Master of Science Thesis in Electrical Engineering
Department of Electrical Engineering, Linköping University, 2021

Simulation of a Current Controller with Dead-Time Compensation

Mattias Granström and Johanna Heide

Master of Science Thesis in Electrical Engineering

Simulation of a Current Controller with Dead-Time Compensation

Mattias Granström and Johanna Heide

LiTH-ISY-EX–21/5372–SE

Supervisor: **Filipe Barbosa**
ISY, Linköping University
Stig Moberg
ABB Robotics
Mikael Norrlöf
ABB Robotics
Erik Thenander
ABB Robotics

Examiner: **Svante Gunnarsson**
ISY, Linköping University

Division of Automatic Control
Department of Electrical Engineering
Linköping University
SE-581 83 Linköping, Sweden

Abstract

This master's thesis is divided into two parts. The first part concerns the development of a simulation model of a current controller and a physical drive unit, both implemented in SIMULINK with the use of legacy code and regulated with field oriented control. The second part concerns the development of a dead-time compensation algorithm. The dead-time is a small delay added to the pulse width modulation signal to diminish the risk of a short circuit in the power electronics. The dead-time causes a voltage distortion, resulting in distorted phase currents, a lower bandwidth and ultimately a decreased machine accuracy. The new simulation environment was able to simulate a real life scenario with promising results. Hence, it could be used to evaluate the dead-time compensation algorithms. Three different dead-time compensation algorithms were implemented and they all showed an increased smoothness of the phase currents as well as an increased controller bandwidth. Both these features are desirable outcomes and all three algorithms show potential to improve accuracy when implemented in a real system.

Acknowledgments

There are a lot of people involved in this project we are grateful to. This spring has been both challenging and exciting in a good way and we both feel that we have learned so much along the way. We would, however, not have been able to finish this master's thesis without all the help provided by our supervisors, examiner, experts at ABB and opponents.

First and foremost, we would like to thank our supervisors at ABB Robotics Erik Thenander, Stig Moberg and Mikael Norrlöf. Your expertise and commitment to this project never ceased and you were always ready to step in and help us past any obstacle we faced, even in these times of lockdowns and working from home. We would also like to thank our examiner Svante Gunnarsson and our supervisor Filipe Barbosa at ISY for their guidance and academic experience. You provided clear answers and good recommendations to all of the questions that arose during the spring. Lastly, we would like to thank our opponents Henrik Fallgren and Viktor Uvesten for making sure this thesis will make as meaningful a contribution to the field of electrical engineering as possible.

Linköping, June 2021
Mattias Granström and Johanna Heide

Contents

Notation

Nomenclature

Notation	Meaning
i	Instantaneous current
I	Constant or average current
F	Arbitrary function
$\mathcal{F}$	Magnetmotive force (mmf)
N	Number of turns for a coil
H	Magnetic field intensity
θ	Angle
p	Number of pole pairs
k	Factor (see index) or time step in discrete time
g	Gravitational constant or gate signal
f	Frequency or arbitrary function
L	Inductance
R	Resistance
u, v	Instantaneous voltage
U, V	Constant or average voltage
λ	Flux linkage
K	Constant
T	Torque or time interval
J	Moment of inertia
B	Viscous damping
t	Time
ω_e	Angular frequency
ω_b	Bandwidth

Nomenclature - Index

Notation	Meaning
a	Phase A
b	Phase B
c	Phase C
0	Zero-sequence component
rms	Root mean square value
$peak$	Peak value
d	Direct axis
q	Quadrature axis
α	α-axis
β	β-axis
ag	Air gap
r	Rotor
s	Stator or sample
C	Concentrated to one slot or collector
ph	Per phase
w	Winding factor index
e	Electrical, electromagnetic or back-emf constant index
t	Torque constant index
l	Load
GE	Gate-emitter
CE	Collector-emitter
RB	Reverse breakdown
FB	Forward breakdown
sat	Saturated
th	Threshold
l, L	Low
h, H	High
on	On-signal
off	Off-signal
d	Dead-time
m	Mid-point
$comp$	Compensated

Abbreviations

Abbreviation	Meaning
BJT	Bipolar junction transistor
FOC	Field-oriented control
GUI	Graphical user interface
IGBT	Insulated gate-bipolar transistor
MOSFET	Metal-oxide-semiconductor field-effect transistor
PM	Permanent magnet
PMSM	Permanent magnet synchronous motor
PWM	Pulse width modulation
RMF	Rotating magnetic field
RMS	Root mean square
SVM	Space vector modulation
VSI	Voltage source inverter

1 Introduction

This master's thesis was performed during the spring of 2021 at ABB Robotics in Västerås. ABB is one of the big leaders in manufacturing and developing world class industrial robots. The robots from ABB are used all over the world in a wide range of industries, for instance the car and food industries. The area of application for the robots include welding, material handling, assembly, picking and painting to name a few. Many of these tasks require high precision and might also be done at high speeds, which increases the demand for well engineered components. The robots are powered by electric machinery and controlled to maximize the performance of the electromechanical parts. The core of the thesis revolves around a simulation model of a current controller and the drive unit hardware connected to it. The first part of the thesis is about the process of building the simulation model. The second part of the thesis is focused on algorithm development, where different methods for dead-time compensation are developed and evaluated.

1.1 Motivation

Simulation models have come to play an important role in both analysis of system behaviour and development of new systems or algorithms over the years. Demands for faster and more accurate systems are ever increasing and tech companies across the globe always strive to improve their products. ABB Robotics is no exception. A solid motion control is paramount when developing industrial robots and a simulation model is a very powerful tool in the development process. Some of the advantages with an accurate simulation model is that it is both efficient and cheap to run simulations compared to running real tests since it requires no hardware. Some ideas can be ruled out with a simple simulation without the necessity to purchase additional parts only to arrive at a dead end.

With a simulation model it is possible to, for instance, swap between different motors and electrical components quickly and with no risk. It is also easy to log any signal of interest whereas measuring data from real systems is generally not an easy task.

The necessity for a good dead-time compensation stems from accuracy demands on the robots. Dead-time is a widely known phenomenon within the field of control theory. The dead-time of interest in this thesis is the one deliberately introduced in the inverter. It has been observed that an increased dead-time will cause the path following of a robot to deteriorate. Naturally, a good way to compensate for the introduced dead-time and reduce the negative effects on the performance of the robots is desirable.

1.2 Purpose

A simulation model has the potential to greatly improve workflow. Having a reliable model would enable ABB to test new algorithms and other ideas to improve the behaviour of the system. Having a simulation model for one module of the robot could also facilitate the process of creating simulation models for other modules as well. This is the purpose of the first part of the thesis.

The development of a dead-time compensation algorithm is also of great interest to ABB. The goal of a good dead-time compensation is of course to improve the performance of the robot. This performance improvement could for instance appear as a better path following or a larger bandwidth for the current controller. Having a good algorithm could also reduce the need for an aggressively tuned, high bandwidth controller, since the errors from the reference could be reduced. This is the purpose of the second part of the thesis.

1.3 Problem Description

As mentioned, the first part of the thesis is the development of a simulation model. This simulation model is to be implemented in SIMULINK. One of the components in this model is the current controller. It is desirable to make use of any legacy code available for the current controller. Therefore, the first obstacle is to find a way to integrate the legacy code written in C with SIMULINK. Apart from the current controller, there are hardware components to be modeled, such as the three-phase inverter and the actual motor. Thus, the first questions are:

- How can a simulation model of a discrete current controller and a continuous physical system be modeled to provide accurate results with a reasonable simulation time?
- What is the best way to make use of legacy code written in C in a SIMULINK model?

The second part of the thesis covers the algorithm development. The chosen phenomenon to try to compensate is dead-time. In short, the current controller uses pulse width modulation (PWM) in order to control the torque of the electric motor. A core concept of PWM is to create short impulses at high frequencies. In real systems it is necessary to add a small delay, a so called dead-time, to the rising edges of the PWM signals resulting in an output voltage that is slightly distorted from the reference voltage. This dead-time is inescapable to avoid a short circuit of the inverter that is subject to the PWM signal. A method to counteract this unwanted effect, so called dead-time compensation, is therefore useful to increase robot accuracy and an algorithm for this is to be developed in the second part of this master's thesis. Thus, the following questions are:

- How does the dead-time affect the behaviour of the system in terms of shape and behaviour of the phase currents and bandwidth of the current controller?
- What compensation methods seem useful to improve the behaviour of the system in terms of the shape and behaviour of the phase currents and the bandwidth of the current controller?

1.4 Delimitations

To be able to finish the thesis on time and still provide useful results some important delimitations have been set. The major delimitations are listed below:

- The focus of the controller part of the simulation environment is the current controller. Only a basic position and speed controller will be used.
- The simulation environment will only include one axis.
- No field-weakening will be considered in the simulation model.
- The legacy code can not be modified in any way.

1.5 Individual Contributions

Throughout this project, we have been working very closely together. No initial areas of responsibility were determined but as the thesis progressed some natural divisions emerged. We worked together on the current controller but as it was time to model the physical system Johanna focused on the inverter and the transistors while Mattias focused on the motor. During the algorithm development, Mattias was more responsible for Algorithms 1 and 2 since they were quite similar while Johanna's main responsibility was Algorithm 3. Other tasks, such as report writing, was divided equally among us.

2

Theory

In this chapter all the theory needed to understand the reasoning behind the chosen methods used in this thesis is presented. The system at hand consists of several different mechanical and electrical components and they all have different characteristics. This chapter therefore aims to describe these characteristics and any relevant physics involved in the components and processes.

2.1 Three-Phase Circuits

Almost all power electronics based on AC electric energy involve some form of polyphase system. A balanced n-phase system consists of n phases with equal impedance where the voltages of each phase are equal in magnitude and separated by a phase angle of $360°/n$. A very common polyphase system is the three-phase system which contains three voltages of equal magnitude that are separated by a phase angle of $120°$. Although there are several ways to achieve a three-phase system, a common method is to connect the three phases symmetrically in the same circuit resulting in a single associated three-phase system. [6]

2.1.1 Wye and Delta Connections

One option to achieve an associated three-phase system is to use a Wye connection, also known as a Y-connection or a star connection. In a Wye connection all three phases are connected in a single point called the neutral. The circuit diagram of the Wye connection is shown in Figure 2.1.

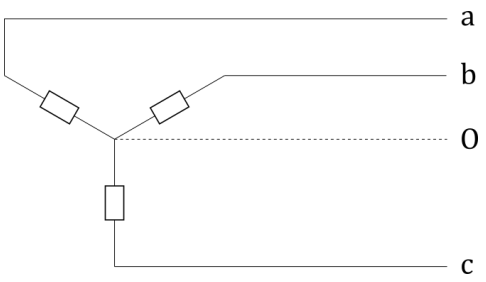

Figure 2.1: *A circuit diagram of a Wye connection.*

Having a neutral conductor is in this case optional. If it is left out, by Kirchoff's current law the sum of the three currents i_a, i_b and i_c in the neutral always amounts to zero, resulting in

$$i_a + i_b + i_c = 0\,. \tag{2.1}$$

Another option to achieve an associated three-phase circuit is to utilize a Delta connection, also known as a Δ-connection. In a Delta connection, the three phases are connected individually as a triangle. The circuit diagram of the Delta connection is shown in Figure 2.2. In this case no neutral exists.

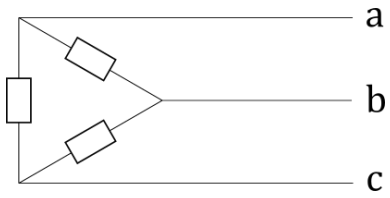

Figure 2.2: *A circuit diagram of a Delta connection.*

2.1.2 Peak and Root Mean Square Currents

Since the voltages and currents in an AC system vary periodically, it is important to distinguish between peak values, instantaneous values and mean values. The root mean square (RMS) value F_{rms} for a periodic function $f(t)$ over a period T is:

$$F_{rms} = \sqrt{\left(\frac{1}{T}\int_0^T f^2(t)\,dt\right)} \tag{2.2}$$

For a sine wave this is equal to $1/\sqrt{2}$ times the peak value. Thus, the ratio between the peak current I_{peak} and the RMS current I_{rms} is:

$$I_{rms} = \frac{I_{peak}}{\sqrt{2}} \tag{2.3}$$

2.2 Field-Oriented Control

The most common method used to control a permanent magnet synchronous motor (PMSM) is called field-oriented control (FOC). A simplified overview of a typical FOC structure is shown in Figure 2.3. The working principle is based on the transformation of the three-phase stator currents in the abc-frame to currents in the orthogonal dq-frame by using the Clarke and Park transforms. The d stands for direct current ant the q stands for quadrature current. The aim of this procedure is to control the currents i_d and i_q. The reference for i_q is obtained from the requested motor torque while i_d is usually controlled towards zero if no field-weakening is required. Field-weakening is briefly explained in Appendix A. The current feedback from the motor is compared to the references, creating error signals that can be used in two PI-controllers, one for i_d and one for i_q. The outputs of the PI-controllers are reference voltages in the dq-frame that via the inverse Clarke and Park transforms are converted to reference voltages in the abc-frame. The inverter uses the voltage references to control the voltages and consequently the currents of the three phases in the motor. Other components can also be implemented in the FOC in order to get a more accurate result, for example a feedforward component of the controller. Converting the reference voltages into a format that can be used by the inverter also requires for instance a method called space vector modulation. [9]

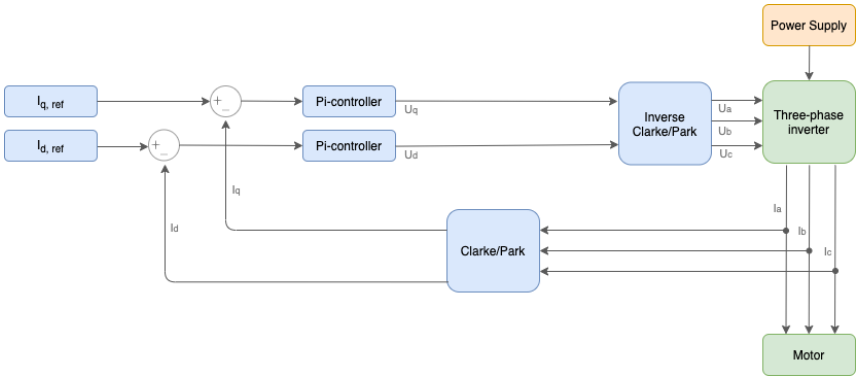

Figure 2.3: *A simplified block diagram of FOC.*

2.2.1 Clarke Transform

The Clarke transform is used to transform the three-phase currents from the fixed abc-frame in the motor into a fixed orthogonal reference frame called the $\alpha\beta$-frame,

$$\begin{bmatrix} i_\alpha \\ i_\beta \\ i_0 \end{bmatrix} = \frac{2}{3} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix}. \tag{2.4}$$

In Equation (2.4), the last component on the left hand side i_0 is known as the zero-sequence component. In a balanced three-phase system, explained in Section 2.1, this component should be equal to zero and the only components of interest are the currents i_α and i_β. Since the third current in a balanced three-phase system is determined by the other two, the Clarke transform for a balanced three-phase system can be written as

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{\sqrt{3}} & \frac{2}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} i_a \\ i_b \end{bmatrix}. \tag{2.5}$$

A visual representation of the Clarke transform is shown in Figure 2.4.

