NTNU

Norwegian University of
Science and Technology

Control of a Permanent Magnet Synchronous Motor (PMSM) with constraints

Shiv Jeet Rai

Master of Science in Cybernetics and Robotics
Submission date: June 2017
Supervisor: Tor Arne Johansen, ITK
Co-supervisor: Jimmy Chau, Kongsberg Automotive As

Norwegian University of Science and Technology
Department of Engineering Cybernetics

ABSTRACT

The aim of the thesis is to design and implement a controller that controls the PMSM without violating the given constraints. This is especially with respect to the battery current.

Two approaches has been implemented in this paper, where both of them have shown to satisfy the given constraints. This was however on expense on the performance of some of the responses, especially $i_d(t)$. As a Surface PMSM (SPMSM) has been used, a nonzero value on $i_d(t)$ only contributes in heat increase in the motor. The approaches have shown different overall performance of these responses. The simulations in this paper was carried for a given test case where an angular position reference was followed. Both approaches have utilized a cascade of P- and PI-controllers, where one replaced the current controllers with a predictive controller, while the other utilized saturation blocks.

The predictive controller has been realized by using a Nonlinear Model Predictive Control (NMPC) structure, as the constraint for the battery current was found to be nonlinear. In this paper two versions of the NMPC has been looked at, where one considered constant angular velocity in the prediction horizon and the other did not. This was motivated by the assumption that a constant velocity in the controller might have significant reduction on the computational time. Based on the results however, this paper has shown that including angular velocity in the prediction horizon might be a better choice. This is in terms of the performance and computational time presented in the results. However as a framework that does not support c-code generation has been used, this paper cannot give any direct results on the computational effort the predictive controllers induce.

The work of this paper should therefore be regarded as an introductory on future work. This is if it is of interest to use the NMPC presented here for controlling the motor. The results alone are not of very big significance as the control structure currently is not deploy-able. However for further work, the results give insights in what to expect when utilizing a framework that supports c-code generation.

The other approach that have utilized saturation blocks can be used directly. However some additional tuning might be required if the performance is not sufficient.

SAMMENDRAG

Formålet med oppgaven er å designe og implementere en styringsmetodikk som styrer PMSM-en uten å bryte noen gitte restriksjoner. Dette er med spesiell vekt på batteristrømmen.

To metoder har blitt implementert i denne oppgaven, hvor begge har tilfredstilt de gitte begrensningene. Dette var imidlertid på bekostning av ytelsen til systemet, spesielt med tanke på $i_d(t)$. Som følge av at en "Surface PMSM" (SPMSM) har blitt brukt, vil en verdi ulik null for $i_d(t)$ bidra til varmeøkning i motoren.

Simuleringene i masteroppgaven ble gjort for et gitt test-case der en ønsket å følge en referanse i vinkelposisjon. Begge metodene har benyttet seg av en kaskade av P- og PI-regulatorer, der den ene erstattet strømregulatorene med en prediktiv regulator, mens den andre benyttet seg av metningsblokker.

Den prediktive regulatoren har blitt realisert ved å bruke en ulinær MPC (NMPC). Dette er som følge av at restriksjonen for batteristrømmen ble funnet til å være ulineær. I denne oppgaven har to versjoner av NMPC-en blitt sett på. Den ene har betraktet konstant vinkelhastighet i prediksjonen, mens den andre betraktet varierende vinkelhastighet. Dette var motivert av antakelsen om at en konstant hastighet i regulatoren kan ha en betydelig reduksjon på nødvendig regnekapasitet. Fra resultatene derimot, så ser man at versjonen som inkluderer vinkelhastighet i prediksjonen kan være et bedre valg. Men som følge av et rammeverk som ikke støtter c-kode generering har blitt benyttet, så kan ikke denne oppgaven gi et konkret svar på hvor stor regnekapasiteten for de prediktive regulatorene er.

Arbeidet bør derfor betraktes som en introduksjon for fremtidig arbeid der NMPC-strukturen har blitt brukt. Resultatene alene er ikke av stor betydning ennå, men for videre arbeid med NMPC-strukturen gir resultatene innsikt i hva man kan forvente når et rammeverk som støtter c-kode generering er benyttet.

Den andre tilnærmingen som har benyttet seg av metningsblokker kan brukes direkte. Det er viktig å bemerke seg at noe tuning kan kreves hvis ytelsen ikke er tilstrekkelig.

ACKNOWLEDGEMENTS

This thesis ends a five-year Master's degree program in Cybernetics and Robotics.

I would first like to thank my family for always believing in me. Thank you for your endless love, support and encouragement.

A huge gratitude goes to my supervisors; Professor Tor Arne Johansen of The Department of Engineering Cybernetics at The Norwegian University of Science and Technology (NTNU), and Jimmy Chau of Kongsberg Automotive. Thank you for this opportunity and the guidance throughout the thesis.

I would also like to thank Kongsberg Automotive AS for their hospitality and ensuring that the necessary equipment were provided during the stay. I would like to express my gratitude towards Jimmy Engman, Joar Molvær and Ole Jhonny Wærp with whom I had the opportunity to share office with. Thank you for providing a great work atmosphere.

I am also grateful to Postdoctoral Fellow Torstein Ingebrigtsen Bø of The Faculty of Information Technology and Electrical Engineering at NTNU, and Research Fellow Andreas Reason Dahl of The Faculty of Engineering at NTNU for their guidance.

A gratitude goes to all my other friends who have supported me along the way.

Thanks for all your encouragement!

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS AND NOMENCLATURE

Abbreviations

AC	Alternating Current
DC	Direct Current
DMPC	Discrete Model Predictive Control
DTC	Direct Torque Control
FOC	Field Oriented Control
IGBT	Insulated Gate Bipolar Transistor
IPMSM	Interior Permanent Magnet Synchronous Motor
MPC	Model Predictive Control
MP-TC	Model Predictive Torque Control
NMPC	Nonlinear Model Predictive Control
PMSM	Permanent Magnet Synchronous Motor
SPMSM	Surface Permanent Magnet Synchronous Motor
PWM	Pulse Width Modulation

Nomenclature

B_v	Friction coefficient
i_d	Stator current in d-direction
i_q	Stator current in q-direction
J_m	Motor inertia
L_d	Stator inductance in d-direction
L_q	Stator inductance in q-direction
R_s	Resistance in the stator
T_l	Torque load
T_e	Electric torque
Z_p	Number of pole pairs
ω_e	Electric angular velocity
ω_m	Mechanical angular velocity
θ_e	Electric angular position
θ_m	Mechanical angular position
ϕ_{mg}	Amplitude of the flux induced

1

INTRODUCTION

1.1 Background and Literature Survey

Permanent Magnet Synchronous Motors (PMSMs) have been widely used in many industrial applications. This is especially in applications that requires high performance and precision like CNC machines and automatic production systems. This is due to their interesting set of characteristics including high efficiency, high torque/power density, maintenance-free operation and high torque-to-inertia ratio [1].

Due to extensive research on different types of PMSMs and comparison with similar motor types, the dynamic model is well known [2] [3]. This has led to the development of several control methodologies, dependent on the control purpose, that utilized different aspects of the motor drive in order to control the PMSM better. For instance, some controllers have utilized a cascade of proportional and integral gains (PI-controllers) with decoupling terms, where the decoupling terms are used to decouple the electrical and mechanical dynamics in the controller. Some others have replaced the inner PI-cascade with hysteresis comparators to achieve better control. The first and latter control methodologies are known as Field Oriented Control (FOC) and Direct Torque Control (DTC), and are the two most common ways to speed-control a PMSM nowadays [4].

Due to the development of better solid-state electronics and cheaper microprocessors, more advanced electric motor drives are now replacing older motor drives to gain better performance, efficiency, and precision. However as some applications requires that the electric motor is used with quickly altered load conditions, while being subjected to multiple stresses such as thermal and mechanical stresses, some precautions are required. These precautions can be thought of constraints the application has to satisfy in order to function properly. These can be current and voltage restrictions in form of a limited power supply or due to safety. Restrictions on the angular velocity and the torque could be considered as well due to safety and physical limits of the application.

These precautions can be satisfied in many different ways. For instance one could make the controller less aggressive by reducing the input and/or by making the reference signals less steep. This is achieved by including saturation and rate-limiter effects to increase the robustness of the controller. This would however reduce the performance of the controller as the controller is «slower» then before. Another way to handle the constraints can be by using a predictive control structure, that based on some initial conditions predicts how the behaviour will change. Based on the problem formulation and aspects such as solver, objective function and constraints, the predictive controller might result in a more optimal control. This usually the case when the optimization problem is convex [5].

As better and cheaper electronics, that may handle the computational cost the predictive controllers induces, exists; control methods like Model Predictive Control (MPC) has gained considerable attention in the recent years [6]. A literature survey has therefore been done in order to see how different researchers have designed their predictive controller(s), which constraints are being handled and which assumptions are made. For instance some researchers have replaced the inner PI-loop in the FOC with their controller [7], while some others have replaced the entire PI-cascade in their implementation [8]. Others have for instance designed controllers that utilizes a motor drive without an external PWM generator. MPC-structures of the latter type are known as Finite Control Set MPC (FCS-MPC) strategies and an example of it can be found in [9].

Overall there exist several types of MPC-structures that have shown to be promising alternatives to today's most popular concepts for speed-control for PMSM applications; FOC and DTC. This is particular in applications similar to those presented in [7] [8] [9]. To restrict the scope of this thesis MPC-structures like FCS-MPC are omitted from this paper. This is due to an external PWM-generator is used in the given application.

In collaboration with Kongsberg Automotive AS and NTNU it is of interest to look at control structures that handles the constraints for a given PMSM application, especially with respect to the battery current constraint. This is due to the requirements to the workspace the motor is to operate in. Due to confidentiality, information about what the motor is used to control is omitted from this paper. However this information is not crucial to understand the work in this paper. For now, it is sufficient to know that the motor is meant to follow an angular position reference that controls a component in an automotive vehicle. Currently, following the angular position gives high spikes in the battery current with the present control structure. According to Kongsberg Automotive AS, violating the battery current constraint would result in in-

ducing unwanted noise in other components in the vehicle, which may reduce the life expectancy of these components.

To cover a small range of how a predictive controller can be implemented for the given application, some specific MPC-structures from the literature are highlighted. These are control structures similar to the Model Predictive Torque Control (MP-TC) as presented by [7] and the Discrete Model Predictive Control (DMPC) as [9]. These are control strategies that have replaced the inner control loop and the entire cascade of PI-controllers in the FOC-scheme with a MPC respectively. Both of them have considered linear restrictions on the voltage and the current in a coordinate system known as $dq0$-frame. The frame is presented in detail in section 2.2, while the highlighted control structures are presented in a bit more detail in section 4.3.

Due to the nature of the given application, it is important to point at that the mentioned control strategies are not directly implementable. For instance the constraints for the control strategies in the literature are linear, while in the given application one of the constraints is found to be nonlinear. This is the battery current constraint. Due to the nonlinear constraint, the optimization problem is a nonlinear optimization problem. Thus a Nonlinear MPC (NMPC) has to be used rather than a MPC [10], as MPCs in general are characterized as predictive controllers with linear models and linear constraints [8].

In this paper, constraints on the voltage and the battery current are looked at. These are thought of as hard constraints, which means that exceeding the bounds are prohibited. In addition restrictions on the angular velocity and the torque are considered as well. These are thought of as soft constraint as a small overshoot is allowed. The constraints are presented in detail in section 2.5.

