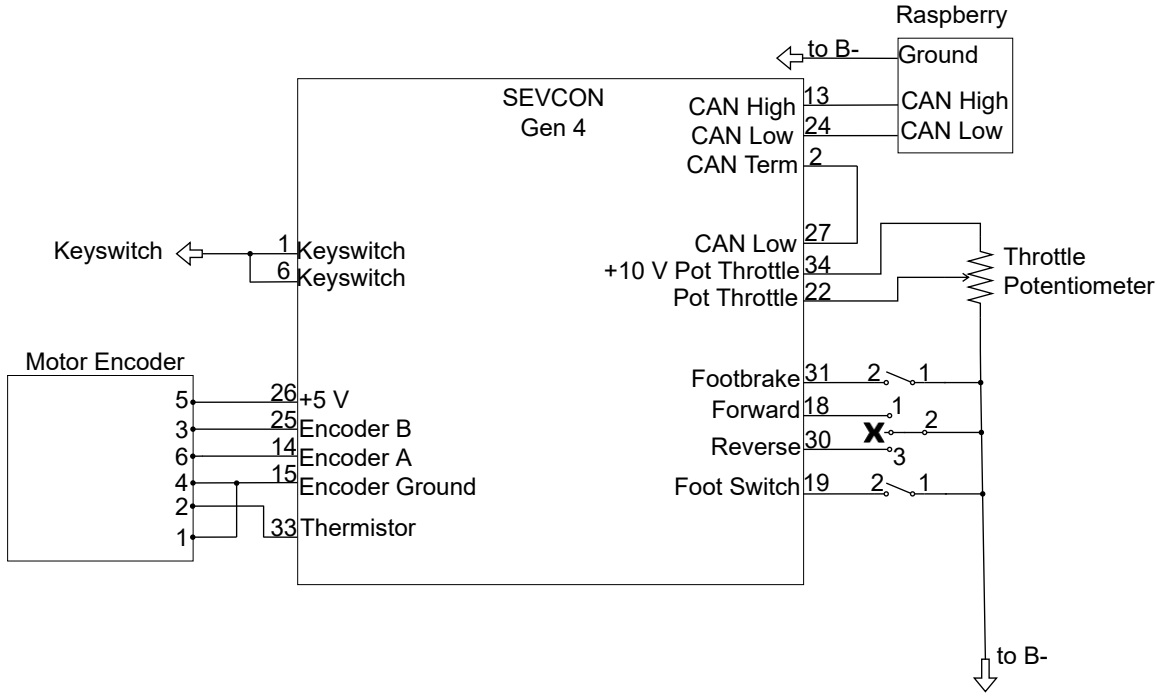
```
        battery_voltage = node.tpdo['Device Measurements.Battery Voltage'].phys*0.0625 #Bits

        #TPDO 5
        max_motor_speed = node.tpdo['Maximum motor speed'].phys #Bits:32
        velocity = node.tpdo['Velocity'].phys #Bits: 32

        fileobj.write('%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\n' % (target_id, target_iq, actual_id, actual_iq, drive_state, torque_demand, torque_actual, torque_max, target_torque, throttle_value,throttle_input_voltage,motor_temperature,actual_ud,actual_uq, battery_current, battery_voltage,max_motor_speed, velocity))

        aux_index = aux_index + 1
        if aux_index > 100:
            print(velocity)
            aux_index = 0


except KeyboardInterrupt:
    print('Program stopped writting to file')
    fileobj.close()
```