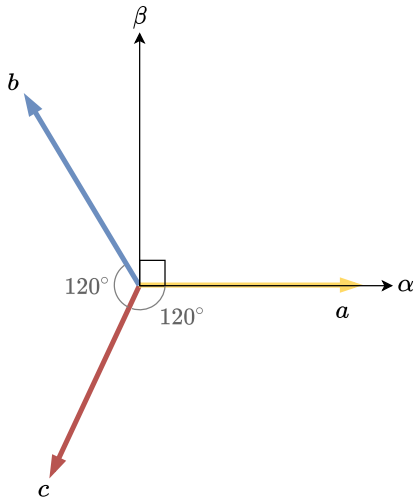

Figure 2.4: *A visual representation of the Clarke transform from the abc-frame to the $\alpha\beta$-frame.*

2.2.2 Park Transform

The Park transform is applied in order to obtain currents in the rotating dq-frame from the fixed $\alpha\beta$-frame. In order to do this it is necessary to know the rotation angle θ. The transform is then expressed as

$$\begin{bmatrix} i_d \\ i_q \\ i_0 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \\ i_0 \end{bmatrix}. \tag{2.6}$$

Once again, the i_0 component can be disregarded in a balanced system. In that case the transform simplifies to

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix}. \tag{2.7}$$

A visual representation of the Park transform is shown in Figure 2.5. In Equations (2.4) - (2.7), the subjects of transformation are the motor currents but both the Clarke and the Park transform can be applied to voltages as well. To go back from the dq-frame to the abc-frame the inverse Park and Clarke transforms are used, which means multiplying with the inverses of the matrices instead.

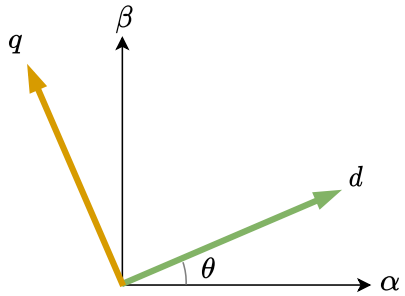

Figure 2.5: *A visual representation of the Park transform from the $\alpha\beta$-frame to the dq-frame.*

2.3 Permanent Magnet Synchronous Motors

Just like the vast majority of electrical machinery, the operation of a PMSM is based on the interconversion of electrical and mechanical energy made possible by shaping and directing magnetic fields. The PMSM is a polyphase AC device and this thesis is focused on the three-phase scenario. The theory in this section (Section 2.3) was sourced from [3], [6] and [9] that together provided a solid foundation for the understanding of PMSMs.

2.3.1 Components of the PMSM

The two main parts of a PMSM are the rotor and the stator. Both the rotor and the stator are made of high-permeability materials, typically iron or alloys of iron. A cross section of a typical PMSM is shown in Figure 2.6. The rotor is mounted on an axle and one or more permanent magnets (PMs) are mounted on or inside the rotor core. The stator encloses the rotor and it contains a number of slots. These slots serve the purpose to make room for the stator windings. The stator windings are windings of some conductor, typically copper, which carry the three-phase currents in order to produce a rotating magnetic field (RMF). Since the PMs on the rotor also give rise to a magnetic field, the rotor can be pulled along by the stator RMF. It is called a synchronous machine since the rotation of the rotor is synchronized with the rotation of the RMF and thus proportional to the electric frequency.

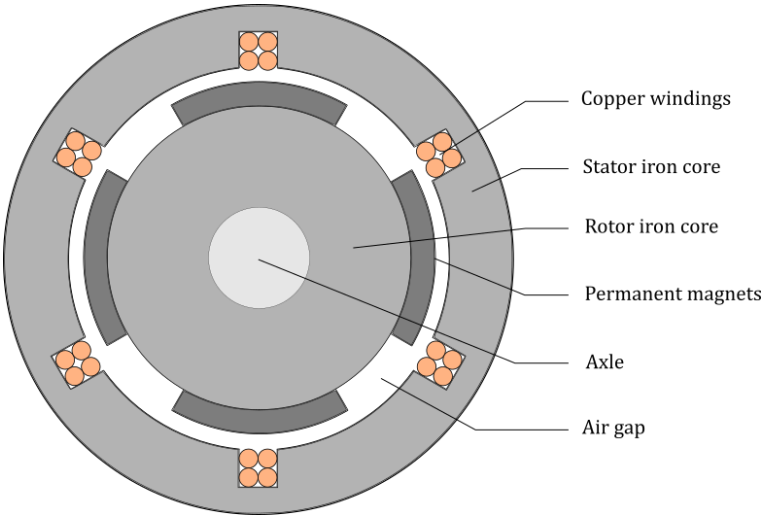

Figure 2.6: *Cross section of a four-pole surface mounted PMSM.*

Permanent Magnets and Magnetic Materials

Unlike an electromagnet, a PM produces a constant magnetic field without the need of electric power. PMs are made of ferromagnetic materials which consist of microscopic domains in which the magnetic moments of all the atoms are parallel, resulting in a net magnetic moment. The magnetic field created by a PM typically gets weaker with increasing temperature and age.

The permeability of a magnetic material is not actually constant. In unmagnetized materials the above mentioned domains are oriented in random directions and the net magnetic flux of the material is very low. However, if the material is exposed to a sufficiently strong external magnetic field, the orientation of the magnetic moment in these domains will start to line up with the direction of the external magnetic field. The effects of the magnetic moments of the domains are added up resulting in a higher magnetic flux density than what would be possible with the external magnetic field alone. Therefore, the effective permeability of the material increases until all the domains are fully aligned with the external field. At this point the material is saturated. When the external field is removed, the orientation of the magnetic moments in the domains will tend to return to their initial state, what is known as their axes of easy magnetization. However, the domains will still have some component of the direction of their magnetic moment in the direction of the external magnetic field. This results in a both nonlinear and multi-valued relationship between the magnetic flux density and the magnetic field intensity known as magnetic hysteresis. Typically, this behaviour cannot be described analytically but can be shown as a hysteresis loop in a graph. For many applications, it is sufficient to only use a single-valued curve between the end points of the hysteresis loop.

Another type of loss occuring in the stator core is caused by the induced electromotive force (emf) occuring during machine operation. This emf causes eddy currents in the stator iron and the eddy current losses are proportional to the square of the product of frequency and flux density.

Stator Windings

It is the currents in the stator windings that give rise to the RMF in the motor. The two main winding types for PMSMs are concentrated windings and distributed windings. In a concentrated winding all the conductors of a coil in a certain phase are placed in the same stator slot. Figure 2.7 shows a simple example of a concentrated winding with only one phase.

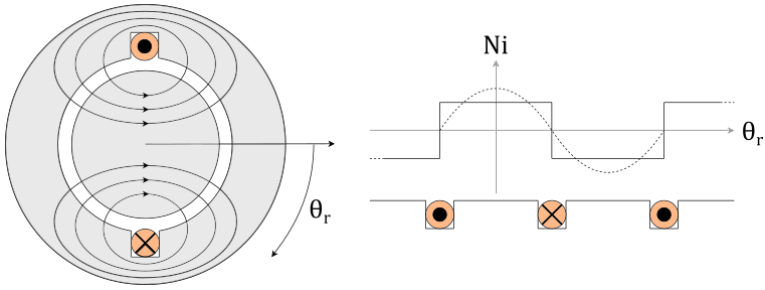

Figure 2.7: *Cross-section of a motor with a single-coil concentrated winding and the resulting mmf.*

The magnetomotive force (mmf) $\mathcal{F}$ produced by a coil with N turns carrying a current i is

$$\mathcal{F} = Ni\,. \tag{2.8}$$

For a symmetrical structure with a uniform air gap, the magnetic field intensity in the air gap H_{ag} at angle θ_r is equal in magnitude but opposite in direction to that at angle $(\theta_r + \pi)$. Assuming all the reluctance in the circuit is focused in the air gap and the flux crosses the air gap twice, the resulting mmf drop across an air gap becomes $Ni/2$. If the slot openings were infinitely narrow the mmf would switch between $-Ni/2$ and $Ni/2$ upon crossing a coil winding. In reality the mmf produced by a coil consists of a fundamental space-harmonic component and a number of higher order harmonic components. By proper distributions of the coil windings, however, the effects of the higher order harmonic components can be reduced. For the machine in Figure 2.7, the fundamental air gap mmf component $\mathcal{F}_{ag1}$ can be described as

$$\mathcal{F}_{ag1} = \frac{4}{\pi}\left(\frac{Ni}{2}\right)\cos(\theta_r)\,. \tag{2.9}$$

In a distributed winding the conductors of a coil in a certain phase are placed in a number of adjacent slots. A simple example of a distributed winding with only one phase is shown in Figure 2.8.

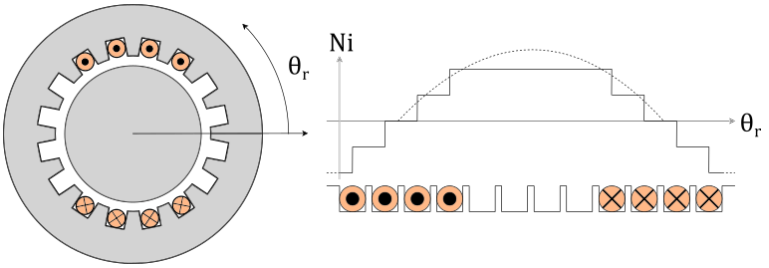

Figure 2.8: *Cross-section of a motor with a single-phase distributed winding and the resulting mmf.*

Again, assuming infinitely narrow slots and negligible iron reluctance, the mmf wave will be a series of steps of size $2N_c i$, where N_c is the number of coil turns in the slot. As can be seen by comparing Figures 2.7 and 2.8, the distributed winding produces a closer approximation to a sinusoidal distributed mmf than the concentrated winding. It is often a reasonable assumption to make that the mmf wave in a PMSM is sinusoidal. Since the magnetic axes of the separate coils in a distributed winding are not aligned with the resultant magnetic axis and taking multiple poles into account, Equation (2.9) is modified into

$$\mathcal{F}_{ag1} = \frac{4}{\pi}\left(\frac{2k_w N_{ph}}{p}\right) i\cos(p\theta_r), \tag{2.10}$$

where p is the number of pole pairs, N_{ph} is the number of turns in series per phase and k_w is a winding factor. The winding factor is typically between 0.85 and 0.95 and it is required because of the different orientations of the magnetic axes in a distributed winding.

Since the stator contains slots for the windings, its geometry is not completely smooth. An effect of this is that when a rotor magnet passes a stator slot the magnetic field around it will change slightly. The result is an oscillating torque during operation called the cogging torque which will affect the output torque of the machine.

Saliency

The rotor of a PMSM can be either salient or non-salient (also known as cylindrical) which refers to if the poles of the rotor are protruding or not. The most important effect of saliency on the machine is the uniformity of the air gap between rotor and stator. In a machine with a uniform air gap of length g it is a reasonable assumption that the magnetic field intensity in the air gap H_{ag} is di-

rected radially and has a constant magnitude across the air gap. It can thus be calculated as

$$H_{ag} = \frac{\mathcal{F}_{ag}}{g}, \tag{2.11}$$

where $\mathcal{F}_{ag}$ is the mmf in the air gap. Evidently, the magnetic field intensity and the mmf are related by a factor $1/g$.

For non-uniform air gaps the distribution of the magnetic field is significantly more complex. In many cases, however, it is possible to make use of the theory for uniform air gaps with some small alterations, such as calculating an effective air gap based on machine dimensions.

Rotor Configuration

A fundamental property of a PMSM is its number of poles which refers to the amount of magnetic poles on the rotor. The poles always appear in pairs and the number of pole pairs will affect the ratio between the electrical angle and the rotor angle. Denoting the electrical angle θ_e, the rotor angle θ_r and the number of pole pairs p, the relation can be expressed as

$$\theta_e = p\theta_r . \tag{2.12}$$

The same logic applies to the angular frequencies, yielding

$$\omega_e = 2\pi f_e = p\omega_r , \tag{2.13}$$

where ω_e is the electrical angular frequency, ω_r is the rotor angular frequency and f_e is the phase current electrical frequency. When analyzing PMSMs with more than one pole pair a useful simplification is to focus on a single pole pair, knowing that all the other pole pairs exist under identical mechanical, electrical and magnetic conditions.

There are several possible configurations for the PMs on the rotor in order to create magnetic fields. Some examples of different configurations are surface mounted PMs, surface inset PMs and interior PMs. These configurations are shown in Figure 2.9. The different configurations have different properties, and one of the main differences is the inductance of the direct and quadrature axes. For a single pole pair motor the direct axis is parallel to the polar axis of the rotor magnet and it is centered in the middle of its north pole. The quadrature axis, on the other hand, is parallel to the inter-polar axis which is oriented 90 electrical degrees ahead of the direct axis. These axes are commonly referred to as the d-axis and the q-axis respectively.

The stator inductance is called direct axis inductance, L_d, when the direct axis is aligned with the stator winding and quadrature axis inductance, L_q, when the quadrature axis is aligned with the sator winding. Both inductances are highly

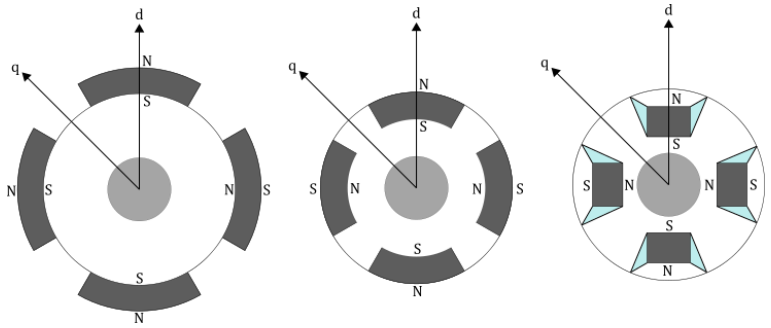

Figure 2.9: *Different configurations of the PMs on a two pole-pair rotor. From left to right: surface mounted, surface inset and interior.*

affected by the thickness of the magnets, the length of the air gap and the placement of the magnets. For the surface inset PMSM and the interior PMSM there is a significant difference between L_d and L_q. In the case of the surface mounted PMSM, if the air gap and PM reluctances are dominant and since the permeability of air is approximately equal to that of a high-grade PM, it is a viable approximation to consider the direct and quadrature axis inductances as equal,

$$L_d = L_q \,. \tag{2.14}$$

2.3.2 Dynamic Model of a PMSM

In this section a dynamic model of a Wye connected, surface mounted PMSM is derived. As with any model, some assumptions and approximations have to be made and those are:

- Stator windings are balanced with a sinusoidal distributed mmf.
- The back-emf is sinusoidal.
- Saturation and parameter changes due to temperature and ageing of the components are neglected.
- Iron losses are neglected.
- There is no leakage flux.