1.2 Problem Formulation

In collaboration with Kongsberg Automotive AS and NTNU it is of interest to look at control structures that handles the constraints for the given PMSM, especially with respect to the battery current constraint. It is therefore a natural choice to look at MPC-methodologies as the model for the motor is well known. Due to the nature of the given application the MPCs found are not directly suited. Nevertheless they offer valuable insight on aspects as design and assumptions which may be applicable for the given application. This can be in terms of if some of states can be regarded as constants in the predictive controller.

Due to the computational cost a predictive controller induces, it is of interest to see

whether utilizing saturation and/or rate-limiter might give a similar response. The aim of this paper is:

> Design and implement at least one control structure that controls the PMSM with the given constraints. Compare the control structure(s) with a standard controller that does not handles the given constraints in a sufficient manner. This is especially with respect to the battery current.

Part I

BACKGROUND AND THEORY

2

PERMANENT MAGNET SYNCHRONOUS MOTOR (PMSM)

This chapter presents some necessary theory about what a Permanent Magnet Synchronous Motor (PMSM) is and which transformations are normally done to achieve a simpler control structure. In addition the dynamic model along with the constraints studied in this paper is presented here.

2.1 ELECTRIC AC MOTOR

A PMSM, as any other electric motor, is an electrical machine that converts electrical energy into mechanical energy. To do so, the motor uses an AC supply to create a rotating magnetic field, which causes the rotor to rotate synchronously with the field. A three-phase system is commonly used to drive the motor. This is done due to several reasons such as; low maintenance cost and the ability to carry more load and power compared to a single-phase system [11]. See Figure 1 for an illustration, where a three-phase input is used to generate mechanical energy. In the figure, T is the electric torque, ω_m is the angular velocity (mechanical) and T_l is the motor load.

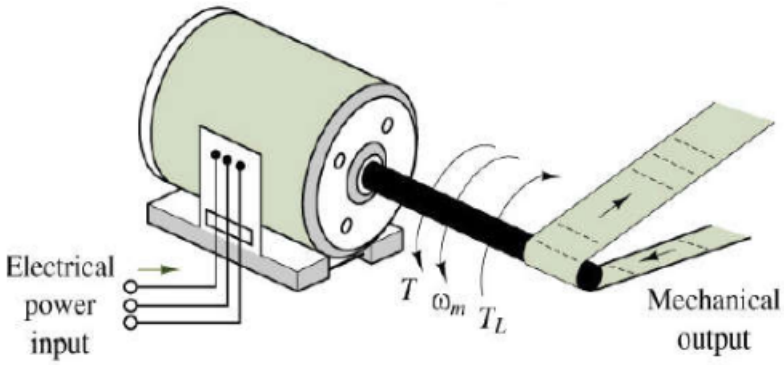

Figure 1.: Electric Motor (Courtesy to the course TMR4290 at NTNU)

An illustration of the working principle and the main components can be seen in Figure 2 and Figure 3. This is given for a motor with one pole pair. In short the working principle can be described as; current flowing through the windings creates a magnetic field that is a function of the values and the signs of the three phase currents. This energizes the magnet, causing the rotor to align to the new field. The strength of field is proportional with the current and number of pole pairs. For instance in Figure 3 at $t = t_1$ i_a is positive. This causes the current to flow in node a' and out of node a. Similarly i_b and i_c being equal and negative, results in i_b and i_c flowing in node b and c and out of b' and c'. That i_a has a higher absolute value is illustrated by the thickness of the dots and the crosses in the figure. By using right hand rule where one bends the fingers in the inflow direction of the current, one will see that the force is indeed between b' and c. The force direction is given by the thumb. Doing such for the different time values one have the orientation of the field. Notice that the dots in the figure represents the inflow while the cross represents the outflow.

Note: For a balanced three-phase system the following relations, among others, hold:

$$X_a(t) = A\cos(\omega t + \phi) = A\angle\phi \tag{1a}$$
$$X_b(t) = A\cos(\omega t + \phi - 120°) = X_a(t)(1\angle -120°) \tag{1b}$$
$$X_c(t) = A\cos(\omega t + \phi + 120°) = X_a(t)(1\angle 120°) \tag{1c}$$

where X is the signal, A is the amplitude, ω is the frequency and ϕ is the initial angle.

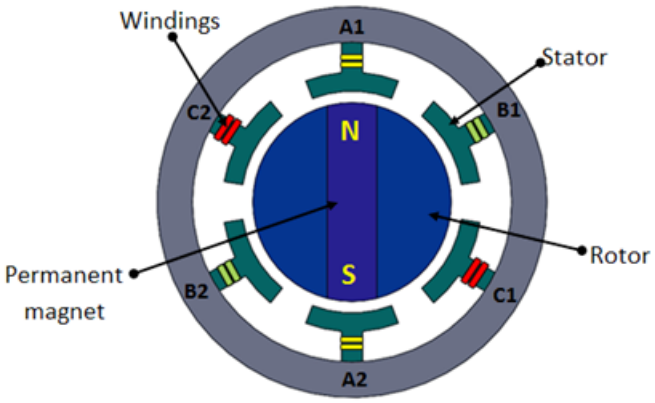

Figure 2.: Main components (Photo: [12])

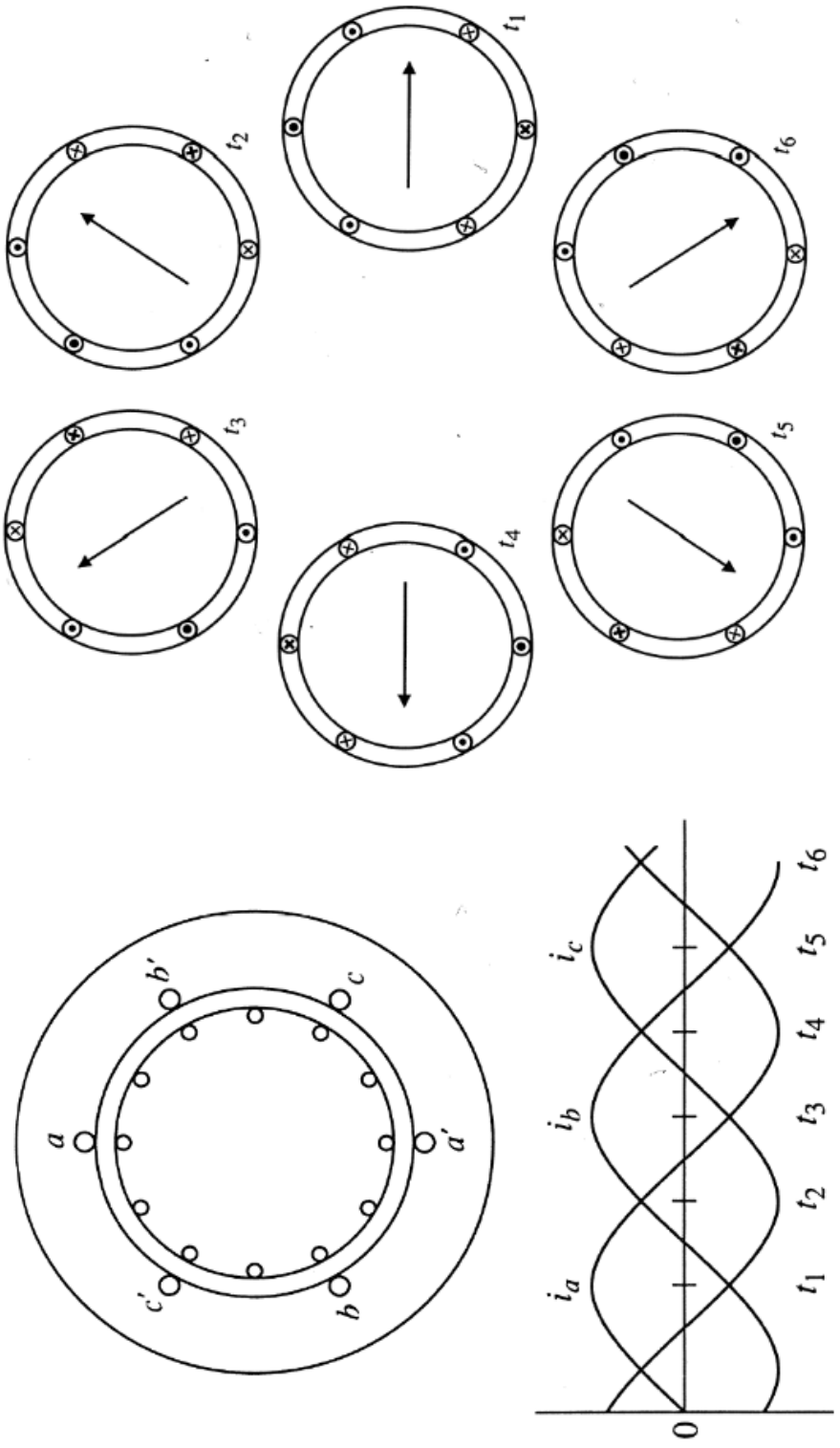

Figure 3.: Rotating of the magnetic field in a three-phase system (Courtesy to the course TMR4290 at NTNU)

2.2 Reference Frames

The motor is driven by a three-phase system. The voltages and currents are given by sinusoidal waveforms as in equations 1a-1c. These waveforms are dependent on the angular electric position of the motor, namely $\theta_e(t)$. In the literature it is quite common to do two transformations to achieve a simpler motor model to control. These are here referred as "alpha-beta-gamma" ($\alpha\beta\gamma$) and "d-q-0" ($dq0$) transformations, but are also known as Clarke and Park transformations in the literature.

There are two main benefits of doing the transformations. First of all, one would be able to represent the three signals in the abc-frame as two signals. This is achieved by doing a transformation from the abc-frame to the $\alpha\beta\gamma$-frame, where α and β are the new axes. This is the case for a balanced three-phase system where γ becomes a zero-axis. Secondly, one is able study the signals as DC signals rather than sinusoidal waveforms. From a control perspective this gives a «simpler» system to work. This is done by doing a transformation from $\alpha\beta\gamma$-frame to the dq0-frame, which is a rotating frame. See Figure 4 for an illustration.

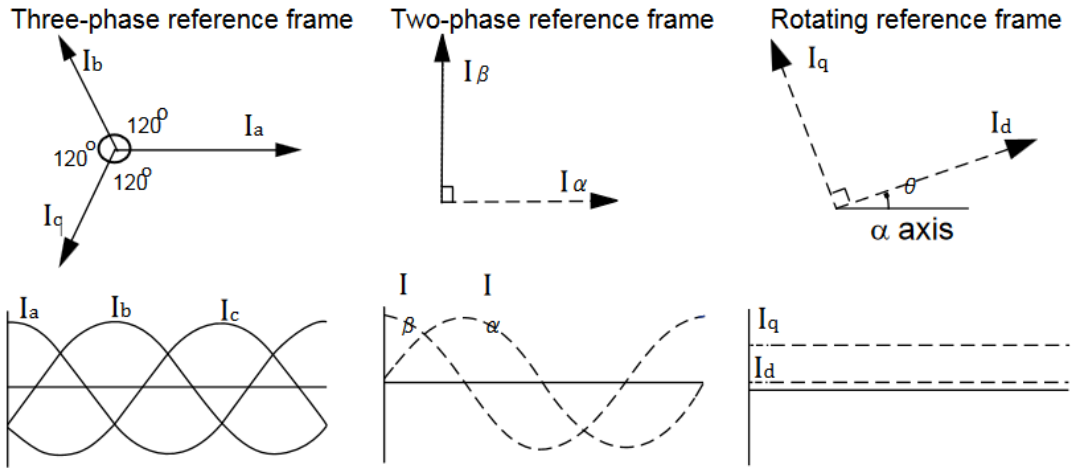

Figure 4.: Signal Comparison in the three different frames (Photo: [13])

2.2.1 $\alpha\beta\gamma$ TRANSFORMATION

The $\alpha\beta\gamma$ transformation is a mathematical transformation that uses the characteristics of a balanced three-phase system to rather study two signals than three. These characteristics are that the angle between each of the three phase voltages/currents studied are 120 degrees, and that the sum of these voltages/currents are 0. With the alignment of the α-axis perpendicular with the a-axis, and β-axis being the imaginary axis, the transformation matrix is:

$$T_{\alpha\beta\gamma} = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \tag{2}$$

See figure 5 for an illustration of the axis alignment. Similarly, the transformation back to abc-frame is known as Inverse $\alpha\beta\gamma$ Transformation. This is the inverse of equation 2.