Code developed - Part 4

Appendix B

Connections diagram

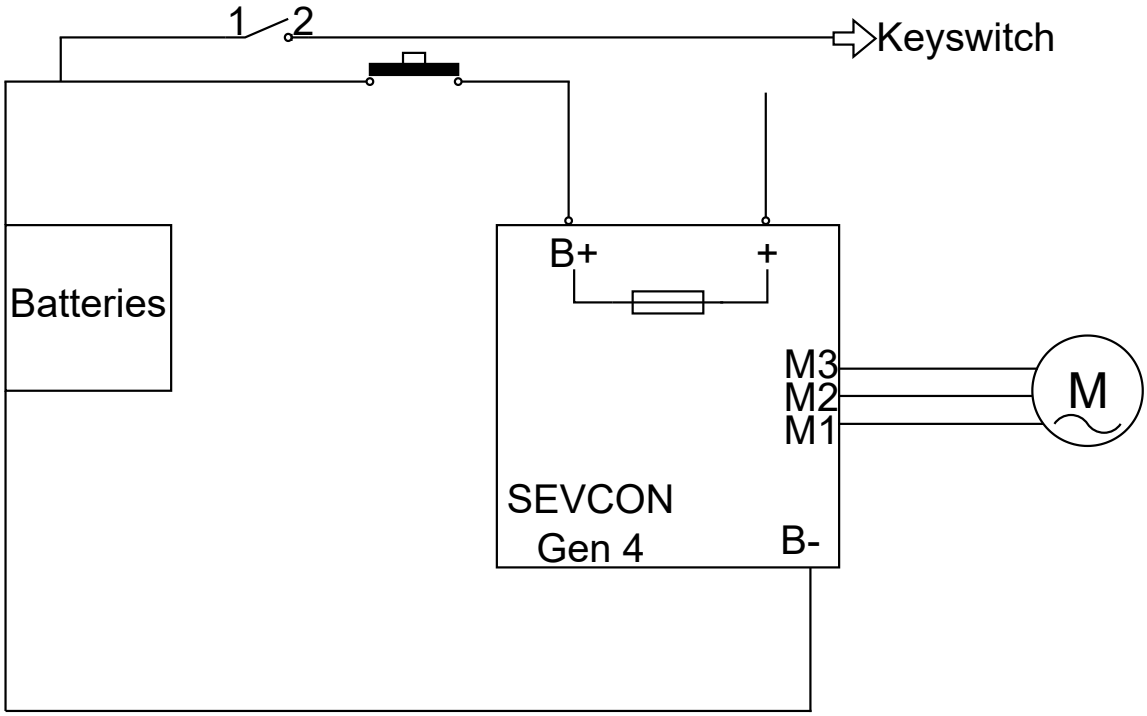
In this appendix, the connections diagram can be seen, along with the inverter pin numbers that correspond to the signals.



Appendix C

Connections diagram - Power

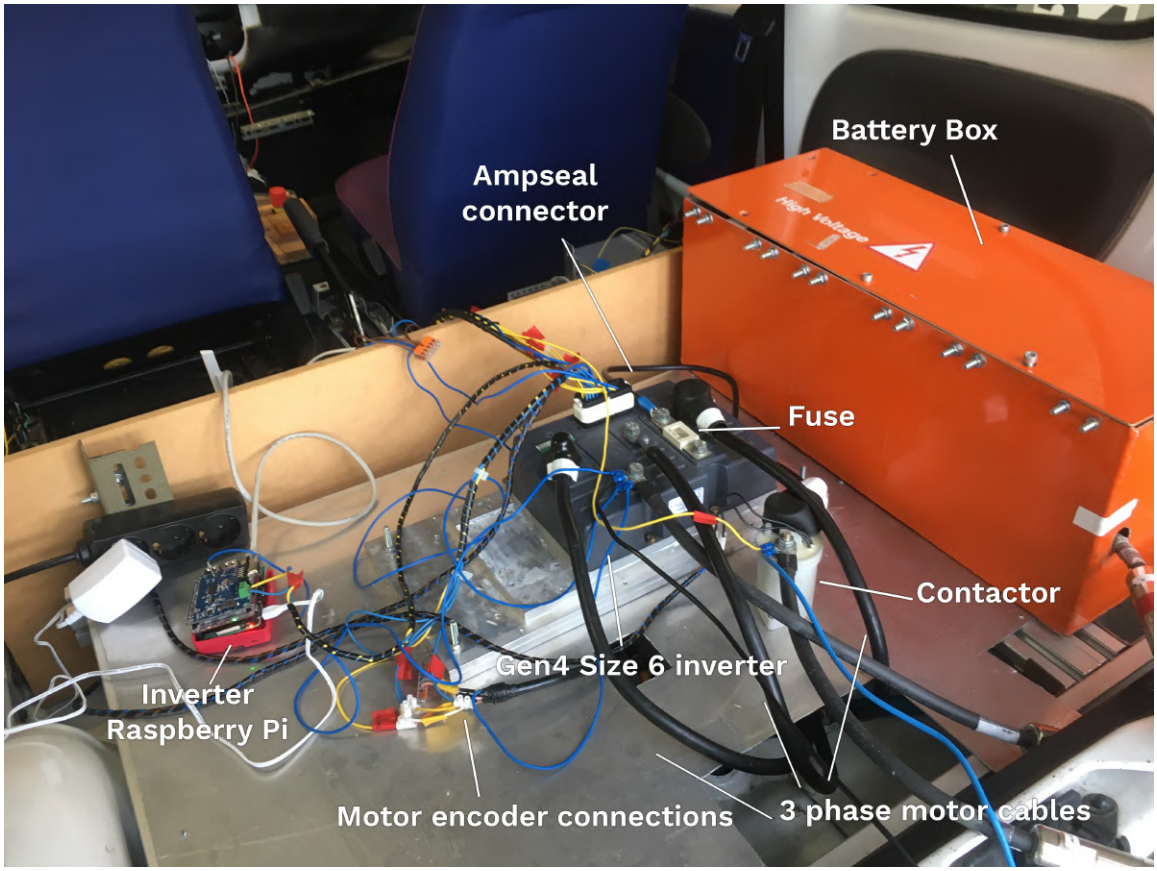
In this appendix, the connections diagram - Power can be seen.