Assuming a stationary reference frame with axes α and β according to Figure 2.10, the electrical equations are

$$v_\alpha = R_\alpha i_\alpha + \frac{d\lambda_\alpha}{dt} \,, \tag{2.15}$$

$$v_\beta = R_\beta i_\beta + \frac{d\lambda_\beta}{dt} \,. \tag{2.16}$$

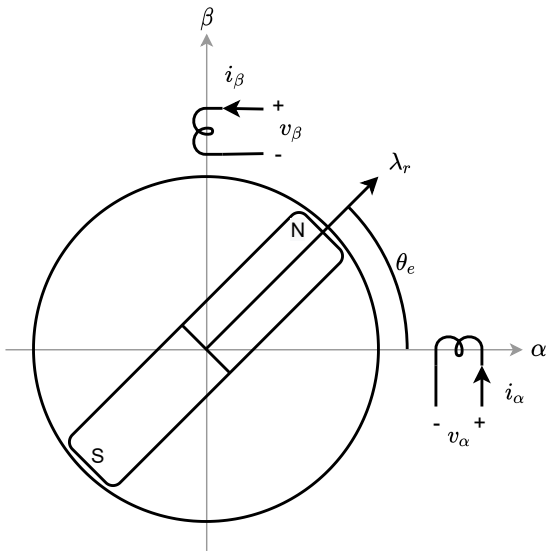

Figure 2.10: *A stationary reference frame for a two-pole PMSM.*

Here v_α and v_β are the voltages, i_α and i_β are the currents, R_α and R_β are the resistances and λ_α and λ_β are the flux linkages in the fixed α- and β-axis stator windings. The flux linkages can be expressed as

$$\lambda_\alpha = L_{\alpha\alpha} i_\alpha + L_{\alpha\beta} i_\beta + \lambda_r \cos(\theta_e), \tag{2.17}$$

$$\lambda_\beta = L_{\beta\alpha} i_\alpha + L_{\beta\beta} i_\beta + \lambda_r \sin(\theta_e), \tag{2.18}$$

where λ_r is the armature flux linkage due to the rotor magnets, θ_e is the instantaneous electrical rotor position measured from the stator α-axis, $L_{\alpha\alpha}$ and $L_{\beta\beta}$ are the self inductances of the α- and β-axes windings and $L_{\alpha\beta} = L_{\beta\alpha}$ are the mutual inductances. Only regarding the case with a cylindrical rotor and balanced windings results in

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = R_s \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + L_s \begin{bmatrix} \frac{di_\alpha}{dt} \\ \frac{di_\beta}{dt} \end{bmatrix} + \lambda_r \omega_e \begin{bmatrix} -\sin(\theta_e) \\ \cos(\theta_e) \end{bmatrix}, \tag{2.19}$$

where R_s is the stator resistance, L_s is the stator inductance and ω_e is the time derivative of θ_e. The Park transform from Equation (2.7) is used to transform the equations from the fixed stator reference frame to the rotating rotor reference frame, yielding

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = \begin{bmatrix} R_s & -\omega_e L_s \\ \omega_e L_s & R_s \end{bmatrix} \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \begin{bmatrix} L_s & 0 \\ 0 & L_s \end{bmatrix} \begin{bmatrix} \frac{di_d}{dt} \\ \frac{di_q}{dt} \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_e \lambda_r \end{bmatrix}. \tag{2.20}$$

The last term $\omega_e \lambda_r$ comes from the motional emf of the rotor flux. Two useful motor constants are the back-emf constant K_e and the torque constant K_t. These

constants describe some fundamental ratios in the motor and are related to the rotor flux by

$$K_e = \frac{K_t}{\sqrt{3}} = p\lambda_r \,. \tag{2.21}$$

The electromagnetic torque T_e produced in the motor is derived as

$$T_e = \frac{3}{2} p[\lambda_r + (L_d - L_q) i_d] i_q \,. \tag{2.22}$$

Equation (2.22) together with Equation (2.14) and (2.21) results in

$$T_e = \frac{\sqrt{3}}{2} K_t i_q \,. \tag{2.23}$$

Neglecting the cogging torque, the electromechanical equation of the motor is given by

$$J\frac{d\omega_r}{dt} = T_e - T_l - B\omega_r \,, \tag{2.24}$$

where ω_r is the mechanical rotor speed, J is the moment of inertia of the load and machine combined, B is the viscous damping of the load and the machine and T_l is the load torque. Combining Equations (2.13), (2.20) and (2.24), the final model can be expressed in state-space form as

$$\frac{di_d}{dt} = \frac{1}{L_s} v_d - \frac{R_s}{L_s} i_d + p i_q \omega_r \,, \tag{2.25}$$

$$\frac{di_q}{dt} = \frac{1}{L_s} v_q - \frac{R_s}{L_s} i_q - p i_d \omega_r - \frac{K_e}{L_s} \omega_r \,, \tag{2.26}$$

$$\frac{d\omega_r}{dt} = \frac{\sqrt{3}}{2} \frac{K_t}{J} i_q - \frac{T_l}{J} - \frac{B}{J} \omega_r \,, \tag{2.27}$$

$$\frac{d\theta_r}{dt} = \omega_r \,. \tag{2.28}$$

2.4 The Three-Phase Inverter

An inverter is used in order to transform DC current into AC current. In an inverter the current is controlled in such a fast manner that regular mechanical switches are not fast enough. Instead, semiconducting power devices are used. There are multiple different types of inverters and the focus in this thesis lies on a three-phase voltage source inverter (VSI). In this report a three-phase inverter that uses insulated-gate bipolar transistors (IGBTs) is presented. A Wye connected three-phase inverter is shown in Figure 2.11. It has three so called legs with two transistors in each leg, and each of these legs correspond to a phase. The phases are separately controlled with pulse width modulation (PWM) by quickly turning the transistors on and off.

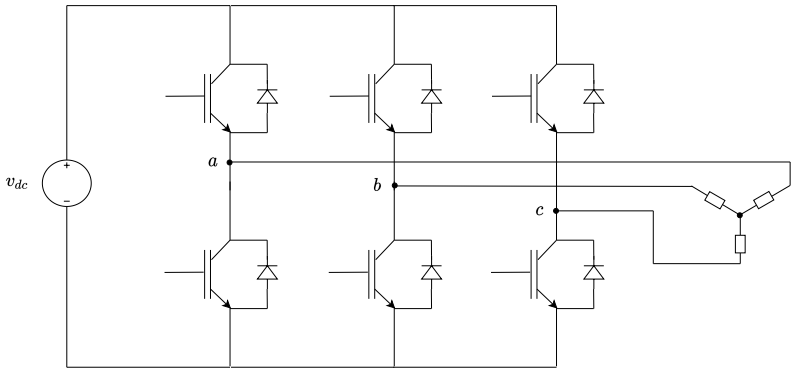

Figure 2.11: *A simplified electrical diagram of a Wye connected three-phase inverter.*

2.4.1 The IGBT

The IGBT is a combination of a bipolar junction transistor (BJT) and a MOSFET. It allows for fast switching and less conduction and switching losses. In Figure 2.12, the symbol for an IGBT is shown. When current flows between collector and emitter the IGBT is in its on-state. For this to happen, a sufficient amount of gate-emitter voltage v_{GE} is required. The collector current is called i_C. In Figure 2.13, a current-voltage $(I - V)$ diagram is shown for an arbitrary IGBT. In this figure it is clear that the more gate voltage is applied, the higher collector current i_C is achieved. Also, if the voltage exceeds the forward breakdown voltage denoted as $(v_{CE})_{FB}$ or the reverse breakdown voltage denoted as $(v_{CE})_{RB}$, the IGBT fails.

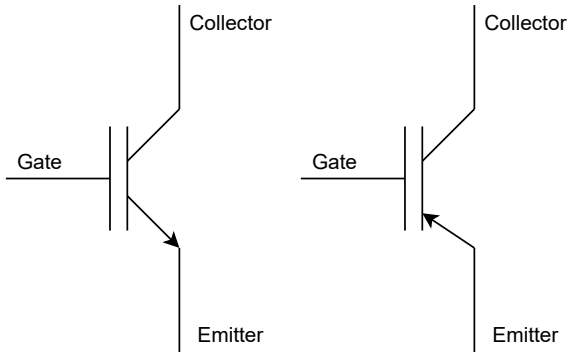

Figure 2.12: *The symbol used for IGBTs. N-channel based on the left and P-channel based on the right.*

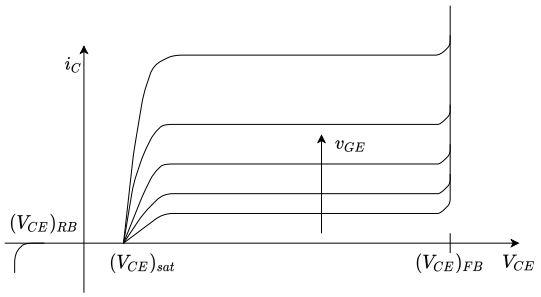

Figure 2.13: *An $I - V$ diagram for an IGBT.*

The IGBT as an Ideal Switch

The voltage drop across the IGBT can be made quite small with a sufficiently large gate signal. Thus, the device could be modeled as a circuit between the collector and the emitter. Since the IGBT carries unidirectional current when closed, it can be modeled as an ideal switch. Additionally, an anti-parallel diode is connected between collector and emitter in order to protect the transistor from high reverse voltages. Figure 2.14 shows an example of this modeling setup. A proof for the reasoning above is shown in Appendix B. [6]

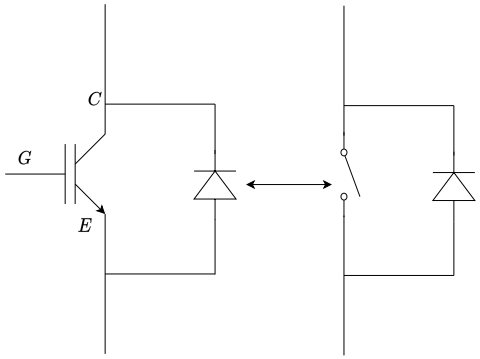

Figure 2.14: *The symbol for the IGBT together with an anti-parallel diode and a presentation of the IGBT working as an ideal switch.*

2.4.2 Pulse Width Modulation

The most common method to produce sinusoidal signals from an inverter is through PWM. This technique uses voltage pulses of different duration in order to achieve an approximated sinusoidal output over time. An example of this is seen in Figure 2.15 where the voltage pulses are obtained by rapidly switching the IGBTs on and off. The result is the generation of an approximately sinusoidal output. A higher switching frequency results in a smoother output.

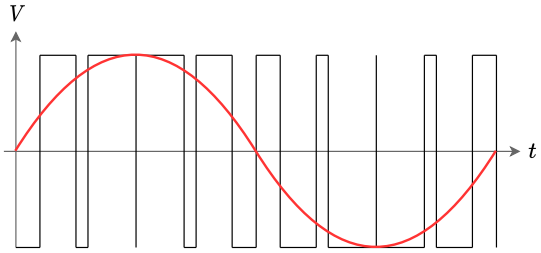

Figure 2.15: *The working principle of PWM. PWM signal shown in black and approximated sinusoidal voltage output in red.*

Space Vector Modulation

A common method to control the switching procedure when using three-phase inverters is space vector modulation (SVM). A three-phase inverter has three legs and two transistors in each leg that can be either on or off. In total there are eight different possible states the inverter can exist in depending on which transistors are on and off. These states are shown in Table 2.1, where S_p is the set of switches for each leg, $p = a, b, c$. The logical value 1 means that the higher switch is on while the lower switch is off and the value 0 represents the opposite case where the lower switch is on and the higher switch is off. Given the inverter in Figure 2.11 where the voltage at the low and high DC voltage source terminals are 0 V and v_{dc} respectively and assuming there is no voltage drop in the IGBTs, the mid-point phase voltage v_p between the transistors in each leg is determined by its state. The line voltages v_{ab}, v_{bc} and v_{ca} are equal to the differences in voltage between each phase. Both the phase voltages and the line voltages for each inverter state are also tabulated in Table 2.1

States	S_a	S_b	S_c	v_a	v_b	v_c	v_{ab}	v_{bc}	v_{ca}
I	1	0	0	v_{dc}	0	0	$+v_{dc}$	0	$-v_{dc}$
II	1	0	1	v_{dc}	0	v_{dc}	$+v_{dc}$	$-v_{dc}$	0
III	0	0	1	0	0	v_{dc}	0	$-v_{dc}$	$+v_{dc}$
IV	0	1	1	0	v_{dc}	v_{dc}	$-v_{dc}$	0	$+v_{dc}$
V	0	1	0	0	v_{dc}	0	$-v_{dc}$	$+v_{dc}$	0
VI	1	1	0	v_{dc}	v_{dc}	0	0	$+v_{dc}$	$-v_{dc}$
VII	0	0	0	0	0	0	0	0	0
VIII	1	1	1	v_{dc}	v_{dc}	v_{dc}	0	0	0

Table 2.1: *Simplified states of the inverter and the corresponding phase and line voltages.*

As the stator windings of each phase are distributed in space, each of these states will produce a certain space voltage vector. Two of the vectors (VII and VIII) are zero-vectors but the other six have specific directions. The six possible non-zero space voltage vectors are shown in Figure 2.16.

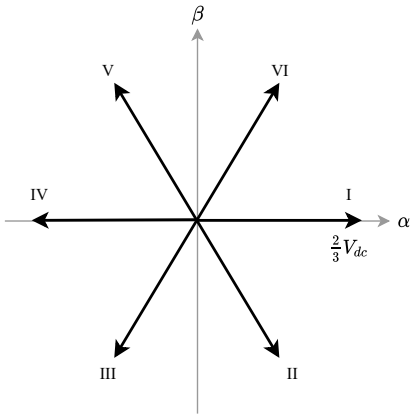

Figure 2.16: *Output space voltage vectors from the inverter for each non-zero switching state.*

It is possible to approximately create any space voltage vector that lies between any two of the six base vectors. This is done by quickly switching between the two base vectors closest to the wanted space vector during a switching period so that the average space voltage vector during that period is the reference space vector.

2.5 Dead-Time

As explained in [9], the three-phase VSI consists of three legs, each with two transistors in series. Figure 2.17 shows a close up of a single leg of an inverter.

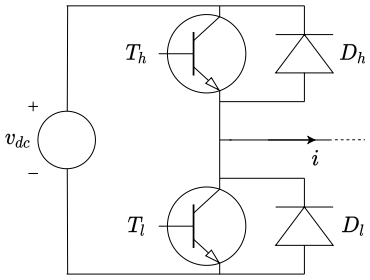

Figure 2.17: *Close up of a single inverter leg with notation.*

One of the main problems encountered using VSIs is the risk of having both transistors in a leg turned on, and thus conducting, resulting in a short circuit of the DC supply. This is known as a shoot-through fault. The two transistors in a leg are denoted as T_h and T_l for high and low. In an ideal VSI, T_h turns off at the exact same time T_l turns on and vice versa. However, in a real system T_h will require some additional time equal to the sum of storage time, current fall off and voltage rise times to completely turn off. Similar dynamics apply when turning the transistor on. To avoid inverter leg shoot-through as an effect of this, a time delay is introduced between the switching off of one transistor and the switching on of the other transistor in the same leg. This is done for all three legs in the inverter. The delay should at least be large enough to cover the time described above and usually a safety margin is added as well. This delay is known as the *dead-time* and the effect of the dead-time on the PWM signal is visualized in Figure 2.18.