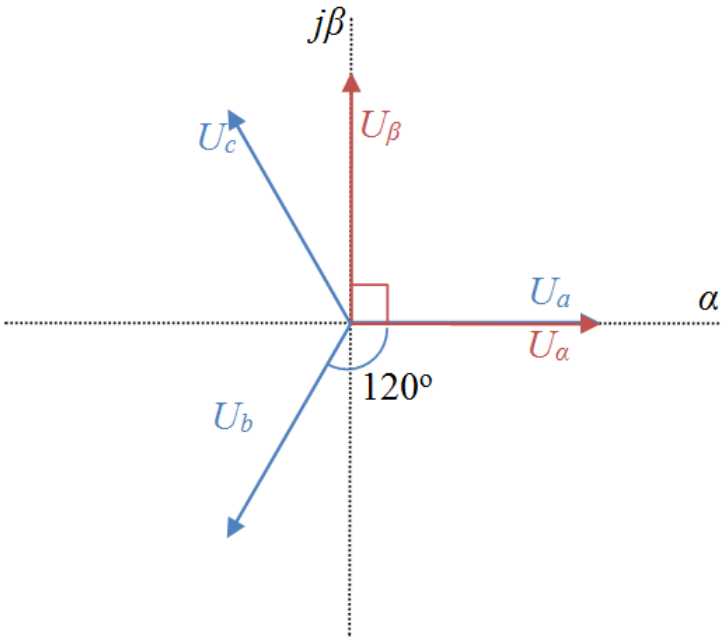

Figure 5.: $\alpha\beta\gamma$ coordinate frame

2.2.2 $dq0$ TRANSFORMATION

As the $\alpha\beta\gamma$ transformation "only" shifts the coordinate frame from a three-phase coordinate system into a stationary two-phase reference frame, the signals are yet sinusoidal. The $dq0$ transformation is further applied to rather treat the signals as stationary values (DC signals) than as sinusoidal waveforms. To achieve this, the $dq0$ transformation rotates the $\alpha\beta\gamma$ frame with the electric angle of the motor, $\theta_e(t)$. See figure 6 for illustration.

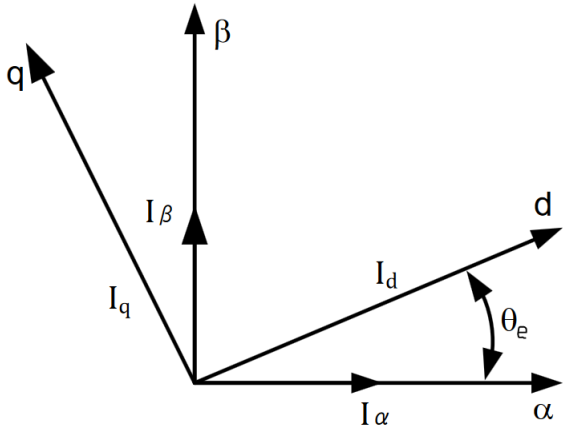

Figure 6.: $dq0$ coordinate frame

The transformation matrix is given as:

$$T_{dq0} = \begin{bmatrix} \cos\theta_e(t) & -\sin\theta_e(t) & 0 \\ \sin\theta_e(t) & \cos\theta_e(t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

Similarly, the transformation from dq0 to $\alpha\beta\gamma$ is known as Inverse $dq0$ Transformation. This is given as the inverse of equation 3 which is always invertible. The reason for this is that the matrix is a simple rotation about one axis. In this case it is the rotation about the 0-axis.

2.3 DC-AC Conversion

As the motor in this thesis is driven by using a DC power supply, a DC-AC conversion is in practice needed, see Figure 7 for an simple illustration. The conversion can by achieved by using IGBTs, which is a transistor type developed for high efficiency and fast switching [14]. IGBT stands for Insulated-Gate-Bipolar-Transistor. To control the transistors, Pulse Width Modulation (PWM) techniques can be used.

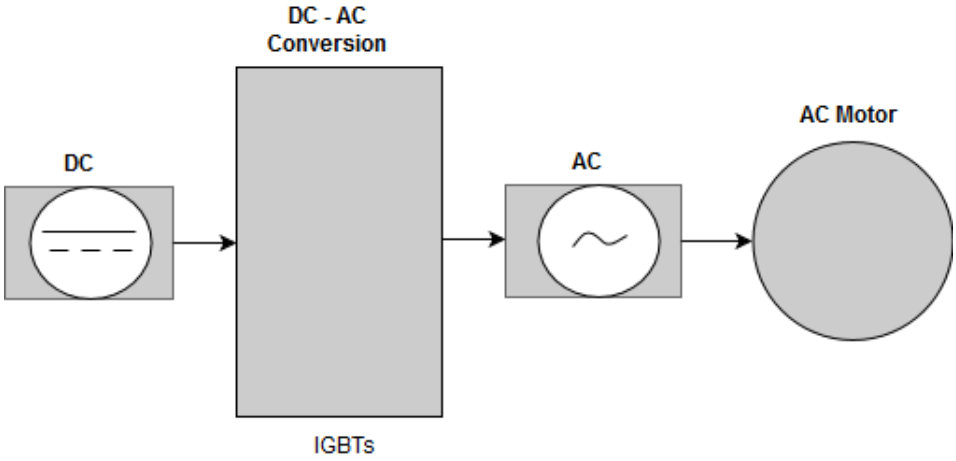

Figure 7.: DC-AC Conversion

In the given application a PWM-technique is used to feed the motor with the desired voltages from the DC source. This is done by controlling the ON/OFF-time of transistors. The transistors can be thought of switches that are opened and closed to obtain the desired three-phase voltages that are fed to the motor. Figure 8 shows an illustration of a three-phase IGBT topology, where the DC source is on the left side and the load (motor) is on the right side of the transistors.

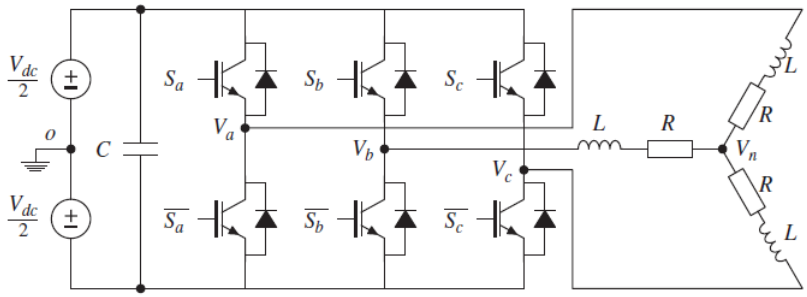

Figure 8.: Topology of three-phase-leg IGBT (Photo: [9])

2.3.1 SIMULINK MODEL

The conversion is realized by using a toolbox named Simscape Power Systems (formerly known as SimPowerSystems) in MATLAB with Simulink. The toolbox provides component libraries and analysis tools for modeling and simulating electrical power systems [15]. In this paper, the DC-AC conversion model is solely used to validate the model of one of the constraints. This is the battery current constraint. To do so, an average model for the DC-AC conversion is used. This is a model type that does not consider the switching losses in the transistors. Thus this Simulink model is only meant to say something on how accurate the model battery current is. See Figure 9 for a illustration of the implementation.

It is important to notice that the electronics, such as the voltage supply and the transistors, are in reality a part of the hardware. The same yields for the motor. They are therefore, in practice, simulated with a sample time that is sufficiently low to capture the dynamical behavior of the motor. The sample time is set by using the powergui-block in Figure 9. The powergui-block allows the user to choose among some methods to solve the circuit provided by Simscape Power Systems Toolbox. In this paper a discrete method named Tustin is used. As one will see later, reducing the sample time for the circuit resulted in a better fit to the battery current model.

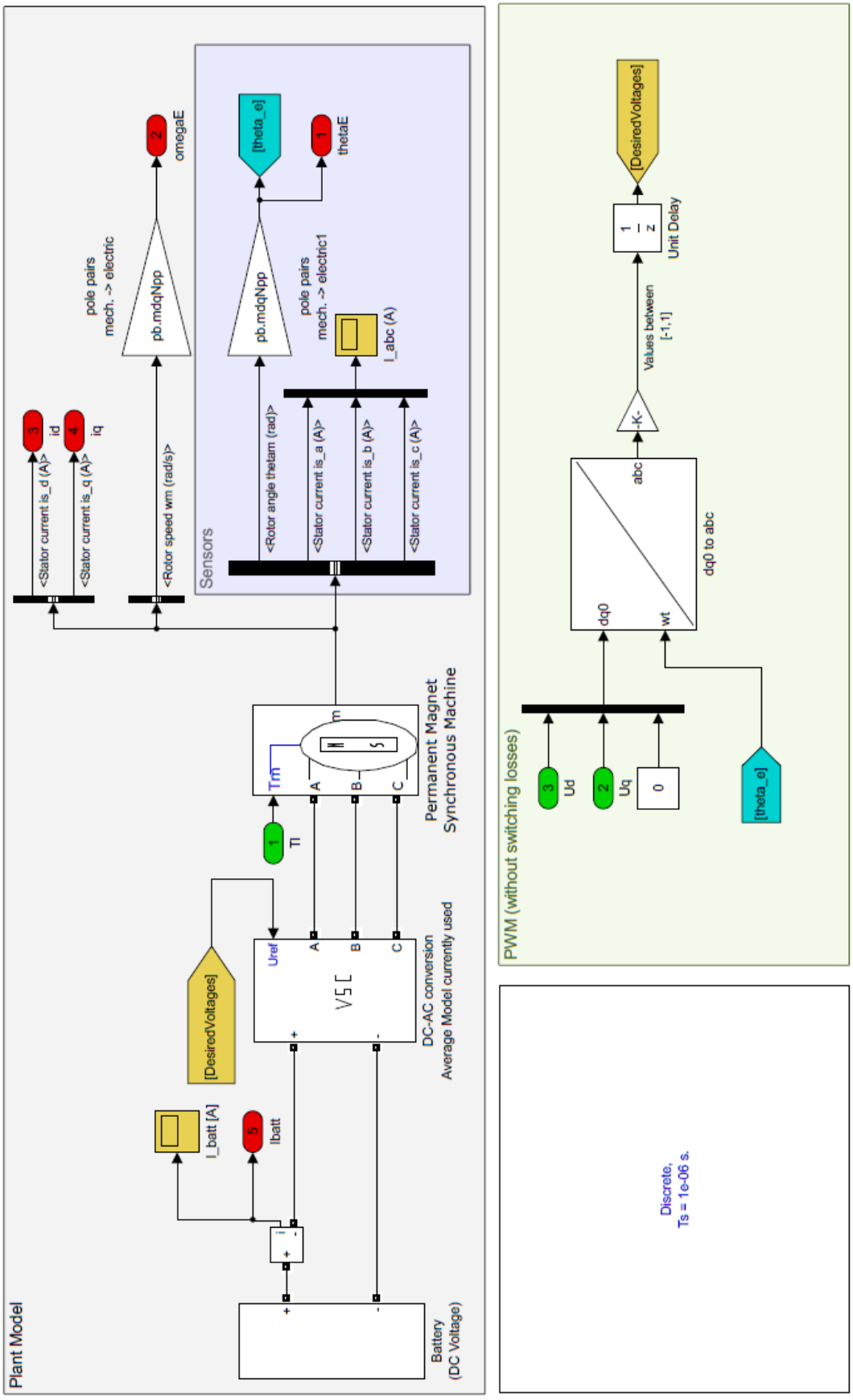

Figure 9.: Simulink Model of DC-AC conversion

2.4 PMSM Model

As briefly mentioned in the previous chapter, the model for the PMSM is well known in the literature. Thus a detailed explanation of the derivations behind the dynamical model used in this paper is not included. The reader is therefore suggested to look at [9] for thorough description of the equations.