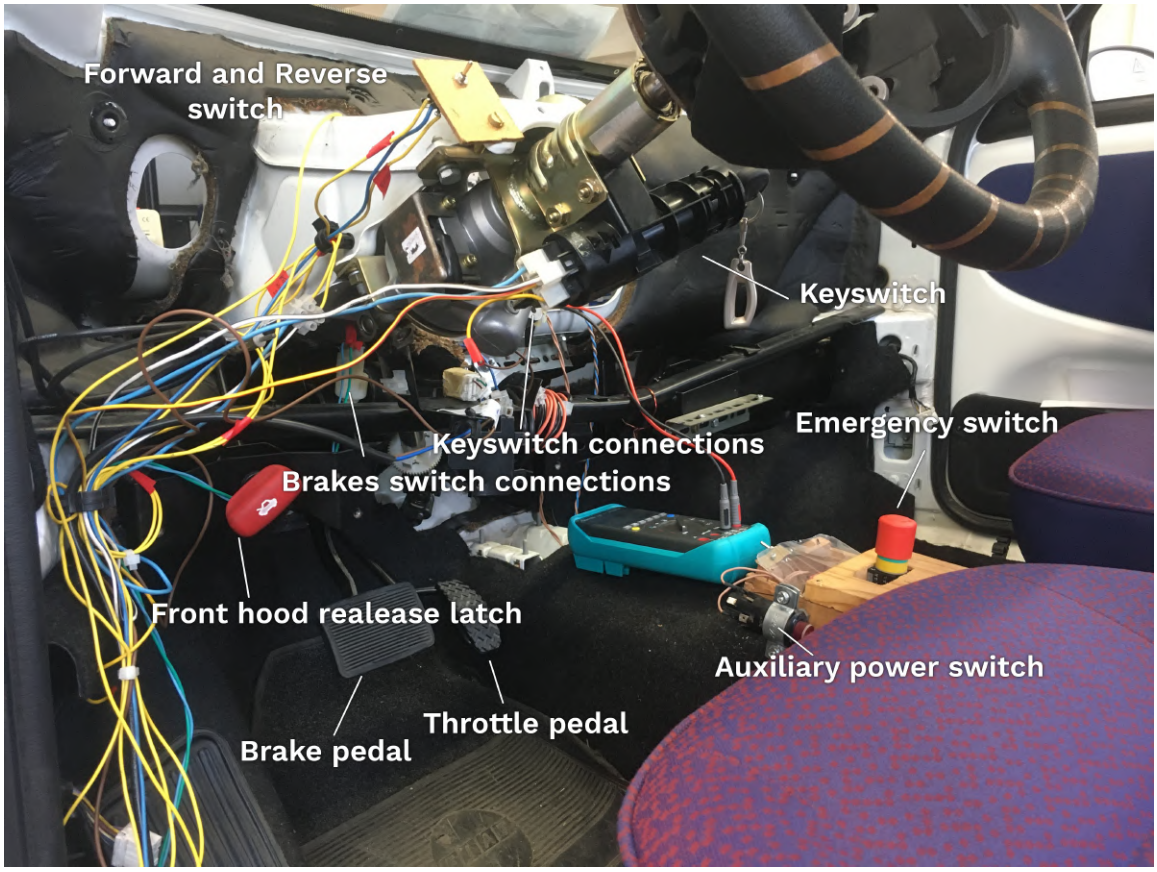
Appendix D

Pictures of the VIENA setup

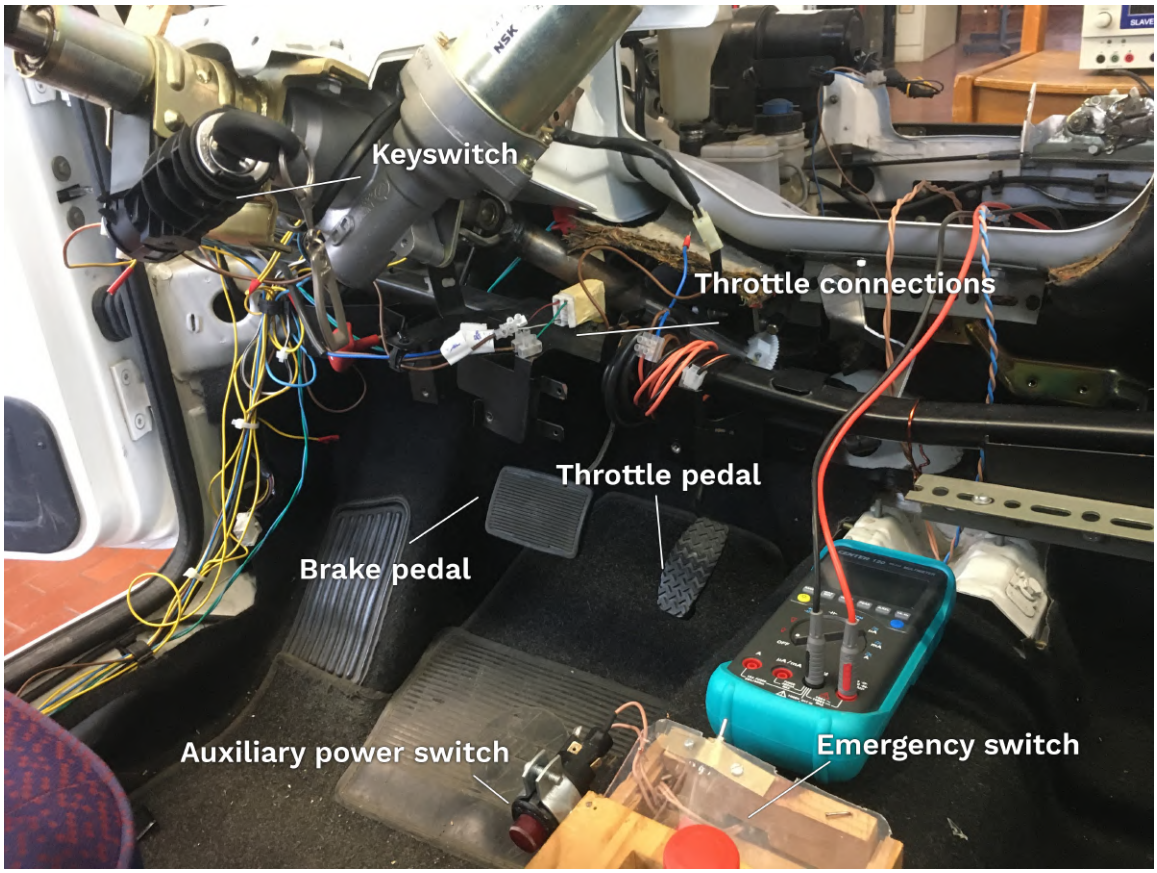
In this appendix, several pictures are shown of the setup used in the VIENA project.