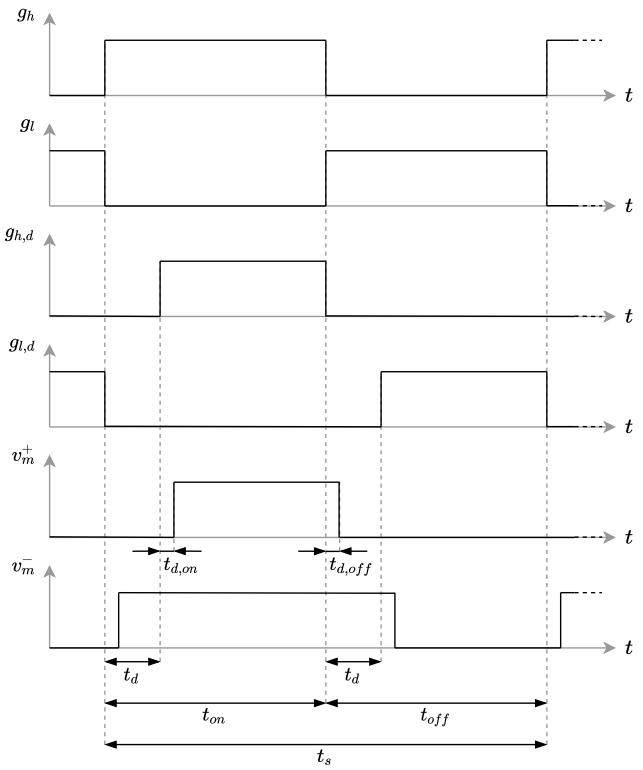

Figure 2.18: *Visualization of the effect of a dead-time delay on the gate signals and the output voltage.*

In Figure 2.18, the top two graphs show the ideal gate signals g_h and g_l for T_h and T_l respectively. T_h is on for a duration of t_{on} seconds and off for a duration of t_{off} seconds. The opposite is true for T_l. In the middle two graphs, a dead-time

t_d has been introduced to the gate signals resulting in the modified signals $g_{h,d}$ and $g_{l,d}$. The introduction of dead-time affects the mid-point voltage v_m between the transistors, referred to as the output voltage, since the ideal PWM signal has been altered. The distortion of the output voltage depends on the direction of the current. The reason for this is that the current direction determines which of the diodes D_h or D_l the current will flow through during the dead-time and thus what terminal of the DC supply the phase is connected to. For the case described in Figure 2.18, assuming the current is positive in the direction defined in Figure 2.17, the mid-point voltage will follow the curve of v_m^+ for a positive current and v_m^- for a negative current. The voltage signals are also slightly distorted from the modified gate signals due to the turn-on and turn-off delays $t_{d,on}$ and $t_{d,off}$ caused by non-ideal IGBTs. The effect on the magnitude of the output voltage is mainly dependent on the length of the dead-time. Since the dead-time is constant, the relative output voltage loss is higher for low voltage commands than for high voltage commands. The resulting phase currents are also distorted, primarily during zero-crossings of the currents, as an effect of the dead-time. The distorted voltages and currents have been found to give rise to issues in the control and performance of the motor and in many industrial applications it is desirable to compensate for these effects, a concept known as dead-time compensation.

2.5.1 Dead-Time Compensation

The impact of dead-time is a well known phenomenon and there is a wide variety of suggestions available on how to compensate for it. In this section the principle of three different compensation algorithms are presented.

Algorithm 1: Fixed Feedforward Compensation

Since the voltage distortion from the dead-time depends on the direction of the current, many compensation methods revolve around receiving reliable information about the current directions in the motor. As noted in for instance [2], [4], [11] and [14], measuring the motor currents and voltages is troublesome since there is a lot of high frequency noise as a result of the PWM. As the current crosses zero the noise will result in non-reliable data concerning the direction of the current. Both [4] and [11] specifically mention that there are two main ways to mitigate this problem. The first option is to use a low-pass filter on the measured motor currents or voltages. This will reduce the noise but will also introduce an unwanted phase delay on the signal. The other option is to skip the use of measurements altogether and instead estimate the current directions based on the reference currents or voltages. An issue with this method is that the reference voltages naturally are different than the actual output voltages from the inverter. The direction of the currents can be determined by using a sign function,

$$sgn(i_p) = \begin{cases} 1 & \text{for } i_p > 0 \\ -1 & \text{for } i_p < 0 \end{cases}. \tag{2.29}$$

Here, $p = a, b, c$ for the three phase-currents. The same principle holds for determining the sign of a voltage. Once the current directions have been decided the PWM signal can be altered to compensate for the dead-time. A suggestion made in [11] is to modify the space voltage vector whereas [1] and [13] suggest simply altering the pulse width just before it is sent to the inverter. Denoting the reference on-time of the PWM pulse of phase p as $t^*_{p,on}$, the real on-time $t_{p,on}$ becomes

$$t_{p,on} = t^*_{p,on} - sgn(i_p) * t_{err} + t_{p,comp}\,, \tag{2.30}$$

where $t_{p,comp}$ is the addition to the pulse length from the dead-time compensation and t_{err} can be derived as

$$t_{err} = t_d + t_{d,on} - t_{d,off}\,. \tag{2.31}$$

Here, t_d is the dead-time of the system and $t_{d,on}$ and $t_{d,off}$ are the turn-on and turn-off delays of the IGBTs. In a balanced three-phase system one of the phase currents always have a different sign than the other two. As explained in [13] two phases with the same current sign have no dead time influence but the single phase with a different sign must be compensated. The compensation term is therefore added to the phase with the odd current.

It is also possible to compensate the reference voltages directly before they are converted to switching times, as explained in [11]. By denoting the reference space voltage vector $\mathbf{U}^*$ the equivalent inverter output voltage vector during a switching interval of t_s seconds becomes $\mathbf{U}^* t_s$. Introducing a dead time t_d will give rise to another output voltage vector $\mathbf{U_d}$ during the dead time and the new output $\mathbf{U_{out}} t_s$ of the inverter during a switching interval will be

$$\mathbf{U_{out}} t_s = \mathbf{U_d} t_d + \mathbf{U}^*(t_s - t_d)\,. \tag{2.32}$$

The dead-time voltage vector $\mathbf{U_d}$ is dependent on the direction of the motor currents and can take one of the six forms displayed in Figure 2.19.

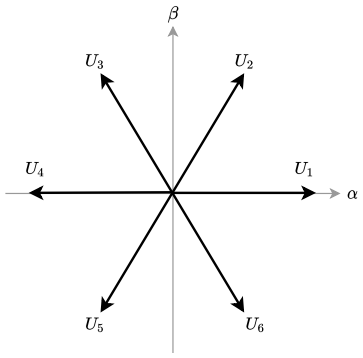

Figure 2.19: *The six possible space vectors as a result of the dead-time.*

The dead-time voltage vector can be determined by utilizing Equation (2.29) and the following equations

$$t_1 = -sgn(i_a)\,, \tag{2.33}$$

$$t_2 = \frac{sgn(i_c) - sgn(i_b)}{2}\,, \tag{2.34}$$

$$\Delta = \frac{3}{2}(1 - t_1) + t_1 t_2 + 1\,. \tag{2.35}$$

This will result in a value of Δ between 0 and 5 and the dead-time voltage vector can be determined as $\mathbf{U_d} = \mathbf{U_\Delta}$. In this case $\Delta = 0$ refers to the space vector $\mathbf{V_6}$.

Algorithm 2: Variable Feedforward Compensation

A slightly more recent approach that has been mentioned in [2] and [5], among other articles, is to find a more detailed relation between the motor currents and the compensation voltages. The compensation voltage refers to the amount of voltage that is desirable to add to or subtract from the reference voltage to compensate the dead-time effects. Instead of always compensating the voltages or pulse lengths with a fixed value, which could result in noisy compensation voltages for low currents, the idea is to also take the current magnitude into account, in addition to the current direction. A similar idea is presented in [10] where a look-up table is used instead of a function between motor current and compensation voltage. Denoting the compensation voltage for phase p as $U_{p,comp}$ and the motor current as i_p, the function could be

$$U_{p,comp} = \begin{cases} k & \text{for } i_p > i_{th} \\ \frac{k}{i_{th}} i_p & \text{for } -i_{th} < i_p < i_{th} \\ -k & \text{for } i_p < -i_{th} \end{cases}. \tag{2.36}$$

Here, k is a predetermined compensation value for when the magnitude of the current is above a certain threshold value i_{th}. The graph of the function in Equation (2.36) is shown in Figure 2.20.

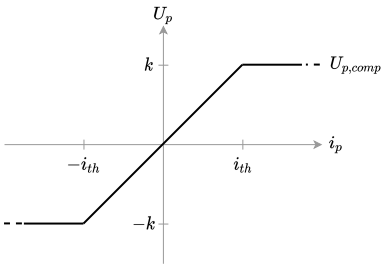

Figure 2.20: *A simple function that takes current magnitude into account at low currents.*

This is a big simplification compared to the functions suggested in [2] and [5] but the principle is the same.

Algorithm 3: Voltage Disturbance Observer

Problems when measuring parameters often occur when using an offline dead-time compensation method. This algorithm is based on the methods presented in [7] and [8], and it uses an online observer to calculate the distortion of voltage. The disturbance voltages are added to the voltage references in the current controller in order to compensate for the mismatch between reference voltage and output voltage due to the dead-time. The observer is based on time-delay control and since the approach is implemented in the current controller the implementation is discrete. The following equation represents the discrete-time model of the algorithm,

$$\mathbf{U}_{comp}(k) = R_s\mathbf{i}(k) + L_s\frac{\mathbf{i}(k+1) - \mathbf{i}(k)}{t_s} + \mathbf{E}(k) + \mathbf{f}(k)\,. \tag{2.37}$$

The bold letters represent the following vectors,

$$\mathbf{i} = \begin{bmatrix} i_d \\ i_q \end{bmatrix},$$

$$\mathbf{U}_{comp} = \begin{bmatrix} U_{d,comp} \\ U_{q,comp} \end{bmatrix},$$

$$\mathbf{E} = \begin{bmatrix} -L_s i_d \omega_r \\ L_s i_q \omega_r + \lambda_r \omega_r \end{bmatrix}.$$

The sampling time is t_s and $\mathbf{f}$ is the disturbance voltage vector defined as

$$\mathbf{f} = \begin{bmatrix} f_d \\ f_q \end{bmatrix}.$$

Furthermore, it is assumed that the variations of the disturbance voltages during a sampling period are nearly zero, resulting in

$$\mathbf{f}(k) \equiv \mathbf{f}(k-1)\,. \tag{2.38}$$

The approximation in Equation (2.38) results in the following estimation of disturbance voltages using the data from the previous time step,

$$\hat{\mathbf{f}}(k) \equiv \mathbf{f}(k-1) = \mathbf{U}_{comp}(k-1) - \left(R_s\mathbf{i}(k-1) + L_s\frac{\mathbf{i}(k) - \mathbf{i}(k-1)}{t_s} + \mathbf{E}(k-1)\right). \tag{2.39}$$

The estimated disturbance voltage can then be added to the reference voltages from the current controller. The measured currents in Equation (2.39) may include some high-frequency noise. Therefore, a discrete low-pass filter can be

added to filter the $\hat{\mathbf{f}}(k)$ signal. After a Z-transform the suggested filter is a first-order low-pass filter defined by

$$G(z) = \frac{at_s(1 + z^{-1})}{(2 + at_s) - (2 - at_s)z^{-1}}\,. \tag{2.40}$$

Here, a should be set as the cut-off frequency[1] of the filter and t_s is the given sample time for the system. In this algorithm no extra hardware is added to the system.

[1] Frequencies above the cut-off frequency are filtered out by the low-pass filter.

3
Method

In this section the methodology for the thesis work is presented. It provides an explanation of why certain methods are used and in what way specific approaches are utilized.

3.1 Simulation Model

A good simulation environment allows for a more efficient testing and development process which can reduce both developing time and cost. In modeling there is almost always a trade-off between model complexity and accuracy. The goal is to create a simulation model that is complex enough to capture the relevant dynamics but still user friendly and has a reasonable simulation time.

3.1.1 Current Controller

Both the real system and the simulation model are based on FOC. This control strategy is explained in Section 2.2. For confidentiality reasons the exact implementation of the current controller will not be presented in the report but it follows the FOC principle.

Usage of C-Code

In order to maintain the exact same execution manner when running a simulation as when operating a robot in the lab, it was determined to use the already existing C-code in the model of the current controller. The different parts of the FOC are implemented by using the SIMULINK block named C Caller. As long as the paths to the source and header files are specified in the *Model configuration parameters* in SIMULINK, the C Caller block can import functions from the

C-code. The C Caller block also allows for choosing inputs and outputs from the block. Input arguments that are pointers in the C-code can be chosen to operate as either inputs, outputs or both. All components in the current controller are implemented with this approach.

Another challenge when importing C-code into SIMULINK is the handling of custom data types. Below, an example is shown which explains the import of different data types and files in MATLAB. The header files and source files with the relevant code are also specified in this MATLAB-command. Here `DATATYPE_X`, `DATATYPE_Y` and `DATATYPE_Z` are examples of the custom data types.

```
1 Simulink.importExternalCtypes('Headerfiles', {'filename1.h', ...
      'filename2.h'}, ...
2 'Sourcefiles', {'filename1.c', 'filename2.c'}, ...
3 'Names', {'DATATYPE_X', 'DATATYPE_Y', 'DATATYPE_Z'})
```

The objects from the C-code are converted to the data types *Bus* or *AliasType* in MATLAB. The bus data type is very useful and easy to handle in SIMULINK.

3.1.2 Physical System

The physical system in this thesis consists of a three-phase inverter and a PMSM. SIMULINK includes a powerful toolbox called SIMSCAPE that serves the purpose to model physical systems of different kinds. SIMSCAPE has a big library of electrical components that will be used to model the physical parts of the system.

Inverter

The method used to implement the inverter is based on starting from a simplified model and gradually making it more complex. A first step is to use simple switches instead of transistors in SIMULINK. If this gives a satisfactory result and is able to capture the relevant dynamics it might be enough. If not, the process continues with more advanced components until a good enough result is obtained. The model of the inverter is initially supplied with a constant DC voltage source. This is also a simplification as the voltage in a real system might vary with time but it is a simple model to start with that can be developed if necessary. In Appendix B, a motivation as to why IGBTs can be modeled as ideal switching devices is presented. This principle is based on the lack of deviation in the collector-emitter voltage v_{CE} when applying a higher gate-emitter voltage v_{GE}. This is considered in the implementation as well.

PMSM

The implementation of the PMSM model follows the same principle as the implementation of the inverter. A good starting point is a purely electrical model of the motor that can simply be implemented with transfer functions. Once the first model shows promising results the process can continue with more advanced

components. Trade-offs considering simulation time, motor parameter implementation, wanted results and ease of use have to be considered. The documentation from the MathWorks website [12] is usually extensive and there are several pre-existing PMSM components that can be made to follow the dynamic PMSM model from Section 2.3.2.