The general PMSM model, for control purpose, in dq0 reference frame is given as:

$$\frac{di_d(t)}{dt} = \frac{1}{L_d}(V_d(t) - R_s i_d(t) + Z_p \omega_m(t) L_q i_q(t)) \tag{4a}$$

$$\frac{di_q(t)}{dt} = \frac{1}{L_q}(V_q(t) - R_s i_q(t) - Z_p \omega_m(t) L_d i_d(t) - Z_p \omega_m(t) \phi_{mg}) \tag{4b}$$

$$\frac{d\omega_m(t)}{dt} = \frac{1}{J_m}(T_e(t) - B_v \omega_m(t) - T_L(t)) \tag{4c}$$

$$\frac{d\theta_m(t)}{dt} = \omega_m(t) \tag{4d}$$

$$T_e(t) = \frac{3}{2} Z_p(\phi_{mg} i_q(t) + (L_d - L_q) i_d(t) i_q(t)) \tag{4e}$$

with the following relation:

$$\theta_e(t) = Z_p \theta_m(t) \tag{5}$$

where θ_m is the mechanical rotor angle.

2.4.1 Surface PMSM (SPMSM)

In the literature it is quite common to differentiate between two types of PMSMs, namely Surface PMSM (SPMSM) and Interior PMSM (IPMSM). The main difference between them is that SPMSM has a uniform air gap and no saliency, while IPMSM is a salient motor with a non-uniform air gap. For the mathematical model this means:

Table 1.: Main difference between SPMSM and IPMSM

SPMSM	IPMSM
$L_d = L_q$	$L_d < L_q$

Thus the bilinear term in equation (4e) can be removed for a SPMSM.

The motor used in this thesis is a SPMSM and will be from now on be referred to as PMSM throughout the paper. For simplicity the general PMSM model, without the bilinear term in equation (4e), is rewritten below. Note that L_d and L_q is not replaced by a common parameter. This is done on purpose, as it will be easier to implement the necessary changes when rather a IPMSM is used.

PMSM model (SPMSM)

$$\frac{di_d(t)}{dt} = \frac{1}{L_d}(V_d(t) - R_s i_d(t) + Z_p \omega_m(t) L_q i_q(t)) \tag{6a}$$

$$\frac{di_q(t)}{dt} = \frac{1}{L_q}(V_q(t) - R_s i_q(t) - Z_p \omega_m(t) L_d i_d(t) - Z_p \omega_m(t) \phi_{mg}) \tag{6b}$$

$$\frac{d\omega_m(t)}{dt} = \frac{1}{J_m}(T_e(t) - B_v \omega_m(t) - T_L(t)) \tag{6c}$$

$$\frac{d\theta_m(t)}{dt} = \omega_m(t) \tag{6d}$$

$$T_e(t) = \frac{3}{2} Z_p \phi_{mg} i_q(t) \tag{6e}$$

with the following relation:

$$\theta_e(t) = Z_p \theta_m(t) \tag{7}$$

2.5 Constraints

Due to physical limitations and safety reasons it is desirable to ensure that the motor satisfies the given constraints.

2.5.1 Voltage Constraint

The voltage constraint is given as:

$$V_d^2(t) + V_q^2(t) \leq V_{\text{MAX}}^2 \tag{8}$$

where

$$V_{\text{MAX}} := \frac{V_{\text{Battery}}}{\sqrt{3}} \tag{9}$$

This is due to the limitation of the DC-AC conversion used in the application. As the constraint is given by a circular area, and thus is nonlinear, it is of interest to linear ap-

proximate it. This is especially with consideration to the predictive controller. Below two linear approximation methods are presented.

2.5.1.1 RECTANGULAR APPROXIMATION

In a rectangular approximation the circular area is replaced with a rectangular area. The idea is to assume a parameter $0 \leq \epsilon \leq 1$ that can be used to scale the height and width of the rectangular box. Using the same notation as [9], the rectangular box is given by:

$$V_d^{\text{MAX}}(t) = \sqrt{1-\epsilon^2}V_{\text{MAX}} \tag{10a}$$
$$V_q^{\text{MAX}}(t) = \epsilon V_{\text{MAX}} \tag{10b}$$

which gives

$$-V_d^{\text{MAX}}(t) \leq V_d(t) \leq V_d^{\text{MAX}}(t) \tag{11a}$$
$$-V_q^{\text{MAX}}(t) \leq V_q(t) \leq V_q^{\text{MAX}}(t) \tag{11b}$$

See figure below for illustration.

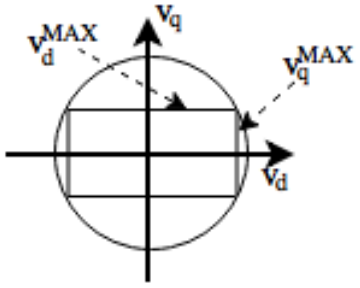

Figure 10.: Voltage constraint with rectangular approximation

The main advantage of using a rectangular approximation is that the constraints are easy to implement in a real-time control system. In addition the constraints in d-axis and q-axis are independent of each other, which enables an upper and lower bound on each axis respectively. This is not the case for the octagonal approximation. However, the shortcoming of this approach is it covers a small portion of the circular area, which may lead to a less optimal solution.

2.5.1.2 OCTAGONAL APPROXIMATION

In an octagonal approximation the circular area is replaced with a octagon instead. This gives better approximation than with a rectangular approximation, but one would rather have to handle more constraints compared to the four constraints in the prior approximation method. In addition, the constraints in a octagonal approximation are given as a linear combination of both axes. This might cause some inconvenience if one would like to treat the bounds on the voltages independently of each other.

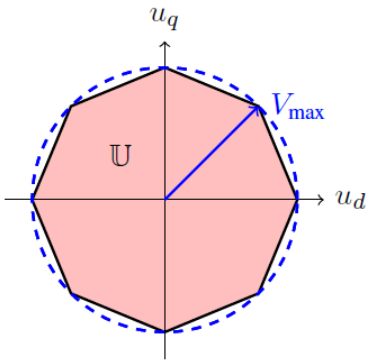

Figure 11.: Voltage constraint with octagonal approximation (Photo [7])

According to [9] the approximation is realized by using the equation below:

$$\begin{aligned} y - \frac{y_k - y_{k+1}}{x_k - x_{k+1}} x &\leq -\frac{y_k - y_{k+1}}{x_k - x_{k+1}} x_k + y_k, \quad k = 1,2,3,4 \\ y - \frac{y_k - y_{k+1}}{x_k - x_{k+1}} x &\geq -\frac{y_k - y_{k+1}}{x_k - x_{k+1}} x_k + y_k, \quad k = 5,6,7 \\ y - \frac{y_8 - y_1}{x_8 - x_1} x &\geq -\frac{y_8 - y_1}{x_8 - x_1} x_8 + y_8, \quad k = 8 \end{aligned} \tag{12}$$

where x_k and y_k are known coordinates, while x and y are signals.

For illustration purposes, an octagonal approximation of an unit circle is shown below. The coordinates values are also shown in Table 2. Notice that k=1 at the positive x-axis and increments counterclockwise.

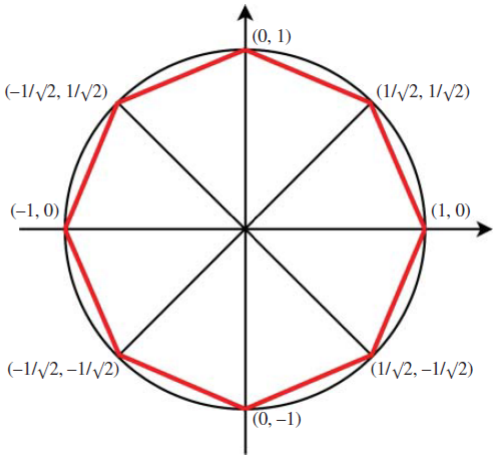

Figure 12.: Approximation of circular constraint area using the area of an octagon. (Photo [9])

Table 2.: The values of the coordinates for octagonal approximation

k	1	2	3	4	5	6	7	8
x_k	1	$\frac{1}{\sqrt{2}}$	0	$-\frac{1}{\sqrt{2}}$	-1	$-\frac{1}{\sqrt{2}}$	0	$\frac{1}{\sqrt{2}}$
y_k	0	$\frac{1}{\sqrt{2}}$	1	$\frac{1}{\sqrt{2}}$	0	$-\frac{1}{\sqrt{2}}$	-1	$-\frac{1}{\sqrt{2}}$

Equation 12 can be expressed in general form:

$$y + \alpha_k x \leq \gamma_k \tag{13a}$$

$$y + \alpha_k x \geq \gamma_k \tag{13b}$$

where

$$\begin{aligned} \alpha_k &:= -\frac{y_k - y_{k+1}}{x_k - x_{k+1}}, \quad \gamma_k := -\frac{y_k - y_{k+1}}{x_k - x_{k+1}} x_k + y_k \quad k = 1,2,3,4,5,6,7. \\ \alpha_k &:= -\frac{y_8 - y_1}{x_8 - x_1}, \quad \gamma_k := -\frac{y_8 - y_1}{x_8 - x_1} x_8 + y_8 \quad k = 8 \end{aligned} \tag{14}$$

From equation (14) it can be seen that only γ_k is dependent of the magnitude of V_{MAX}. By multiplying γ_k with V_{MAX}, while x := $V_d(t)$ and y := $V_q(t)$, the octagonal approximation for the voltage constraints are obtained.

2.5.2 BATTERY CURRENT CONSTRAINT

In this thesis it is of interest to restrict the battery current to ensure a safe operation. The constraint is given as:

$$I_{\text{MIN}} \leq i_{\text{Battery}}(t) \leq I_{\text{MAX}} \tag{15}$$

The main challenge with this constraint is that an estimate of $i_{\text{Battery}}(t)$ is needed. One way to achieve this is by using a power relationship, namely:

$$P_{\text{Battery}}(t) = P_{\text{Motor}}(t) + P_{\text{Losses}}(t) \tag{16}$$

where $P_{\text{Losses}}(t) \geq 0$, as power is dissipated. Thus:

$$P_{\text{Battery}}(t) \geq P_{\text{Motor}}(t) \tag{17}$$

This gives a lower bound on the battery current as losses due to switching, heat etc in reality occurs. However from control perspective this might be acceptable as one can increase the bounds on equation 15 to compensate for the unmodeled losses.