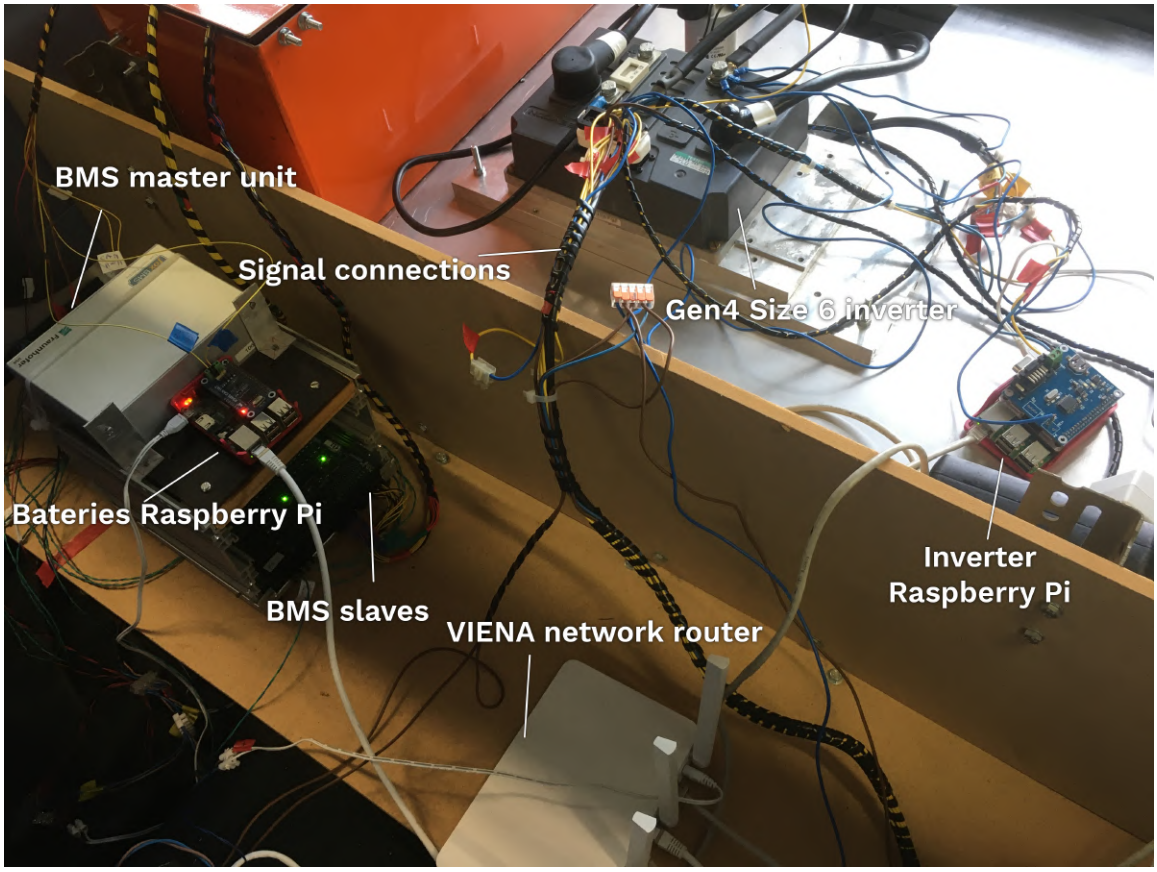
VIENA setup - Part 1



VIENA setup - Part 2



VIENA setup - Part 3

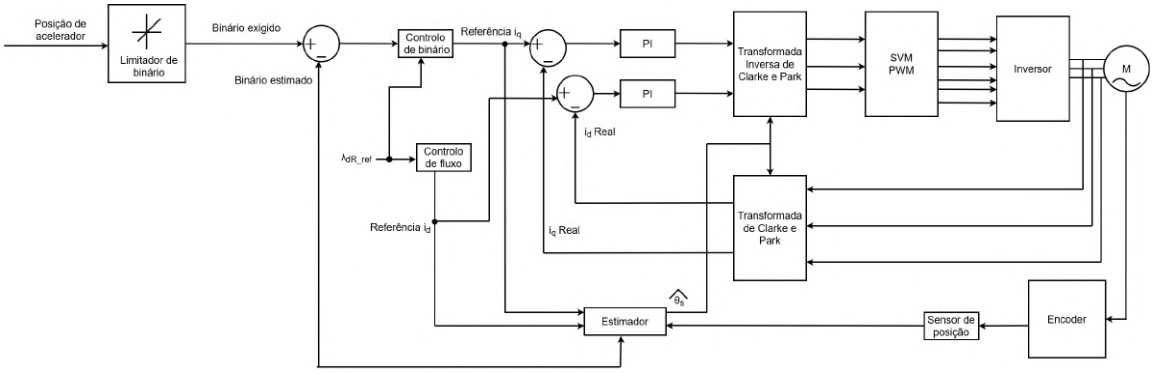


VIENA setup - Part 4

Appendix E

FOC Diagram

In this appendix, the FOC diagram used in the VIENA project is shown.



FOC Diagram used by VIENA.