3.1.3 Complete System

Connecting the current controller to the physical system will result in the complete system. The complete system should be able to run in a closed loop with both the current controller and the physical system and it is the final result of the first part of the thesis. One important decision when setting up large simulation environments with many different parts is the selection of a solver. The solver will affect both the simulation time and accuracy of the simulation. In a stiff model that contains both fast and slow dynamics as well as discrete and continuous parts, the choice of solver takes some consideration. An appropriate solver is selected based on the frequency of the current controller and the time constants in the physical system. The goal is to capture all the dynamics of interest in the physical model while maintaining a reasonable simulation time.

3.2 Dead-Time Compensation

The implementations of all dead-time compensation algorithms are based on the theoretical background given in Section 2.5.

3.2.1 Algorithm 1 - Fixed Feedforward Compensation

Two methods are used to compensate the dead-time effects using this principle. The first method compensates on the pulse length while the second method compensates on the reference voltage.

Algorithm 1A - Compensate on Pulse Length

As mentioned previously, there are some known issues regarding accurately determining the direction of the motor currents. One workaround which does not rely on feedback is to use the reference voltages to estimate the current directions in the motor. Very similar to Equation (2.29), the sign function can be written for the reference voltage u_p^* of phase p as well. Since it is only desirable to compensate on the phase with the odd current a modified sign equation $sgn_0(u_a^*)$ for phase A can be written as

$$sgn_0(u_a^*) = \begin{cases} 1 & \text{for } u_a^* > 0,\ sgn(u_b^*) = sgn(u_c^*) \\ 0 & \text{for } sgn(u_b^*) \neq sgn(u_c^*) \\ -1 & \text{for } u_a^* < 0,\ sgn(u_b^*) = sgn(u_c^*) \end{cases}. \tag{3.1}$$

Equations following the same principle can be written for phases B and C as well. In order to achieve $t_{p,on} = t^*_{p,on}$ in Equation (2.30) the compensation term $t_{p,comp}$ should be

$$t_{p,comp} = sgn(i_p) * (t_d + t_{d,on} - t_{d,off}) . \tag{3.2}$$

With the modified formula from Equation (3.1) and disregarding turn-on and turn-off delays in the IGBTs the final compensation term simply becomes

$$t_{p,comp} = sgn_0(u^*_p) * t_d . \tag{3.3}$$

Algorithm 1B - Compensate on Reference Voltage

Simply replacing the currents in Equation (2.29) with reference voltages results in

$$sgn(u^*_p) = \begin{cases} 1 & \text{for } u^*_p > 0 \\ -1 & \text{for } u^*_p < 0 \end{cases} . \tag{3.4}$$

Using Equation (3.4) together with Equations (2.33) - (2.35) results in the appropriate dead-time voltage vector $\mathbf{U_d}$. According to Equation (2.32) the difference between output voltage and reference voltage as an effect of the dead time is $(\mathbf{U_d} - \mathbf{U^*})t_d$. A compensation term $\mathbf{U_{comp}}$ is therefore added to the reference voltage which will result in the output

$$\mathbf{U_{out}} t_s = \mathbf{U^*}(t_s - t_d) + \mathbf{U_d} t_d + \mathbf{U_{comp}} . \tag{3.5}$$

To achieve $\mathbf{U_{out}} = \mathbf{U^*}$ in Equation (3.5) the compensation term is set to

$$\mathbf{U_{comp}} = -(\mathbf{U_d} - \mathbf{U^*})t_d = (\mathbf{U^*} - \mathbf{U_d})t_d . \tag{3.6}$$

3.2.2 Algorithm 2 - Variable Feedforward Compensation

As in Algorithm 1, the directions of the currents are estimated from the reference voltages instead of being measured. Equation (2.36) shows a voltage based compensation but the same principle applies to compensating the pulse length. A general compensation function $f_{p,comp}(u^*_p)$ based on the magnitude and sign of the reference voltages can be written

$$f_{p,comp}(u^*_p) = \begin{cases} 1 & \text{for } u^*_p > u_{th} \\ \frac{1}{u_{th}} u^*_p & \text{for } -u_{th} < u^*_p < u_{th} \\ -1 & \text{for } u^*_p < -u_{th} \end{cases} . \tag{3.7}$$

Here, u_{th} is the threshold voltage for the compensation function and it can be used as a tuning parameter. The value chosen in this method is 1% of the DC bus voltage. Using the same reasoning as in Algorithm 1A, the compensation term to add to the pulse length is

$$t_{p,comp} = f_{p,comp}(u^*_p) * t_d . \tag{3.8}$$

3.2.3 Algorithm 3 - Voltage Disturbance Observer

The theoretical background to this algorithm is presented in Section 2.5. This is an online algorithm based on compensating the voltage distortion due to dead-time by treating this discrepancy as a disturbance voltage. Furthermore, a time-delay controller is added based on the disturbance voltages. Equation (2.39) is implemented in SIMULINK and the new block diagram of the FOC is shown in Figure 3.1. The compensated voltages are

$$\begin{bmatrix} U_{d,comp} \\ U_{q,comp} \end{bmatrix} = \begin{bmatrix} \hat{f}_d + U_d \\ \hat{f}_q + U_q \end{bmatrix}, \tag{3.9}$$

where $\hat{\mathbf{f}}$ is found in Equation (2.39).

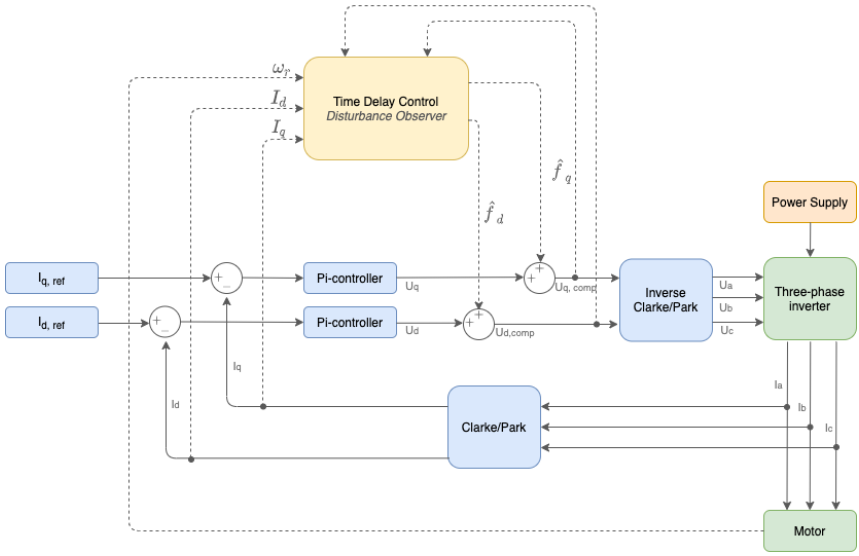

Figure 3.1: *A simplified block diagram of the new FOC with a time-delay observer*

3.3 Model Validation

Model validation is an important and usually time consuming part of the modeling work. The methods to validate the different parts of and ultimately the complete simulation model are presented in this section.

3.3.1 Current Controller

Each of the components implemented with the C Caller block can be verified by comparing the outputs of the block to measurements from the real current controller when using the same inputs. There are low-voltage motor cradles available

in the ABB office that can be used to collect measurement data. By logging the appropriate signals, i.e. the exact inputs and outputs of a certain block, and then using that data in the simulation model a proper verification can be carried out. For some simple components, like the Clarke and Park transforms, a verification can also simply be made by comparing the output of the SIMULINK block with the same matrix multiplication made in MATLAB. The whole current controller is also verified open-loop using the same principle as for the separate components.

3.3.2 Physical System

The physical system could be verified in a similar way as the current controller, but as this requires measuring physical quantities rather than logging digital signals some issues with noise and accuracy arises. This is especially troublesome when the currents are low and not all motor or inverter parameters are known. Another way of evaluating the physical model is by observing the behaviour of the motor to see if it behaves as expected. One example of a metric is the shape of the phase currents. The phases should have an offset of 120° to one another and be sinusoidal. There should also be a ripple in the current with the same frequency as the current controller. The mechanical parts of the motor should also behave in a physical way where the angle, speed and torque are dependent on each other. The predominant approach to verifying the motor is to close the loop and observe the whole system.

3.3.3 Complete System

By closing the loop the whole system can be verified. Different approaches can be used in order to verify its validity and stability. A good approach to the validation is to study the step response after a step in the torque reference is made. When the system is able to produce a satisfactory step response the test case can be made a bit more advanced. A good test case involves both an acceleration and a deceleration phase as well as a small period with constant velocity. This can be achieved by adding a position and speed controller to determine the torque reference. The output of the motor model in terms of rotor angle, rotor speed and torque can then be analysed and compared to the references. A good result here is a satisfactory reference following with little noise and smooth sinusoidal shaped phase currents.

3.3.4 Dead-Time Compensation Algorithms

When verifying dead-time compensation algorithms the behaviour of the entire system needs to be considered. This can be done in both the time domain and the frequency domain. For analysis in the time domain, phase currents from the motor can be observed. The addition of dead-time in a system typically causes plateaus in the phase currents when either one of the phases changes its polarity. This is a well-known issue that can be reduced with a compensation algorithm.

Secondly, a frequency analysis says a lot about the system. Using a chirp-signal[1] as input will generate a sinusoidal output as well. A suitable choice for the input signal is the quadrature current reference i_q^* since it is derived directly from the reference torque. The output is the actual quadrature current i_q obtained from the motor currents. These signals can be plotted in a Bode diagram to evaluate the frequency response. Different parameters can be set for the chirp-signal and for each algorithm three cases with different chirp-signal amplitudes are tested. The data for each test case can be seen in Table 3.1, where f_{start} and f_{end} are the starting and ending frequencies of the chirp-signal respectively, t_{sim} is the simulation time and T_{peak}^* is the amplitude of the torque reference chirp-signal resulting in the amplitude $i_{q,peak}^*$ of the quadrature current reference chirp-signal. The amplitude of the torque signal is selected to achieve what is considered low, medium and high motor currents in the three different test cases.

Parameters	Case 1	Case 2	Case 3
f_{start}	1 Hz	1 Hz	1 Hz
f_{end}	1000 Hz	1000 Hz	1000 Hz
t_{sim}	20 s	20 s	20 s
T_{peak}^*	0.5 Nm	1 Nm	5 Nm
Resulting $i_{q,peak}^*$	1.08 A	2.16 A	10.78 A

Table 3.1: *Data for the frequency analysis test cases.*

The motor and simulation parameters used for the evaluation of the dead-time compensation algorithms are presented in Table 3.2

Parameter	Value
Stator resistance, R_s	2.758 Ω
Stator inductance, L_s	9.751 mH
Pole pairs, p	5
Torque constant, K_t	0.656 Nm/A
Back-emf constant, K_e	0.379 Vs/rad
DC bus voltage, V_{dc}	600 V
Moment of inertia, J	0.01 kgm^2
Viscous damping, B	0.149e-3 Nms
Load torque, T_l	0 Nm
Dead-time, t_d	2 µs

Table 3.2: *Motor and simulation parameters for the test case.*

[1] A chirp-signal is a sinusoidal signal with an increasing frequency.

4

Results

In this chapter the results from both the development of the simulation model and the dead-time compensation algorithms are presented.

4.1 Simulation Model

The simulation model was implemented with the methods explained in Section 3.1. First of all, some results regarding the simulation time are presented. The time it takes to run a simulation varies with the selected resolution of the PWM-signal since this dictates the maximum time step. The time step needs to be at least 1 µs in order to capture the relevant dynamics. Simulating 1 s with a resolution of 1 µs takes around 46 s. This simulation time increases almost linearly with an increased resolution, meaning the same simulation with a resolution of 0.1 µs takes approximately 460 s.

4.1.1 Current Controller

In this section both the implementation results and the performance results for the current controller are presented.

Implementation Results

In addition to the components of the FOC, SVM is also used in the current controller model. The final output of the current controller is the switching times. These have to be processed in a switching logic block before they are sent to the inverter. An example of an implementation using the C Caller block is shown in Figure 4.1. As shown in the figure, the inputs and outputs of the block can be

modified in the GUI. All FOC components in the current controller were implemented with this approach in SIMULINK and connected to each other.

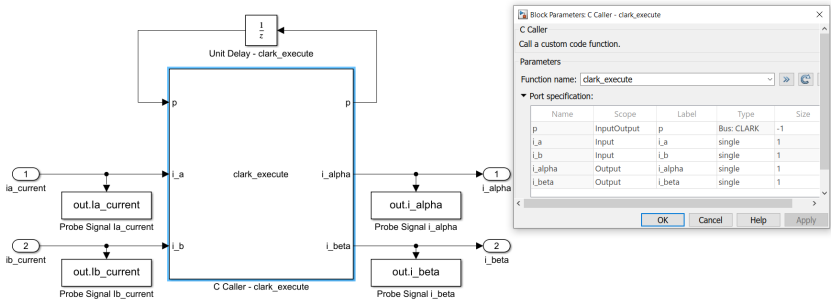

Figure 4.1: *The implementation of the Clarke transform component in SIMULINK using the C Caller block.*

Performance Results

As mentioned in Section 3.3.1, the verification of the current controller was mostly done with measurements from the motor cradle. The results from running the open-loop current controller with measured signals as inputs are shown in Figure 4.2. The simulated signals are almost identical to the measured values.

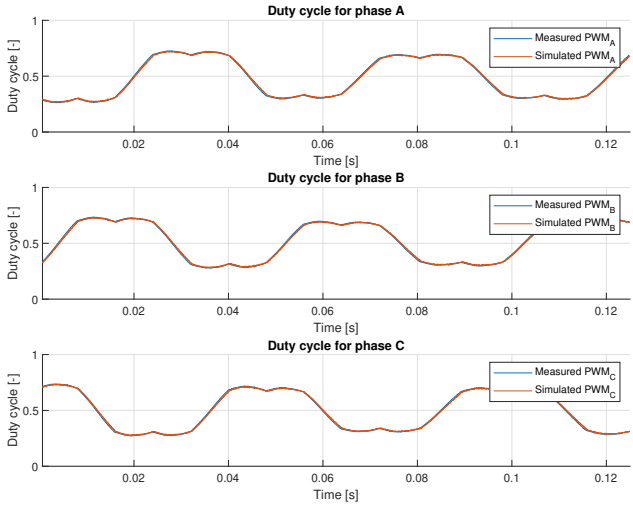

Figure 4.2: *Duty cycle for all phases when running the open-loop current controller.*

4.1.2 Physical System

The final version of the inverter model that had the best accuracy compared to the level of complexity can be seen to the left in Figure 4.3. The inputs A_L, A_H, B_L, B_H, C_L and C_H are logic signals obtained from the current controller and the consequent switching logic block. The dead-time is also implemented in the logic and can simply be set in an initiation m-file that is run before the simulation is started. The choice to model the IGBTs as ideal in their switching was made. The IGBTs and the voltage source used are imported from the SIMSCAPE library. For the motor model, the factors explained in Section 3.1.2 were considered. The model that had the best trade-off was an existing model in SIMSCAPE, seen to the right in Figure 4.3.