By using:

$$P_{\text{Battery}}(t) = V_{\text{Battery}} i_{\text{Battery}}(t) \tag{18a}$$

$$P_{\text{Motor}}(t) = \frac{3}{2}(V_d(t) i_d(t) + V_q(t) i_q(t)) \tag{18b}$$

one have:

$$i_{\text{Battery}}(t) \approx \frac{\frac{9}{4}(V_d(t) + V_q(t))(i_d(t) + i_q(t))}{V_{\text{Battery}}} \tag{19}$$

which can be used to rewrite equation 15 as:

$$\bar{I}_{\text{MIN}} \leq i_{\text{Battery}}(t) \leq \bar{I}_{\text{MAX}} \tag{20}$$

where $\bar{I}_{\text{MIN}}$ and $\bar{I}_{\text{MAX}}$ are the compensated bounds.

Note: At closer look at equations 19 and 20, one could see that the battery current constraint can be regarded as a power constraint as well. This can be interpreted as: *The active power in the dq0-frame should not exceed the given bounds when the constraint is multiplied with* $V_{Battery}$. For simplicity, this constraint is referred to as battery current constraint in this paper.

2.5.2.1 VALIDATION OF BATTERY CURRENT MODEL BY SIMULATIONS

To validate whether or not equation 19 gives a good representation of the the battery current, simulations have been carried. These simulations compared one model, where DC-AC conversion was used, versus the PMSM model with the battery current model. The main difference between the two models is that first measure the DC current from the voltage source by using a scope, see «I_batt [A]» in Figure 9, while the other calculates it. Thus, the simulations can be easily be used to verify the accuracy of the battery current model. Note that the switching effects are not considered. This is realized by using a average model for the DC-AC conversion in the first model.

Two scenarios are shown below. In the first scenario, a step on both $V_d(t)$ and $V_q(t)$ are given. Both steps occur at 100[ms], $V_d(t)$ has a step of 1[V] and $V_q(t)$ has a step of 2[V]. The battery current is shown for three different sampling values on the powergui-block, see Figure 13. The powergui, see Figure 9, controls the sampling rate the electric power circuit is sampled with. It is seen in the figure below that a lower sample time on the powergui gives a better fit, although this is hard to see. The reason for this is that the PMSM model presented in this chapter has fast dynamics and is modeled as a continuous model. As previously mentioned, the electric circuit should be sampled with a sufficiently low sampling time as it is in reality a part of the hardware (continuous). In the second scenario the effects of the three sampling values are better seen. Here, a testcase for the given application is shown. This scenario is better understood after having read the next chapters. For now, it is sufficient to know that the result is obtained by following an angular position request.

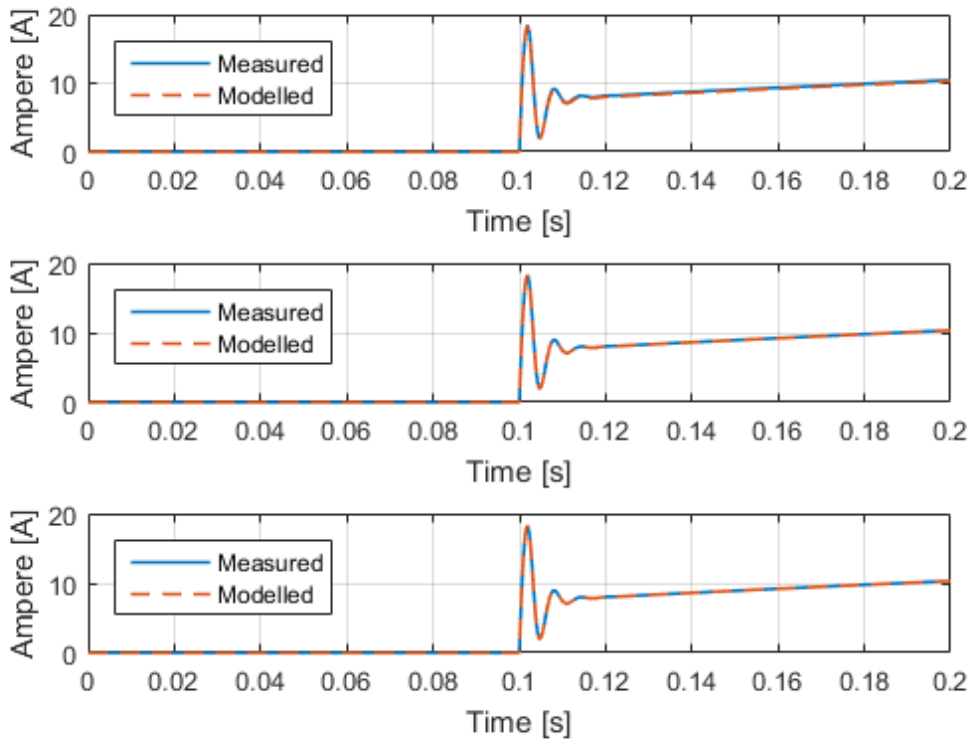

Figure 13.: Battery Current Scenario 1: Measured Vs Modelled. The top shows the measured signal with a sample time Ts = 10[μs], the middle with Ts = 1[μs] and the bottom with Ts = 0.1[μs] for the powergui.

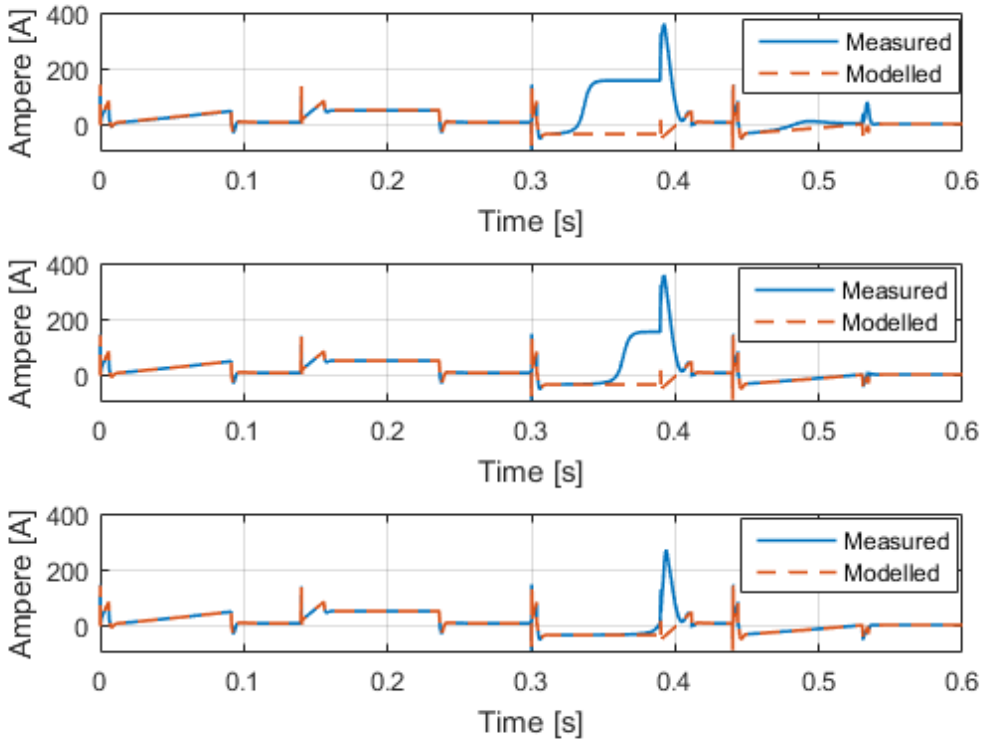

Figure 14.: Battery Current Scenario 2: Measured Vs Modelled. The top shows the measured signal with a sample time Ts = 10[μs], the middle with Ts = 1[μs] and the bottom with Ts = 0.1[μs] for the powergui.

It can be seen from both figures that the battery current model gives a good approximation when a sufficiently small sample time on the powergui-block is used. The model is therefore assumed acceptable for portraying the behaviour of the battery current.

2.5.3 ANGULAR VELOCITY CONSTRAINT

The constraint on the angular velocity is given as:

$$\omega_{m_{\text{MIN}}} \leq \omega_m(t) \leq \omega_{m_{\text{MAX}}} \tag{21}$$

where the bounds are known. As this is a linear constraint it cannot be more simplified.

2.5.4 TORQUE/CURRENT IN Q-DIRECTION CONSTRAINT

As the torque, $T_e(t)$ and $i_q(t)$ has a linear relationship, see equation 6e, this restriction can be regarded as either a torque constraint or a constraint on the current in q-direction. Thus the constraint can be either written as:

$$T_{e_{\text{MIN}}} \leq T_e(t) \leq T_{e_{\text{MAX}}} \tag{22}$$

or

$$i_{q_{\text{MIN}}} \leq i_q(t) \leq i_{q_{\text{MAX}}} \tag{23}$$

where the both bounds are known. As these are linear constraints they cannot be more simplified.

3

NONLINEAR MODEL PREDICTIVE CONTROL (NMPC)

This chapter presents some theory about what a optimization problem is and some important aspects on what one should consider when solving a such problem. This is presented with consideration on a Nonlinear Model Predictive Control (NMPC).

3.1 INTRODUCTION

Nonlinear programming is the process of solving an optimization problem where some of the constraints or the objective function are nonlinear [16]. The optimization problem is defined by a system of equalities and inequalities, over a set of unknown real variables, that are maximized or minimized with help of an objective function. According to [17] the general form of a nonlinear programming problem is given as a minimization problem with a scalar-valued function f, of several variables x, subject to some constraints. In mathematical terms:

$$\text{minimize } f(x) \tag{24a}$$

$$\text{such that} \quad c_i(x) \leq 0, \quad \forall i \epsilon I \tag{24b}$$

$$c_i(x) = 0, \quad \forall i \epsilon \mathscr{E} \tag{24c}$$

where each $c_i(x)$ is a mapping from R^n to R, and $\mathscr{E}$ and I are index sets for equality and inequality constraints respectively.

Nonlinear Model Predictive Control (NMPC), as presented in [10], is a subset of nonlinear programming problems in which a quadratic objective function is commonly used. In addition the NMPCs utilizes a prediction horizon to forecast the behavior of the dynamical model N steps ahead. In this prediction the control structure minimizes the objective function while trying to find a feasible solution. With a feasible solution it is meant; the set of all possible points (sets of values of the choice variables) of an optimization problem that satisfies the given constraints. Figure 15 shows an illustration of the prediction horizon, where the length of the horizon is given by p

instead of N.