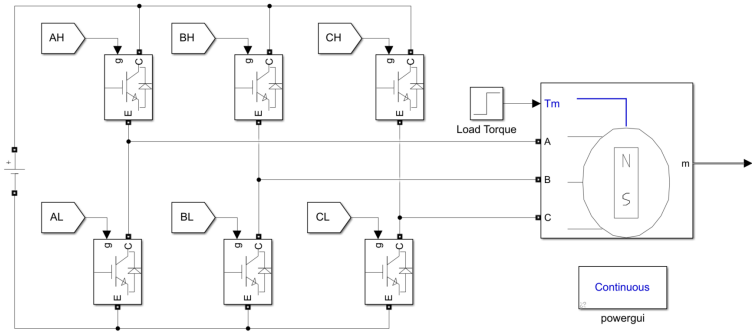

Figure 4.3: *The SIMULINK model of the inverter and the PMSM.*

A number of different motor parameters can be set in this model. The parameters are set in the initiation m-file. The different motor parameters that can be changed are listed in Table 4.1.

Parameter	Description
R_s	Stator phase resistance
L_s	Stator phase inductance
p	Number of pole pairs
K_t / K_e / λ_r	Torque constant / Back-emf constant / Rotor flux
J	Moment of inertia
B	Viscous damping
T_f	Static friction
T_l	Load torque
V_{dc}	DC-bus voltage
R_{on}	IGBT on-state resistance
t_d	Deliberate dead-time

Table 4.1: *Different motor and inverter parameters used by the physical model of the PMSM and the inverter.*

4.1.3 Complete System

For the complete system it is important to close the loop so that feedback behaviours can be analyzed correctly. The signals from the motor that are fed back into the current controller are rotor angle, rotor speed and the motor currents for phase A and B. An overview of the complete system is shown in Figure 4.4

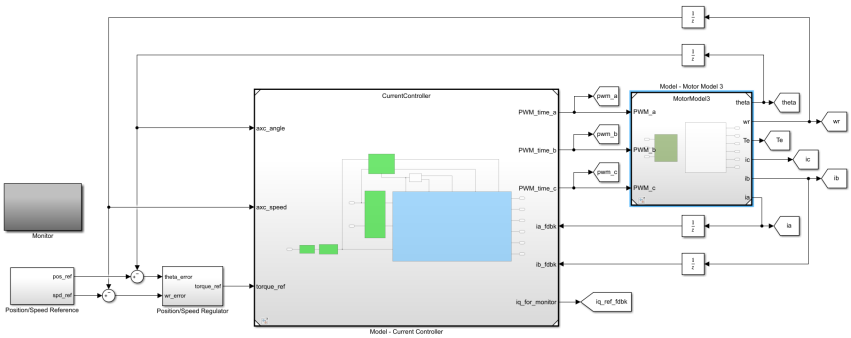

Figure 4.4: *Overview of the entire simulation model. The big block in the middle is the current controller and the smaller block on the right is the physical system model.*

The test case consisted of an acceleration for 0.8 s followed by a section of constant speed during 0.2 s and finally finished by a deceleration phase for another 0.8 s. Using a position and speed controller to determine the reference torque, the results shown in Figures 4.5 and 4.6 were obtained.

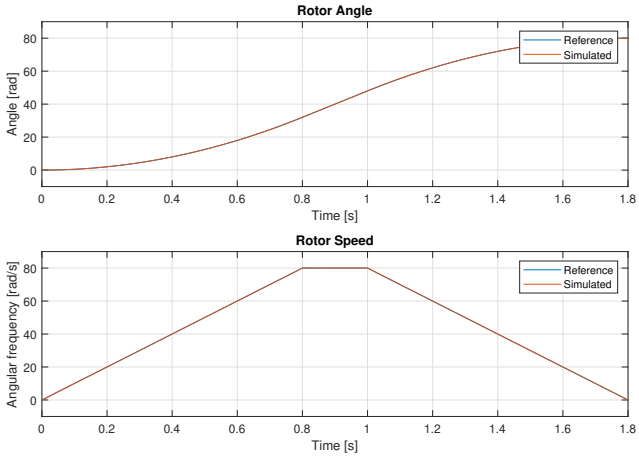

Figure 4.5: *Final simulated rotor angle and speed compared to references.*

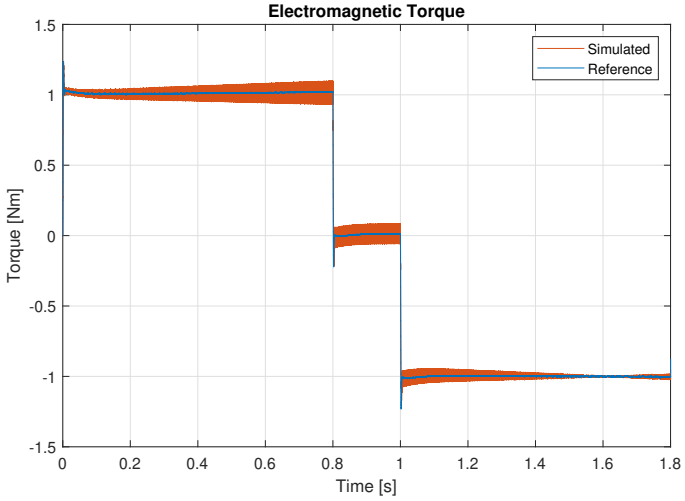

Figure 4.6: *Final simulated electromagnetic torque compared to reference. A negative torque is obtained when the motor is decelerating.*

The simulation model seems to follow the references well but it appears to be some speed dependent noise in the electromagnetic torque. The duty cycle for phase A from the current controller is shown in Figure 4.7.

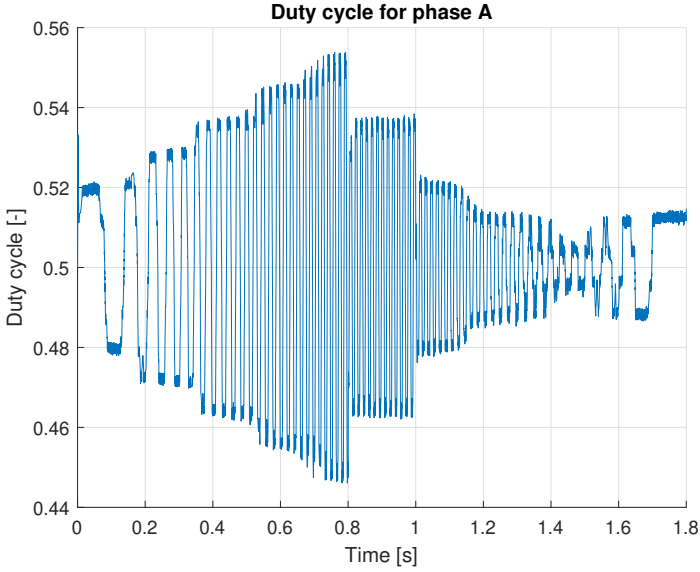

Figure 4.7: *Duty cycle for phase A when running the closed-loop simulation.*

The phase currents from a portion of the acceleration phase are shown in Figure 4.8. The currents are offset by 120° from one another. It can also be seen that the period of the sinusoidal currents gets shorter with time. The reason for this is that the currents are sampled in the acceleration phase. There are also no plateaus at the zero crossings since this simulation was run without any dead-time.

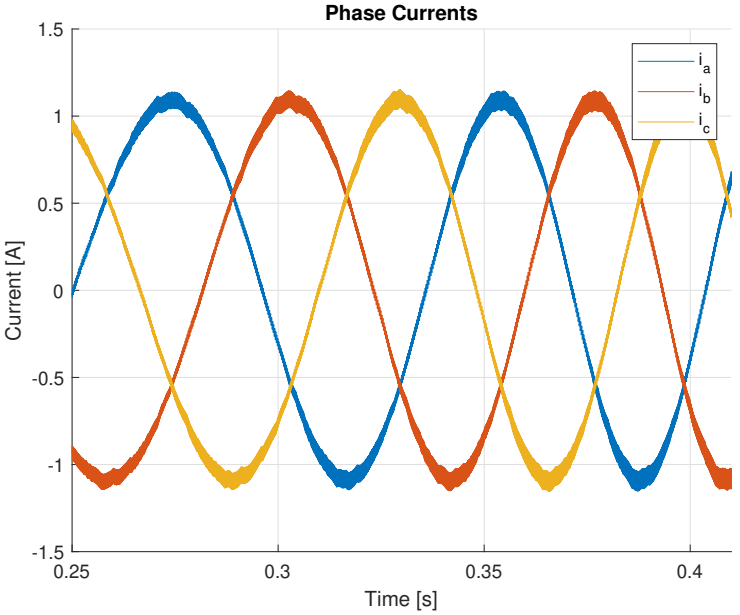

Figure 4.8: *The three-phase currents in the motor during a portion of the acceleration phase.*

4.2 Dead-Time Compensation

In this section the results regarding the performance of the different dead-time compensation algorithms are presented. For all methods, a figure showing the phase current with and without compensation is shown, for acceleration as well as deceleration. A figure of the frequency response from the reference quadrature current i_q^* to the actual simulated quadrature current i_q is also presented for each of the methods. The frequency response when using a dead-time of 2 µs, an input amplitude of roughly 2 A and no dead-time compensation shows a resulting bandwidth of $\omega_b = 150$ Hz of the current controller.

4.2.1 Algorithm 1 - Fixed Feedforward Compensation

As this method is divided into two sub-methods the results for each sub-method are presented individually.

Algorithm 1A - Compensate on Pulse Length

In Figure 4.9, the motor current of phase A when using Algorithm 1A is shown. For both acceleration and deceleration the method compensates for the plateau occurring at the zero-crossing. During the deceleration phase the signal appears to get a bit more noisy. From the frequency response in Figure 4.10 it is clear that the bandwidth of the current controller increases with this algorithm. The improved bandwidth appears to be $\omega_b = 350$ Hz, which is an improvement of roughly 200 Hz from the non-compensated case.

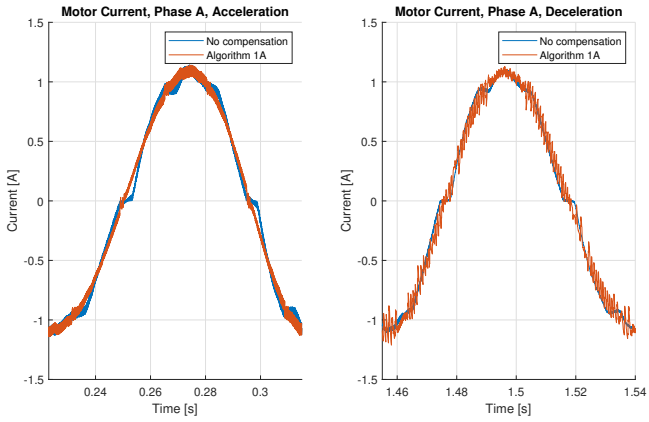

Figure 4.9: *Motor current i_a without compensation compared to Algorithm 1A. Acceleration phase to the left and deceleration phase to the right.*

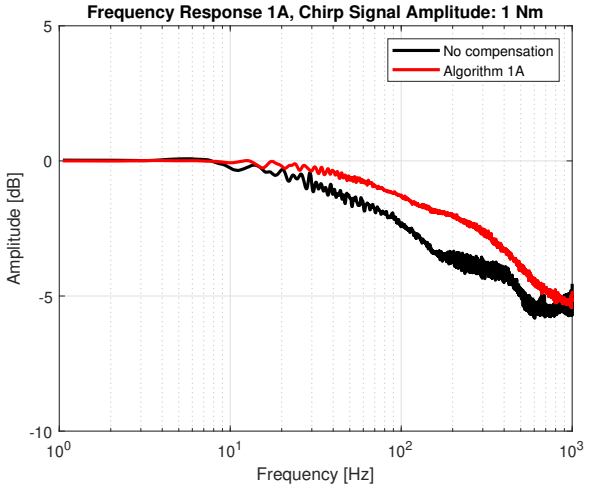

Figure 4.10: *Frequency response of the system with Algorithm 1A compared to no compensation. The amplitude of the chirp signal is 1 Nm (≈ 2 A).*

Algorithm 1B - Compensate on Reference Voltage

The results appear similar to those of Algorithm 1A. In Figure 4.11 the phase A current using Algorithm 1B is shown, compared to the non-compensated case. The algorithm seems to reduce the plateau that occurs at zero-crossings in this case as well. Similar to Algorithm 1A, during deceleration the signal appears to be more noisy. From the frequency response in Figure 4.12 it can be seen that the resulting bandwidth reaches roughly $\omega_b = 275$ Hz.

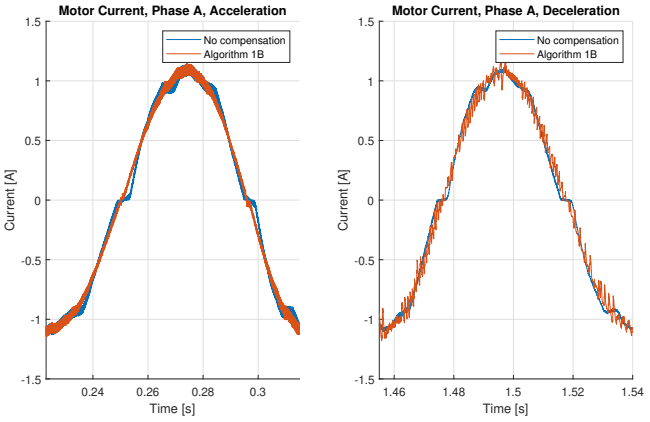

Figure 4.11: *Motor current i_a without compensation compared to Algorithm 1B. Acceleration phase to the left and deceleration phase to the right.*

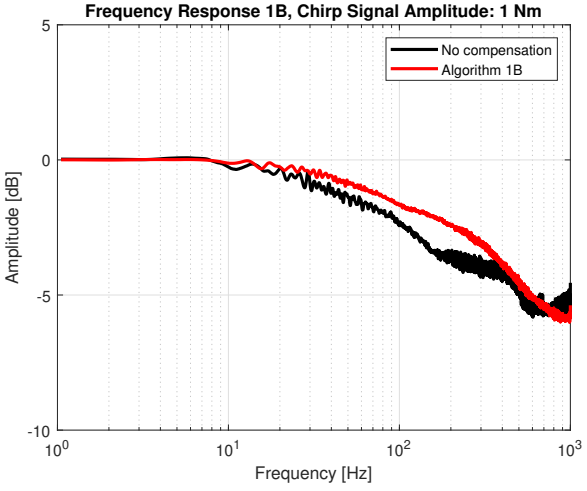

Figure 4.12: *Frequency response of the system with Algorithm 1B compared to no compensation. The amplitude of the chirp signal is 1 Nm (≈ 2 A).*

4.2.2 Algorithm 2 - Variable Feedforward Compensation

In Figure 4.13 the phase A current for Algorithm 2 is shown. This algorithm also results in a reduction of the zero-crossing plateau in the acceleration phase. However, the reduction of the plateau is a lot less prominent during the deceleration phase. On the other hand, the current is actually less noisy in the deceleration phase than in the acceleration phase. From the frequency response in Figure 4.14 the bandwidth of the system is improved compared to the non-compensated case, roughly reading $\omega_b = 350$ Hz.

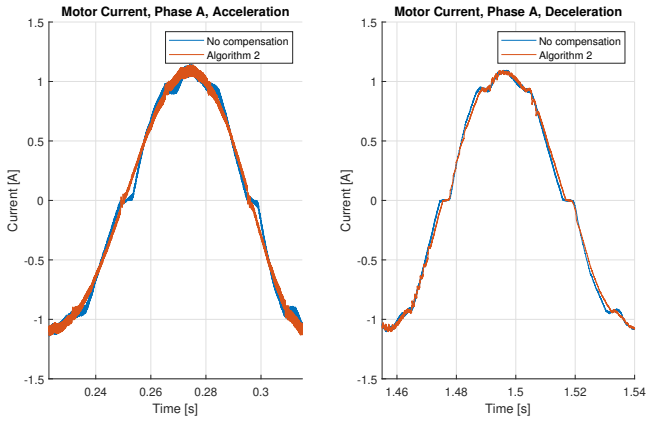

Figure 4.13: *Motor current i_a without compensation compared to Algorithm 2. Acceleration phase to the left and deceleration phase to the right.*

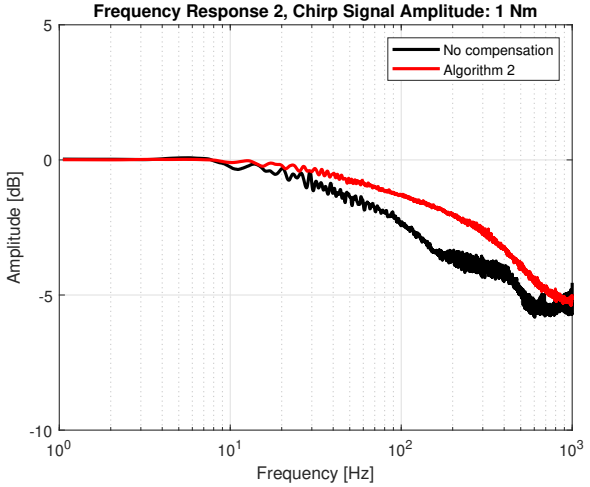

Figure 4.14: *Frequency response of the system with Algorithm 2 compared to no compensation. The amplitude of the chirp signal is 1 Nm (≈ 2 A).*

4.2.3 Algorithm 3 - Voltage Disturbance Observer

The usage of an observer in the dead-time compensation algorithm seems to give good results. Figure 4.15 shows the phase A current when using Algorithm 3 compared to the non-compensated case. The zero-crossing plateau is slightly reduced in both the acceleration phase and the deceleration phase. However, the curve is still not quite sinusoidal. As with Algorithm 2, the current is actually less noisy in the deceleration phase. Figure 4.16 shows the frequency response of the system while using Algorithm 3 compared to the non-compensated case. This algorithm results in a system bandwidth of roughly $\omega_b = 450$ Hz.

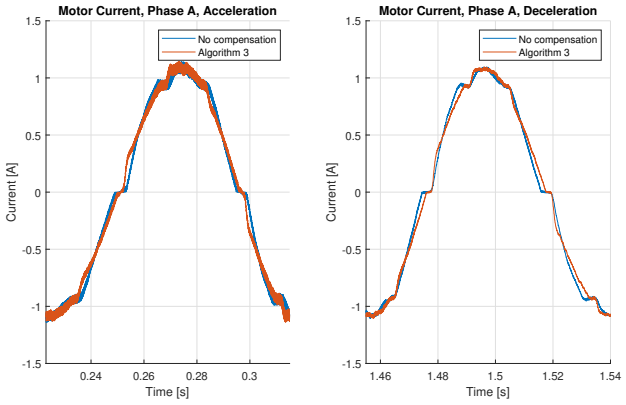

Figure 4.15: *Motor current i_a without compensation compared to Algorithm 3. Acceleration phase to the left and deceleration phase to the right.*

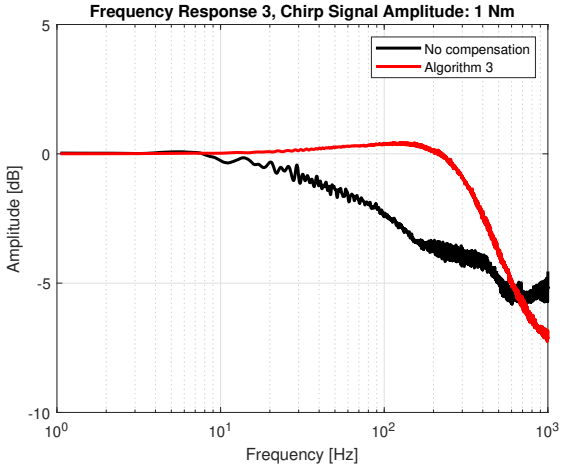

Figure 4.16: *Frequency response of the system with Algorithm 3 compared to no compensation. The amplitude of the chirp signal is 1 Nm (≈ 2 A).*

4.2.4 Comparison

It is clear that all compensation methods show an improvement both concerning shape of the phase currents and bandwidth of the system. In Figure 4.17, the frequency responses of all algorithms are shown together for comparison.

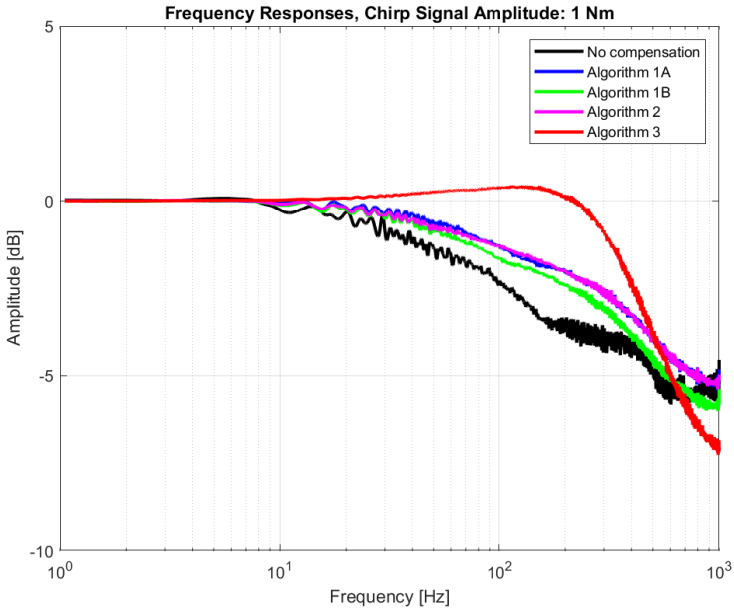

Figure 4.17: *Frequency response of the system for all the algorithms. The amplitude of the chirp signal is 1 Nm (≈ 2 A).*

All the responses from Figure 4.17 were obtained with a chirp signal with an amplitude of 1 Nm, or roughly 2 A, as input. The frequency responses of all the algorithms have also been obtained for two more input amplitudes. The responses for all algorithms with a chirp signal amplitude of 5 Nm, or roughly 11 A, are shown in Figure 4.18 and the responses for all algorithms with a chirp signal amplitude of 0.5 Nm, or roughly 1 A, are shown in Figure 4.19.

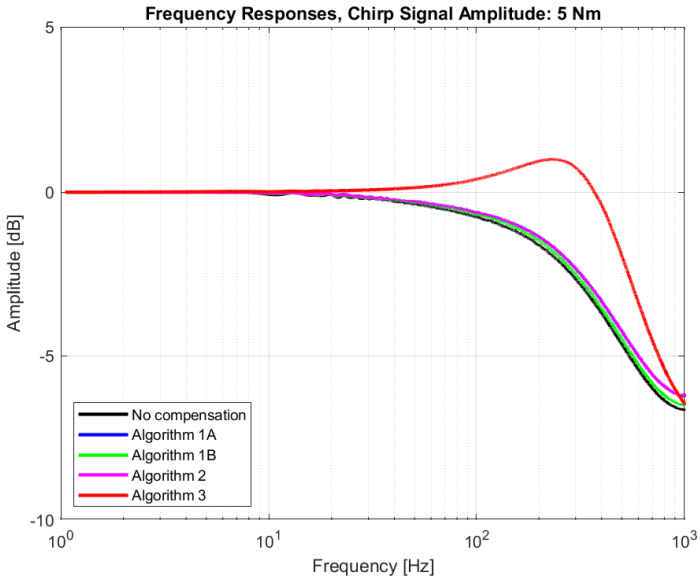

Figure 4.18: *Frequency response of the system for all the algorithms. The amplitude of the chirp signal is 5 Nm (≈ 11 A).*

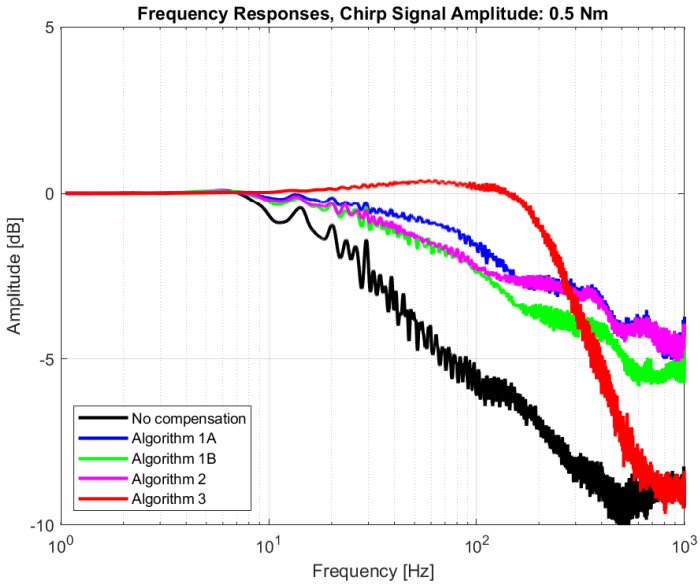

Figure 4.19: *Frequency response of the system for all the algorithms. The amplitude of the chirp signal is 0.5 Nm (≈ 1 A).*

One prominent result is that a larger current amplitude results in less noise in all cases. The system bandwidth is also increased for all algorithms with a larger current amplitude. For Algorithm 3, a resonance peak of roughly 1 dB is seen at around 250 Hz. The opposite effect is observed when decreasing the amplitude of the current. In this case a higher noise and a more narrow bandwidth was observed for all algorithms. A summation of the different bandwidths observed for all the algorithms in these three cases is shown in Table 4.2.

Case	No comp.	Alg. 1A	Alg. 1B	Alg. 2	Alg. 3
Amplitude 0.5 Nm ($\approx$ 1 A)	30 Hz	250 Hz	150 Hz	250 Hz	300 Hz
Amplitude 1 Nm ($\approx$ 2 A)	150 Hz	350 Hz	275 Hz	350 Hz	450 Hz
Amplitude 5 Nm ($\approx$ 11 A)	325 Hz	375 Hz	350 Hz	375 Hz	580 Hz

Table 4.2: *System bandwidths for the four different dead-time compensation algorithms as well as no compensation for three different amplitudes.*

5
Discussion

This chapter contains discussions and reflections concerning the results and the chosen methods.

5.1 Results

In this section the obtained results presented in Chapter 4 are discussed and evaluated.

5.1.1 Simulation Model

The results from the simulation model are best divided into two parts. The first part revolves around the current controller and the second part revolves around the physical system.

Current Controller

The results from the open-loop test of the current controller model showed very promising results. Both the inputs and the outputs of a real controller were measured and the output of the new current controller model using the same inputs as the real model was almost identical to the measured values. The current controller alone runs very quickly compared to the physical system and is thus in no risk of being the bottleneck of the model. There is a case to be made that the model of the current controller should be able to match the real controller exactly since it uses the same code. The tiny difference observed between the model and reality can be attributed to mainly two things; differences in initial conditions and the fact that the code used in the model is only a small part of the code necessary to run the whole system. The difference in initial conditions

is a minor issue since it is easy to simply change them in the model. The fact that the model of the current controller only uses a fraction of all the code used in a real system is more problematic. Some flags and parameters that are necessary to run the model need to be fabricated in order for the model to run. Still, as made apparent in Figure 4.2, the model of the current controller works very well given the right parameters.

Physical System

One of the goals with the model of the physical system was to be able to change motor parameters in a simple manner. This is easily done in the current implementation owing to an initiation file, but it is limited to the parameters listed in Table 4.1. These parameters are evidently enough to create a viable physical model that works together with the current controller, but it should be noted that some simplifications have been made. One specific simplification worth mentioning is the one concerning iron losses. Iron losses were only briefly mentioned in Section 2.3 but they often have a notable impact on the performance of a motor. They are typically greater at higher speeds and in bigger motors but are generally always present. Due to the complicated manner that these losses have to be modeled, they have been neglected in this model. This probably leads to a worse conformity between model and reality. This is a classic dilemma in modeling and a trade-off was made in favour of model stability and simulation speed.

Complete System

As seen in Figures 4.5 and 4.6, the rotor angle, rotor speed and torque all follow their references quite well. The responses for angle and speed are both smooth but the torque shows some noise. This noise appears to increase with increasing motor speed. A possible explanation could be that the amount of motor current zero-crossings increase with an increasing speed, which combined with the fact that the dead-time effects are more pronounced at zero-crossings results in a higher dead-time effect at higher speeds. It should also be noted that the motor shows signs of an almost ideal behaviour since the rotor speed during the constant speed portion of the test case can be maintained with essentially no torque. In a real motor there would be more losses, for instance due to friction. The simulation model allows the user to set a static friction as well but in the test case it was set to zero which could explain the near-ideal behaviour. The shape of the phase currents in Figure 4.8 are also promising with no notable distortions.

One challenge during the set up of the complete system was selecting appropriate motor parameters. The system seems to be rather sensitive to which motor parameters are used and some combinations showed great results while others did not perform as well. In general it was found that 1 μs was a sufficiently small time step to capture the essential dynamics but a smaller time step resulted in less noise.

5.1.2 Dead-Time Compensation

As presented in Section 4.2, all the tested dead-time compensation algorithms show promising results in one way or another. Two things they all have in common are that they seem to increase the bandwidth of the current controller and reduce some dead-time plateaus.

Frequency Response

It is interesting to observe the behaviours of the system when applying different amplitudes on the input signal. This provides information about the behaviour of the system when exposing it to both high and low currents without changing any motor parameters. In Figure 4.19, where the input amplitude was approximately 1 A, the frequency responses are significantly noisier than in the case where the input amplitude was roughly 2 A. If the experiment had been done with a smaller motor, which implies smaller currents, this phenomenon might not have been so apparent. The input magnitude in this case seems to be a bit too small for this particular motor. Another important note is that Algorithms 1A and 1B add a constant compensation term based on the dead-time and current direction to the pulse length and the reference voltage respectively. Running the system with a low current reference will cause the compensation term to be large relative to the reference. The noise could therefore occur as a result of overcompensation. A similar explanation could be applied to Algorithms 2 and 3, although for Algorithm 2 the added term is actually dependent on the magnitude of the reference. This highlights the difficulty to find a good compensation method that works for small currents as well. For Algorithm 3, the compensation is based on the motor parameters and adds a term that is based on the deviation from the previous sample. This is probably why Algorithm 3 is less noisy than the others. A common way to reduce noise is to introduce a filter. However, filters tend to slow down simulations and they introduce an unwanted time delay into the system and should thus be used with caution. The bandwidth is also decreased when using a low amplitude input compared to a high amplitude input. However, all of the algorithms show better results in terms of a larger bandwidth than if no compensation is used at all.