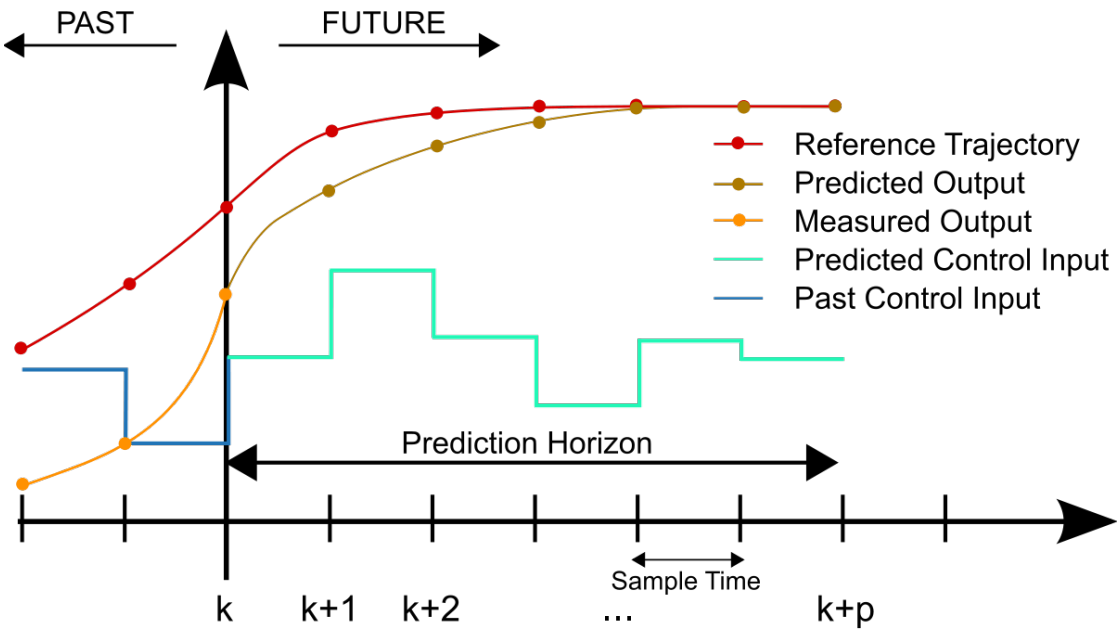

Figure 15.: Prediction Horizon (Photo [18])

From the figure it is seen that the control structure utilizes the present states and previous inputs to find an optimal sequence of future inputs, which results in the predicted output to reach the reference. The first set of inputs are used and the prediction for the new N steps ahead is repeated at the next time step.

3.2 Problem formulation

A NMPC problem can be formulated as:

$$\min_{\substack{x_0,\dots,x_N \\ u_0,\dots,u_{N-1}}} J = \min_{\substack{x_0,\dots,x_N \\ u_0,\dots,u_{N-1}}} ||(\mathbf{P}_{\vec{y}}(\vec{y}_{k+N} - \vec{r}_k)||^2 + \sum_{i=0}^{N-1} (||(\mathbf{Q}_{\vec{y}}(\vec{y}_{k+i} - \vec{r}_k)||^2 + ||\mathbf{R}_{\Delta\vec{u}}\Delta\vec{u}_{k+i}||^2) \tag{25a}$$

$$s.t \quad \vec{x}_{k+i+1} = f(\vec{x}_{k+i}, \vec{u}_{k+i}) \tag{25b}$$

$$\vec{y}_{k+i} = g(\vec{x}_{k+i}, \vec{u}_{k+i}) \tag{25c}$$

$$\vec{u}^{\text{lo}}_{k+i} \leq \vec{u}_{k+i} \leq \vec{u}^{\text{up}}_{k+i} \tag{25d}$$

$$\vec{x}^{\text{lo}}_{k+i} \leq \vec{x}_{k+i} \leq \vec{x}^{\text{up}}_{k+i} \tag{25e}$$

$$\vec{r}^{\text{lo}}_{k+i} \leq r(\vec{x}_{k+i}, \vec{u}_{k+i}) \leq \vec{r}^{\text{up}}_{k+i} \tag{25f}$$

$$\text{with } \vec{x}_k \text{ and} \vec{u}_{k-1} \text{ given and } i = 0, 1, \dots N-1 \tag{25g}$$

where J is the objective function and **P**, **Q**, **R** are weight matrices. Further, f and g are functions describing the dynamical model and the relations to the output $\vec{y}$. Equations 25d-25f describes the constraints on the inputs, states, and nonlinear constraints that may consist of both input and states.

Other constraints, or additional penalties, are also possible to add. For instance one could include a rate change on some of the states as a constraint or as a penalty.

Notice that the objective function is minimized with respect to both states and inputs. This is known as «Implicit prediction form» [19]. Although minimizing with respect to both states and inputs yields a larger optimization problem, it has a lot of structure and sparsity, which typically is very well exploited by different solvers [19].

3.3 Solvers

Based on how the optimization problem is formulated and the given application, different types of solver may be better suited. For nonlinear optimization problems these are usually found by try-and-error approach. The reason for this is that a nonlinear optimization problem has sub-optimal solutions, which may or may not be acceptable for studied application. Thus to ensure that the solver used provides satisfactory results, different scenarios of the application have to be run.

In this paper the solvers FMINCON and IPOPT have been used. Both of these are suited for nonlinear optimization problems. However IPOPT is commonly used in large-scale optimization problems [20]. It is also supported by ACADO, which is a framework for solving optimization problems with c-code generation support. In this

paper another framework, Yalmip, has rather been used. This is a well documented framework for solving optimization problems with a lot of different functions to ensure good flexibility for the user. However it should be noted that the framework is not suited for c-code generation yet. Using a framework called ACADO would therefore have been a better choice. Due to lack of time and that using Yalmip with MATLAB is better documented, Yalmip was used in this paper.

The reader is made aware that other solvers might be more optimal for this application. Finding the most optimal solver has not been the scope of this paper, as other test cases similar to the actual application have to be carried. One test case alone is not sufficient. A list of different solvers can be found in [21]. These are supported by the framework used in this paper.

4

APPLICATION

The following chapter gives the required information about the given application. In addition different ways of placing the controllers are presented here.

4.1 LAYOUT

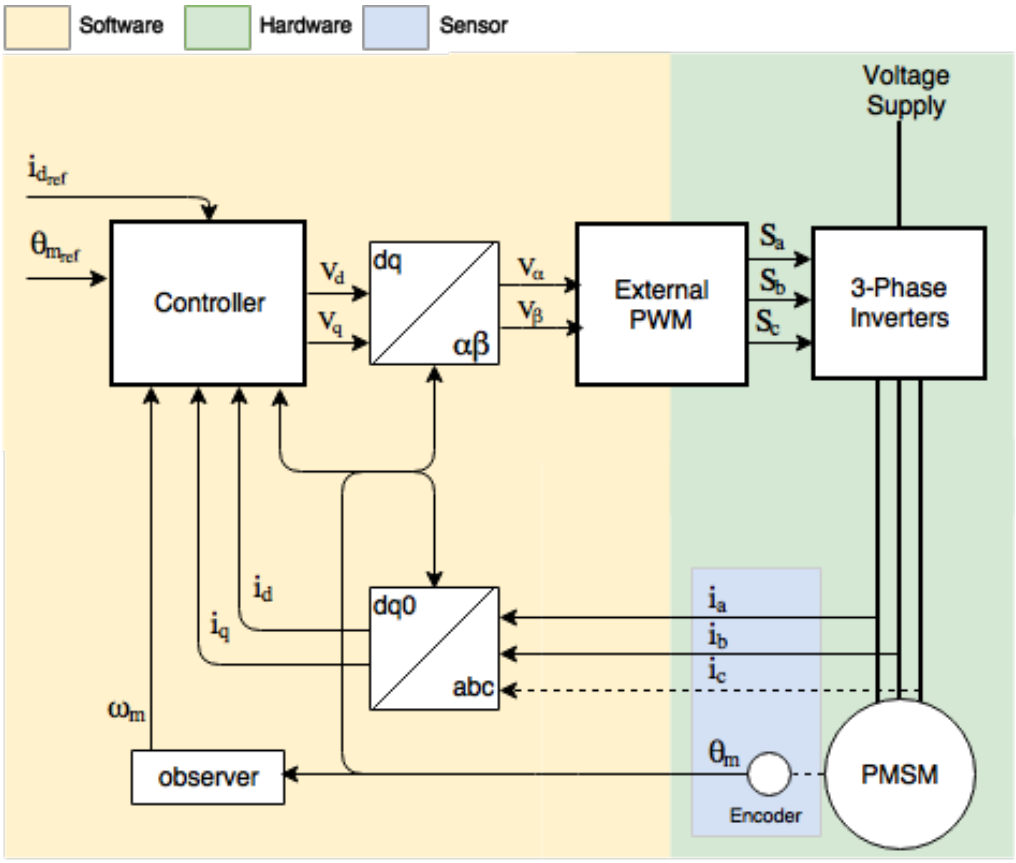

Figure 16.: General layout for control of the given application.

Due to confidentiality, information about what the motor is used to control is omitted from this paper. However this information is not crucial to understand the given application and the further analysis.

In the given application the PMSM is controlled to follow a position reference, $\theta_{m_{\text{ref}}}$, while $i_d(t)$ is controlled to $i_{d_{\text{ref}}}$. To achieve this, a layout as illustrated in Figure 16 is used. A brief run-through of the flow is; a request in angular position gives desired voltages in the dq0-frame. These voltages are transformed to reference signals the PWM tries to achieve in the abc-frame. This is done by controlling the ON-OFF time of the transistors. The currents along with the angular position is measured and sent back to "Controller". The block "Controller" consists here of a cascade of controllers to obtain a robust control, see Figure 17. Here, with robust control it meant that all of the states in the PMSM model are controlled.

This is a well used approach for applications where motors such as PMSMs are used. As stated in section 2.2, the transformations in Figure 16 enables the user to obtain a simpler system to work with when designing the controllers.

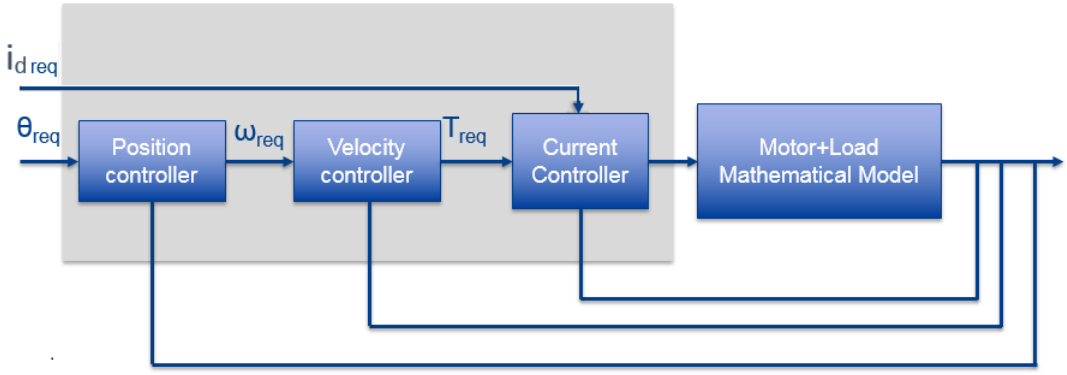

Figure 17.: Common cascade for block "Controller"

As one will see later in this chapter, there are some varieties of where to put the predictive controller in Figure 17. These are presented after having presented the given test case.

4.1.1 SENSORS

The available measurements for this application can also be seen from Figure 16. These are in form of the mechanical angular position and the two of the three-phase currents. The third current is obtained by using Kirchhoff's Current Law (KCL) that states; "the sum of currents flowing into that node is equal to the sum of currents flowing out of that node". For a balanced three-phase system this gives:

$$i_c(t) = -i_a(t) - i_b(t) \tag{26}$$

The angular velocity is estimated. This can be done in several manners such as by using a Kalman filter or by taking the derivative. Note that the latter approach is not recommended when dealing with noisy signals. However for simulation purposes, and to avoid unnecessary sources of error, it is assumed that the velocity is perfectly known.

4.2 TEST CASE

A test case has been given for this application. It is of interest to follow four steps in the angular position as fast as possible, see Figure 18. This gives a high angular velocity demand, which again gives a high torque request to the current controller to follow. Recall from section 2.4.1 that the torque, $T_e(t)$, is proportional to the current in q-direction, $i_q(t)$. It is therefore of interest to control $i_d(t)$ to 0, as a nonzero value has no directly contribution to the rotation of the motor. This is in sense of the torque being independent of $i_d(t)$. In addition a nonzero value would lead to unnecessary increase in heat in the motor, which is undesired.