The noise is highly reduced in the case with an input amplitude of roughly 11 A. All algorithms have an increased bandwidth for this high-amplitude case. The reduction of noise could be explained with the same but opposite reason described above. The added compensation is smaller compared to the reference. In addition to the reduction of noise, Algorithm 3 shows a small resonance peak at around 250 Hz. Resonance peaks are usually unwanted features in high precision systems. In this case however, the peak is rather small, around 1 dB, and will probably not make a predominant impact on the system. Overall, Algorithm 3 has the most prominent improvement of all the tested algorithms concerning the frequency response.

Phase Currents

The reduction of the zero-crossing plateau due to the dead-time is a good result from the dead-time compensation algorithms. However, the different algorithms showed varying results. Algorithms 1A, 1B and 2 all give a smooth reduction of the irregularities in the sinusoidal shape during acceleration. During the deceleration of the motor, noise appears for both Algorithms 1A and 1B. When observing the switching times for phase A in Figure 4.7, it is clear that during the deceleration phase the magnitude of the switching times is lower than in the acceleration phase. The added compensating factor is however the same in the deceleration phase as in the acceleration phase for Algorithms 1A and 1B. Thus, the added term will have a great influence during the deceleration phase. This could be an explanation to why these two algorithms are noisier in the deceleration phase than in the acceleration phase.

Algorithm 2 shows a smooth phase curve for the acceleration and no increase in noise during the deceleration. However, the shape of the phase curve deteriorates somewhat during the deceleration. This is probably linked to the phenomenon described above where the magnitude of the reference voltages is lower during the deceleration. This leads to a smaller compensation and the phase curve looks more like the uncompensated phase curve. The phase curve from Algorithm 3 does not look as good as the first two algorithms as it barely reduces the zero-crossing plateaus. It does however reduce the noise a bit, and similarly to Algorithm 2, there is no extra noise during deceleration.

5.2 Method

In this section the methods used in this thesis are discussed and evaluated.

5.2.1 Simulation Model

The method used to develop the simulation model is best divided into two parts. The first part is the model of the current controller and the second part is the model of the physical system.

Current Controller

The fact that the model of the current controller uses exactly the same code as the real system is advantageous in many ways. Firstly, once the process of importing data types and functions from the language of C into SIMULINK was determined, building the controller was rather simple. The risk of making implementation mistakes was actually lower than if the modeling would have been done from scratch in SIMULINK since the correct functions and methods already existed in C and no mapping or alteration was needed. Secondly, it is also a good way to make sure that the components of model of the current controller behaves in ex-

actly the same way as the components in the real system.

A slight drawback with this method is that the initial hurdle to get the import to work smoothly was quite large and time consuming. This type of implementation also makes troubleshooting a bit more complicated. There were some obstacles in the validation process of the components in the current controller, mainly connected to the inability to put break points in the simulation. There were generally workarounds for all the encountered problems but it was still a bit more complicated than it would have been with a pure SIMULINK model. Additionally, the measurements for the verification would have been easier to collect if working from the office was an option. However, due to the situation with covid-19 this was not possible and the measurements had to be collected via our supervisors at ABB.

Physical System

The method of starting simple and then advancing turned out to be a good approach to the modeling of the physical system. The first model of the inverter and the motor was not even done with SIMSCAPE components but rather with transfer functions. At first, only the electrical part of the motor was considered, essentially resulting in a model of a Wye connection. This was useful to get some insight concerning how the calculated switching times from the current controller were converted into motor voltages and consequently motor currents. Eventually, a model was built with SIMSCAPE components. The choice to use a pre-existing model of a PMSM was possible because of the extensive documentation on the MathWorks website [12]. The documentation made it possible to parameterize the component to make it behave like the dynamic model described by Equations (2.25) - (2.28).

A strength of this method is that there is always an option to stop the development once the model is able to capture the relevant dynamics. This reduces the risk of developing an unnecessarily complicated model, which might for instance cause longer simulation times than necessary to study the phenomenon at hand. This became obvious when modeling the IGBTs. It is possible to make very complicated models of the transistors but the setup shown in Figure 4.3 was evidently sufficient to capture the differences between the compensation algorithms.

5.2.2 Dead-Time Compensation

For the dead-time compensation algorithm development, the chosen approach was to perform a literature study including several ideas and concepts and from these choose at least three different implementation strategies. This plan seems to have provided good results. All algorithms address the issue of dead-time in different ways. What this method lacks is deeper insight into a specific algorithm which was traded off for a broader study of multiple ideas. The choice to evaluate the algorithms by observing the phase currents and the frequency response was

also made to be as descriptive as possible yet still easy to understand. The results from the dead-time compensation algorithms are all obtained through the simulation environment created as the first part of this thesis. A natural next step is of course to test these algorithms on real systems since the end goal is to improve the performance of an actual robot. Results from such tests would improve the results and conclusions of this thesis, but it was unfortunately not possible due to time and covid-19 constraints.

6 Conclusions

This chapter summarises the final conclusions of the thesis and gives a foundation for what future efforts might benefit to focus on.

6.1 Simulation Model

In Chapter 1 of this thesis, a wish for a simulation environment that allows testing of new algorithms and ideas was presented. The questions considered were:

- How can a simulation model of a discrete current controller and a continuous physical system be modeled to provide accurate results with a reasonable simulation time?
- What is the best way to make use of legacy code written in C in a SIMULINK model?

The simulation model was implemented and the results show that the model is running as expected. To get results similar to the real system and to be able to observe relevant system behaviours, the largest accepted time step of the simulation is 1 μs. For this level of model complexity, using C Caller blocks in the implementation seems to be a solid approach. It is rather easy to implement and gives accurate results. Further extending the simulation model with a model of the physical system that uses SIMSCAPE components and is running as a continuous system worked well. By closing the loop with both the current controller and the physical system a working simulation model of the entire drive unit system is achieved. It can be concluded that both the questions above have been answered in this report.

6.2 Dead-Time Compensation

The simulation model was subsequently used to test system behaviours when applying the different dead-time compensation algorithms. In Chapter 1, a wish to develop a successful dead-time compensation algorithm was presented. This wish resulted in the following questions:

- How does the dead-time affect the behaviour of the system in terms of shape and behaviour of the phase currents and bandwidth of the current controller?
- What compensation methods seem useful to improve the behaviour of the system in terms of the shape and behaviour of the phase currents and the bandwidth of the current controller?

Firstly, the effect of dead-time is prominent in all figures related to the dead-time compensation in this report. Due to the dead-time, the phase current curves get a plateau whenever one of the phases changes polarity. Another unwanted effect is the reduction of the current controller bandwidth that appears when introducing a dead-time. Secondly, all compensation algorithms improved the system behaviour both in terms of the shape of the phase currents and the bandwidth. Concerning the bandwidth, Algorithm 3 performed the best out of the tested algorithms. Algorithm 2 creates the smoothest phase current curve out of the tested algorithms. Both the questions above have thus been answered in this thesis.

6.3 Future Development

If there was more time to improve the simulation model and the compensation algorithms, some adjustments could be interesting to test. These are presented below.

6.3.1 Simulation Model

The simulation model in its current state is capable of simulating a lot of interesting scenarios. However, when running the model now there are still some minor bugs that could slow down the simulation. Also, not all of the chosen implementation methods are optimized for simulation speed. The model can, of course, still run with an acceptable simulation time, but if more time on the project was given maybe the simulation time could be reduced. Additionally, there is always a possibility to build a more complex physical model. This could increase the possibility to add more parameters into the system and therefore include more dynamics. For instance, being able to simulate iron losses would be a big advantage. An investigation into the lack of performance when using certain sets of motor parameters would also be interesting to consider.

6.3.2 Dead-Time Compensation

The field of dead-time compensation is vast and there is a wide range of literature available. It is possible to spend much more time looking into different methods and concepts. Something that would be particularly interesting as a continuation of this work would be to look deeper into one of the algorithms, for instance Algorithm 3, as this thesis merely scratches the surface of its potential. Of course, testing the algorithms on real robots is another obvious next step. Another interesting idea is that since all the algorithms have their advantages and disadvantages, it could be natural to investigate if a fusion of algorithms is possible. The substantial increase of bandwidth from Algorithm 3 could maybe be achieved together with the smoothness of the phase currents provided by Algorithm 2.

Appendix

A

Field-Weakening

This is a brief summary of the field-weakening operation sourced from [9]. A rotating motor will generate back-emf voltages across its coils. These voltages are proportional to the rotational speed of the motor, ω_r, but the speed reaches a limit when the generated back-emf is greater than the available voltage from the inverter electronics. In this state, the inverter is unable to supply currents to the coils resulting in the motor not generating any torque.

In order to solve this problem and achieve higher motor speeds, the back-emf must be reduced. The back-emf is not only dependent on the motor speed, but also the magnetic flux between rotor permanent magnets and stator coils. Thus, reducing the magnetic flux will reduce the back-emf and the speed at which the inverter enters saturation is increased. Current can now flow through the coils and the motor can still generate torque at higher speeds.

As mentioned in Section 2.2, the field-weakening is controlled with the i_d current. When running the motor under normal conditions where no field-weakening is required, i_d is kept at zero. However, if field-weakening is required it is desirable to reduce the magnetic flux. This is done by increasing the current in the d-direction while reducing the current in the q-direction, resulting in a higher top speed at the cost of produced torque.

B

IGBT as an Ideal Switch

This example is from Chapter 10 in [6]. Consider the circuit seen in Figure B.1. The IGBT will control the current i_C through the resistor R. v_0 is a DC voltage source. The characteristics of the IGBT are the ones shown in Figure 2.13. For this example it is also known that $v_0 >> (V_{CE})_{sat}$.

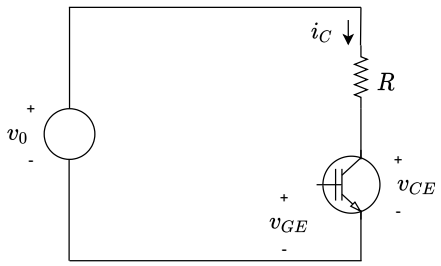

Figure B.1: *Example of a circuit controlled by an IGBT.*

In Figure B.1, using Kirchoffs voltage law results in the expression

$$v_0 = i_C R + V_{CE} \Rightarrow i_C = \frac{v_0 - V_{CE}}{R}. \tag{B.1}$$

Now, an $I - V$ diagram can be drawn. This is shown in Figure B.2. A, B and C are all operating points with different characteristics. At point A it holds that $v_{GE} < v_{th}$. Therefore, there is no collector current, the transistor is off and $v_{CE} = v_0$. Applying $v_{GE} > v_{th}$ moves the operation point up the load line until point B is reached. The collector current i_C increases whilst v_{CE} decreases. A further increase in v_{GE} will eventually result in reaching operating point C. Here the curves start to crowd together meaning a further increase of v_{GE} will only result in a fairly small decrease of v_{CE}. Operating under this condition, the voltage

across the transistor is equal to the saturation voltage $(v_{CE})_{sat}$. This shows that when applying a large gate-emitter voltage v_{GE}, a low voltage drop is obtained and these dynamics can be neglected.

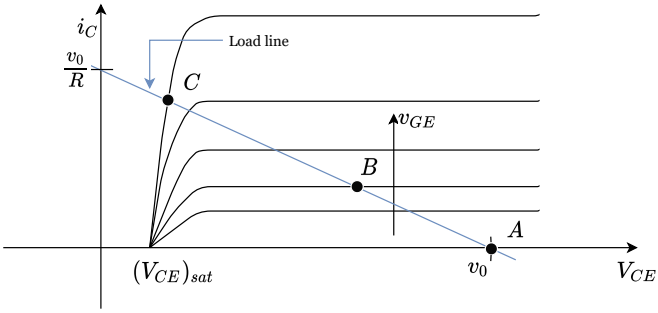

Figure B.2: *$I - V$ diagram for the example in Section 2.4.1 where the load line is $i_C = \frac{v_0 - v_{CE}}{R}$.*

Bibliography

[1] C Attaianese, D Capraro, and G Tomasso. A low cost digital SVM modulator with dead time compensation. In *2001 IEEE 32nd Annual Power Electronics Specialists Conference (IEEE Cat. No. 01CH37230)*, volume 1, pages 158–163. IEEE, 2001.

[2] Wei Chen, Baisong Li, Dianguo Xu, and Liang Cai. A dead-time compensation method for voltage source inverters. In *2019 22nd International Conference on Electrical Machines and Systems (ICEMS)*, pages 1–6. IEEE, 2019.

[3] John Chiasson. *Modeling and High-Performance Control of Electric Machines*. John Wiley & Sons, Inc., first edition, 2005. ISBN 0-471-68449-X.

[4] Jong-Woo Choi and Seung-Ki Sul. Inverter output voltage synthesis using novel dead time compensation. *IEEE transactions on Power Electronics*, 11 (2):221–227, 1996.

[5] Artur Cichowski and Janusz Nieznanski. Self-tuning dead-time compensation method for voltage-source inverters. *IEEE Power Electronics Letters*, 3 (2):72–75, 2005.

[6] A. E. Fitzgerald, Charles Kingsley, and Stephen D. Umans. *Electric Machinery*. McGraw-Hill, sixth edition, 2003. ISBN 0-07-366009-4.

[7] Myung-Joong Youn Hyun-Soo Kim, Hag-Wone Kim. A new on-line dead-time compensation method based on time delay control. (6):1184–1189, 2001.

[8] Hyun-Soo Kim, Hyung-Tae Moon, and Myung-Joong Youn. On-line dead-time compensation method using disturbance observer. *IEEE Transactions on Power Electronics*, 18(6):1336–1345, 2003.

[9] Ramu Krishnan. *Permanent Magnet Synchronous and Brushless DC Motor Drives*. CRC Press, first edition, 2010. ISBN 978-0-8247-5384-9.

[10] Yan-Siun Li, Po-Yi Wu, and Ming-Tzu Ho. Dead-time compensation for permanent magnet synchronous motor drives. In *2020 International Automatic Control Conference (CACS)*, pages 1–6. IEEE, 2020.

[11] Y-H Liu and C-L Chen. Novel dead time compensation method for induction motor drives using space vector modulation. *IEE Proceedings-Electric Power Applications*, 145(4):387–392, 1998.

[12] MathWorks. Mathworks documentation. URL `https://se.mathworks.com/help/`.

[13] Won Seok Oh, KP Hong, YT Kim, and Hee Jun Kim. Dead-time compensation of a current controlled inverter using the space vector modulation method. *International journal of electronics*, 80(2):277–289, 1996.

[14] J Zitzelsberger and W Hofmann. Space vector modulation with current based dead time compensation using kalman-filter. In *IECON 2006-32nd Annual Conference on IEEE Industrial Electronics*, pages 1533–1538. IEEE, 2006.