The test case is constrained by the constraints presented in section 2.5. To summarize:

Objective: Follow $\theta_{m_{\text{ref}}}(t)$ while satisfying the given constraints. $i_d(t)$ is controlled to 0 to avoid unneccessary increase in heat in the motor.

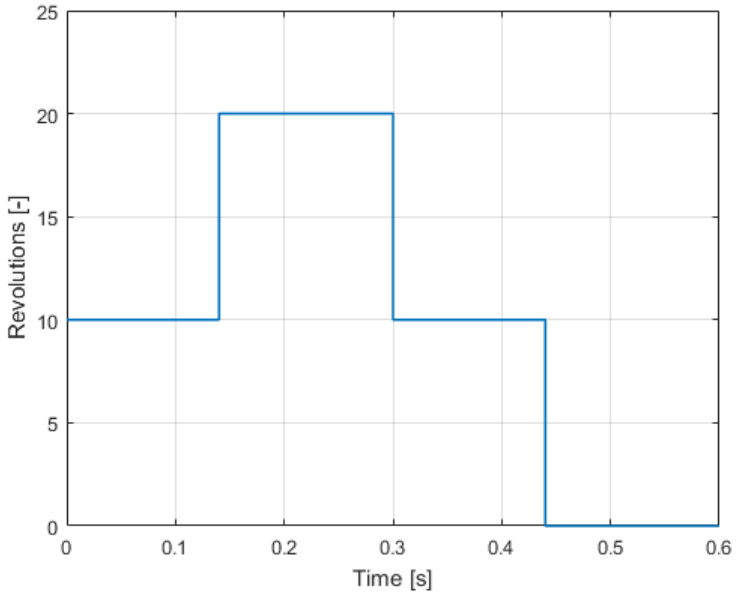

Figure 18.: Angular Position Reference (Mechanical).
In total four steps at $t = 0[s]$, $t = 0.14[s]$, $t = 0.30[s]$ and $t = 0.44[s]$ respectively.

4.2.1 Load Profile

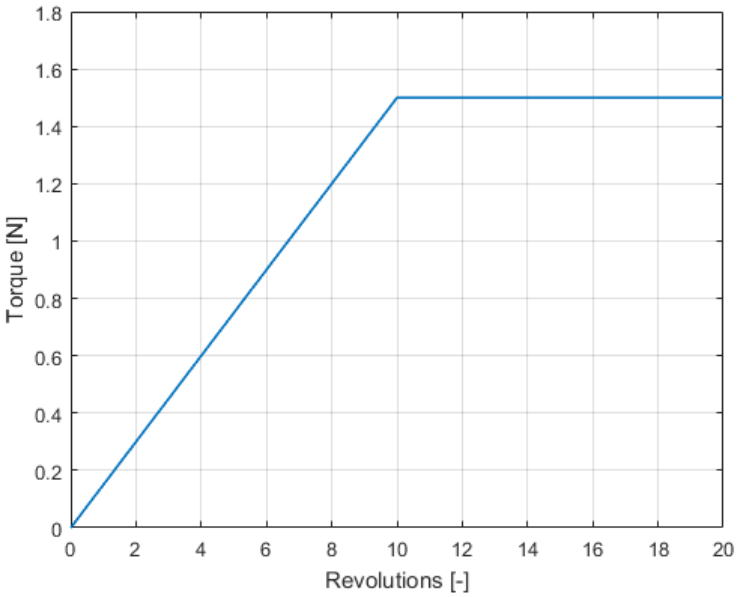

Figure 19.: Load Profile (Mechanical)

The load profile, $T_l(t)$ used in this paper is given in Figure 19. This is a function of the angular position.

4.3 Varieties of Controller Combinations

There is a variety of controller combinations that may be suited this application. For instance, one could replace the entire cascade of controllers in Figure 17 with one or more predictive controllers. Other possibilities are to replace one or two of the controllers with a predictive controller.

These varieties offers both advantages and disadvantages that are worth considering, see below. It should be noted that the remaining controller(s) in Figure 17 is implemented as a P- or PI-controller, where the P-controller is solely used in the position controller.

- **Option 1: Replacing the entire cascade with a predictive controller:**
  - Advantage:
    - No need for saturation blocks and rate limiters as these can be taken care of in the controller. This is by appropriate usage of the objective function (what to penalize, weighting etc), correct constraints and a suitable solver.
  - Disadvantage:
    - Several states to account for which increase complexity in form of weighting and tuning.
    - As the reference signals for the entire cascade is $\theta_{m_{\text{req}}}(t)$ and $i_{d_{\text{req}}}(t)$, where the position varies much slower than the current, both a small sample time and long prediction horizon is needed. A small enough sample time is needed to capture the transient response of the fastest reference signal (current), while a long enough prediction is needed to capture how the slowest reference signal (position) changes. Thus it natural to think that this would increase the computational effort compared to having the predictive controller on one or two of the controllers in Figure 17.

- **Option 2: Replacing the current controllers with a predictive controller:**
  - Advantage:
    - The reference signals would be a torque and a current reference. As the torque reference, $T_{e_{\text{req}}}(t)$, is proportional with the current reference in q-direction, $i_{q_{\text{req}}}(t)$, a smaller prediction horizon can be chosen. This is due to $i_q(t)$ having faster dynamics than $\theta_m(t)$.
  - Disadvantage:
    - The dynamics, or an estimate, of how $\omega_m(t)$ changes must be included in the controller. This is if and only if the variation in the velocity has a big influence on the current dynamics in the prediction horizon. This is important to consider in order to ensure that the predicted values, from the controller, are close to the real response.
    - Nonlinear optimization problem due to the battery current constraint.
- **Option 3: Replacing the position and speed controller with a predictive controller:**
  - Advantage:
    - In this case the only reference signal for the predictive controller to follow would be the position reference, $\theta_{m_{\text{req}}}(t)$. Thus, a higher sample time could be used compared to the two options above. As a higher sample time results in a longer time horizon where the controller predicts, the prediction horizon could also be reduced as well. See Figure 15 while keeping the prediction horizon, p, constant with an increased sample time. One should than see that the time horizon where the controller predicts becomes bigger.
    - If correctly tuned, one would expect the controller to give a better and more optimal torque request for the current controller to follow. The restrictions on the torque and velocity should be included here instead of using saturation and rate limiters.
    - As this control structure does not consider the voltage and battery current constraint, a linear MPC can be used.

- Disadvantage:
  - No assurance given by the predictive controller that the generated input, $T_{e_{\text{ref}}}(t)$, would not cause the "spikes" in the battery current. See «Modelled» in Figure 14 in section 2.5.2.1 where the battery current model was validated. There, one can see that the spikes occur at $t = 300$[ms] and $t = 440$[ms].

Other options exists as well. For instance it could be of interest to combine Option 2 and Option 3.

In this paper the focus has been on a control structure where a predictive controller has replaced both of the current controllers, namely option 2. This is motivated by the highlighted control structures that were briefly introduced in section 1.1. This is especially with weight on the control structure Model Predictive Torque Control (MP-TC) presented in [7]. The reason for this is that the author of this thesis found the assumption of a constant velocity in the prediction horizon of interest. This is an assumption that have shown good results for the studied application in [7]. This would, for the predictive controller, reduce the number of states in the model from 3 to 2 states, see chapter6. This could, from control perspective, result in less computational effort if the assumption holds here as well.

One of the drawbacks of [7] is that they have linearized the system matrices around a nominal value for the velocity, which is currently unknown for the application given here. As the given application by Kongsberg Automotive AS results in a large range of allowable values for the velocity, the system matrices found in [7] have to be linearized around new points during the operation. This is essential in order to have a sufficient model that is used by the predictive controller. If the linearizations have to occur more often than what is acceptable, a dynamical of the velocity could be of interest to include. This should result in less demand for linearization of system matrices. The latter approach is inspired by the Discrete Model Prective Control (DMPC) [9], which had placed the predictive controller on the both the speed and current controllers.

Note: As the used framework does not support c-code generation, the computational effort for deploying the controllers on an embedded target cannot be quantified.

Part II

CONTROLLERS

5

FIELD ORIENTED CONTROL (FOC) WITH POSITON CONTROL

This chapter presents the «traditional controller» that is compared with the control structure(s) presented in the next chapter. The traditional controller does not limit the battery current. A modified version of the controller is presented here as well. This limits the battery current.

5.1 SCHEMATIC

Figure 20 shows a schematic of the «traditional controller» used in this paper. This methodology is a combination of Field Oriented Control (FOC) and a position controller.

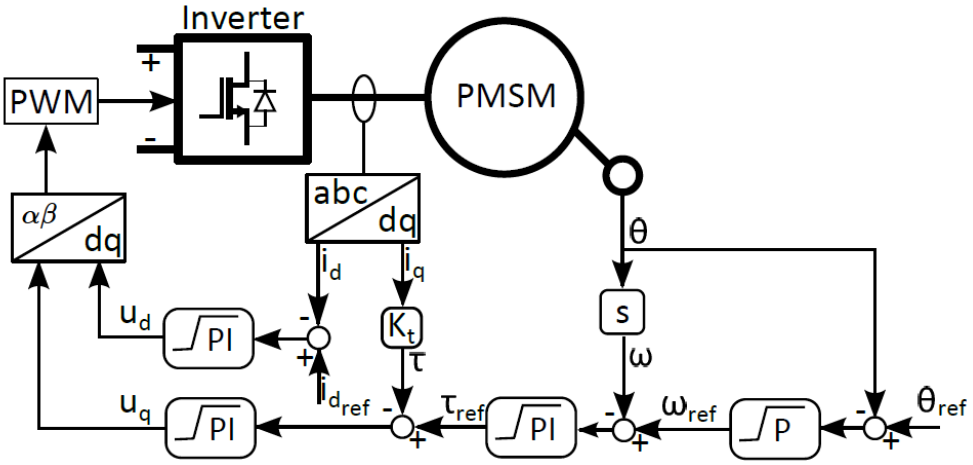

Figure 20.: Drive control system with FOC + Position Control (Modified version of the figure found in [7])

Note: In Figure 20 K_t is defined as the relation between $i_q(t)$ and $T_e(t)$, namely:

$$K_t := \frac{3}{2} Z_p \phi_{mg} \tag{27}$$

5.2 Anti-Windup and Usage of saturation blocks

As one will see in the next sections, the current controllers and the speed controller are implemented by using both an anti-windup method and saturation blocks. This is due to the integrator in the PI-controller for each of them. As the position controller consists of a P-controller an anti-windup method is not needed there. In this paper the anti-windup method «back-calculation» is used. See Figure 21 where an illustration with a PID-controller and a process G(s) is given.

In this paper T_t is set equal to T_i, which is the time constant for the integral gain.

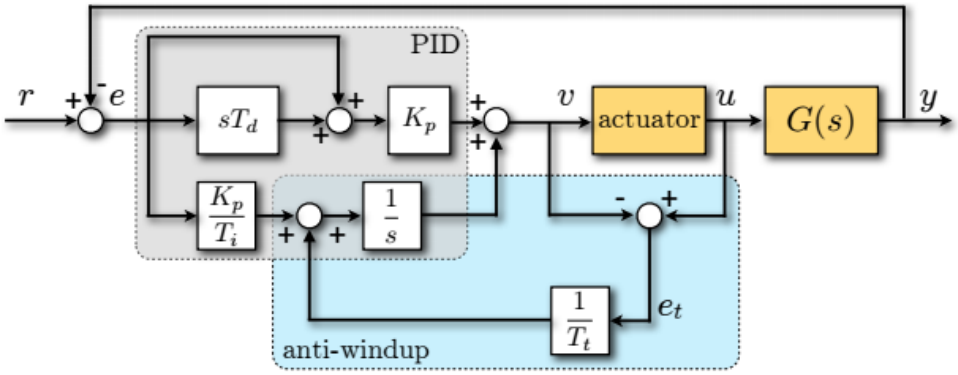

Figure 21.: Anti-windup: Back-Calculation (Photo [22])

5.3 Current Controllers

The current controllers are designed by looking at the dynamics for the currents. For simplicity, equations 6a and 6b are rewritten below.

$$\frac{di_d(t)}{dt} = \frac{1}{L_d}(V_d(t) - R_s i_d(t) + Z_p \omega_m(t) L_q i_q(t)) \tag{28a}$$

$$\frac{di_q(t)}{dt} = \frac{1}{L_q}(V_q(t) - R_s i_q(t) - Z_p \omega_m(t) L_d i_d(t) - Z_p \omega_m(t) \phi_{mg}) \tag{28b}$$

Due to the bilinear terms in the equations it is difficult to design a PI controller without doing some sort of modification or a trick. By using an approach similar to Feedback Linearization the terms with ω_m can be compensated for. This gives two first order transfer functions between the currents and the voltages. Notice that this is not Feedback Linearization in its true sense as a linear term with ω_m is also removed [23]. The new equations are:

$$\frac{di_d(t)}{dt} = \frac{1}{L_d}(\hat{V}_d(t) - R_s i_d(t)) \tag{29a}$$

$$\frac{di_q(t)}{dt} = \frac{1}{L_q}(\hat{V}_q(t) - R_s i_q(t)) \tag{29b}$$

where

$$V_d(t) := \frac{1}{L_d}(\hat{V}_d(t) - Z_p \omega_m(t) L_q i_q(t)) \tag{30a}$$

$$V_q(t) := \frac{1}{L_q}(\hat{V}_q(t) + Z_p \omega_m(t) L_d i_d(t) + Z_p \omega_m(t) \phi_{mg}) \tag{30b}$$

The transfer for the linear systems are:

$$\frac{i_d(s)}{\hat{V}_d(s)} = \frac{\frac{1}{L_d}}{s + \frac{R_s}{Ld}} \tag{31a}$$

$$\frac{i_q(s)}{\hat{V}_q(s)} = \frac{\frac{1}{L_q}}{s + \frac{R_s}{Lq}} \tag{31b}$$

The PI-controller has been tuned by looking at the closed loop step response for the systems. The gains were found by ensuring that the step response satisfied the closed loop settling time and overshoot requirements given by Kongsberg Automotive AS. The tuning was done by using the app "Control System Designer" in MATLAB.

A saturation on voltages was set on the traditional controller. This was done by using the voltage constraint described in equation 8. The procedure satisfying the voltage constraint is given below.

Algorithm 1 Satisfying the voltage constraint

```
function VOLTAGE SATURATION
    if generated voltages is outside the circle then
        Normalize the voltages to the border of the circle
    else
        Do nothing
    end if
end function
```

For the modified controller, which in additional satisfies the constraint on the battery current, the procedure is defined as below. Note that equations 19 and 20 are used with equation equation 8.

Algorithm 2 Satisfying the voltage and the battery current constraint

function VOLTAGE AND BATTERY CURRENT SATURATION
  if *generated voltages is outside the circle* **then**
    Normalize the voltages to the border of the circle
  else
    Do nothing
  end if

  if *generated voltages along with the currents in dq0-plane does not satisfy the bounds on the battery current* **then**
    Normalize the new voltages such that the battery current is at the border
  else
    Do nothing
  end if
end function

5.4 SPEED CONTROLLER

The speed controller was designed by neglecting the influence of the torque load. In other words, after rewriting:

$$\frac{d\omega_m(t)}{dt} = \frac{1}{J_m}(T_e(t) - B_v\omega_m(t) - T_L(t)) \tag{32}$$

as

$$\frac{d\omega_m(t)}{dt} \approx \frac{1}{J_m}(T_e(t) - B_v\omega_m(t)) \tag{33}$$

The transfer function for the system was found as:

$$\frac{\omega_m(s)}{T_e(s)} = \frac{\frac{1}{J_m}}{s + \frac{B_v}{J_m}} \tag{34}$$

A PI-controller has been tuned by looking at the closed loop step response for the systems. The gains were found by ensuring that the step response satisfied the closed loop settling time and overshoot requirements given by Kongsberg Automotive AS. The tuning was done by using the app "Control System Designer" in MATLAB.

The saturation used here was on the torque constraint, see equation 22. As the constraint is linear, this was simply implemented by using a saturation block in Simulink.

5.5 Position Controller

The position controller was tuned by a design requirement that said: *The motor should at a specified position have a specified velocity*. Thus, the proportional gain was found by:

$$\text{Proportional gain} = \frac{\omega_m(t = t^*)}{\theta_m(t = t^*)} \tag{35}$$

where $\omega_m(t = t^*)$ and $\theta_m(t = t^*)$ are known velocity and position values.

The saturation used here was on the velocity constraint, see equation 21.

5.6 Traditional VS Modified Controller

The main difference between the traditional and the modified controller is that the prior is implemented with algorithm 1 while the latter is with algorithm 2. The reason for doing such is that the traditional controller is used to show a control structure that violates the battery current constraint. This led the motivation of this thesis, as the battery current constraint and the other restrictions were to be satisfied. The modified controller, as seen in results, ensures that.

6

PREDICTIVE CONTROLLER INSTEAD OF CURRENT CONTROLLERS

This chapter presents a control structure where the current controllers in the previous chapter are replaced with a predictive controller. Here a NMPC has been used. Two versions of the predictive controller are presented, where one assume constant velocity in the prediction horizon while the other does not.

6.1 INTRODUCTION

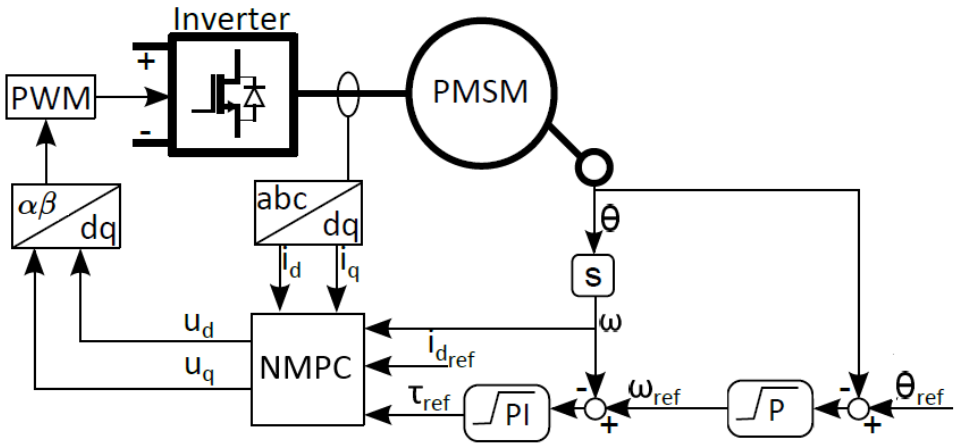

Figure 22.: Drive control system with Predictive Controller (Modified version of the figure found in [7])

As mentioned in section 4.3 the focus in this paper is on a predictive controller that replaced the current controllers. Due to the nonlinear constraint caused by the battery current a NMPC has been looked at. A shecmatic of the drive control system can be seen in Figure 22. Two versions of the predictive controller are presented in the

next sections, where one assumes constant velocity in the prediction horizon while the other does not. Both of them are supposed to follow a torque request while controlling the current in d-direction towards zero. Remember that the torque, $T_e(t)$ is proportional with the current in q-direction, $i_q(t)$. The controllers for the outer loops are implemented as described in chapter 5.

6.2 CONTROLLER WITH 2 STATES

This version of the predictive controller consider the angular velocity to be constant in the prediction horizon. The benefit of doing such is that the predictive controller then considers a model plant with two states and two inputs. The linearized system, with assuming $\omega_m(t)$ constant, can be written as:

$$\frac{d\vec{x}(t)}{dt} = \mathbf{A}_c\vec{x}(t) + \mathbf{B}_c\vec{u}(t) + \mathbf{G}_c\mu_c^0 \tag{36a}$$

$$\vec{y}(t) = \mathbf{C}_c\vec{x}(t) \tag{36b}$$

with $\vec{x}(t) = [i_d(t) \quad i_q(t)]^T$, $\vec{u}(t) = [V_d(t) \quad V_q(t)]^T$, $\vec{y}(t) = [i_d(t) \quad T_e(t)]^T$, $\mu_c^0 = \omega_m^0$ and the matrices $\mathbf{A}_c$, $\mathbf{B}_c$, $\mathbf{C}_c$ and $\mathbf{G}_c$ as

$$\mathbf{A}_c = \begin{bmatrix} -\frac{R_s}{L_d} & \frac{L_q}{L_d}Z_p\omega_m^0 \\ -\frac{L_d}{L_q}Z_p\omega_m^0 & -\frac{R_s}{L_q} \end{bmatrix} \tag{37a}$$

$$\mathbf{B}_c = \begin{bmatrix} \frac{1}{L_d} & 0 \\ 0 & \frac{1}{L_q} \end{bmatrix} \tag{37b}$$

$$\mathbf{C}_c = \begin{bmatrix} 1 & 0 \\ 0 & K_t \end{bmatrix} \tag{37c}$$

$$\mathbf{G}_c = \begin{bmatrix} 0 \\ -\frac{\phi_{mg}}{L_q} \end{bmatrix} \tag{37d}$$

where $\mu_c^0 = \omega_m^0$ is the linearized point and K_t is given by equation 27.

By discretizing the system, one could use the following optimization problem

$$\min_{\substack{x_0,\dots,x_N \\ u_0,\dots,u_{N-1}}} J = \min_{\substack{x_0,\dots,x_N \\ u_0,\dots,u_{N-1}}} ||(\mathbf{P}_{\vec{y}}(\vec{y}_{k+N} - \vec{r}_k)||^2 + \sum_{i=0}^{N-1} (||(\mathbf{Q}_{\vec{y}}(\vec{y}_{k+i} - \vec{r}_k)||^2 + ||\mathbf{R}_{\Delta\vec{u}}\Delta\vec{u}_{k+i}||^2) \tag{38a}$$

$$s.t \quad \vec{x}_{k+i+1} = \mathbf{A}\vec{x}_{k+i} + \mathbf{B}\vec{u}_{k+i} + \mathbf{G}\mu_k \tag{38b}$$

$$\vec{y}_{k+i} = \mathbf{C}\vec{x}_{k+i} \tag{38c}$$

$$\mathbf{A}_{\vec{u}_{k+i}}\vec{u}_{k+i} \leq \mathrm{b}_{\vec{u}_{k+i}} \tag{38d}$$

$$\mathrm{b}^{\mathrm{lo}}_{g(\vec{x}_{k+i},\vec{u}_{k+i})} \leq g(\vec{x}_{k+i},\vec{u}_{k+i}) \leq \mathrm{b}^{\mathrm{up}}_{g(\vec{x}_{k+i},\vec{u}_{k+i})} \tag{38e}$$

$$\text{with } \vec{x}_k \text{ and} \vec{u}_{k-1} \text{ given and } i = 0, 1, \dots N-1 \tag{38f}$$

where equation 38d is the octagonal approximation of the voltage constraint (equation 13a-13b) and equation 38e is the battery current restriction (equation 20). Notice that k is the present step at which the predictive controller is called at while i is the steps in the prediction horizon. μ_k is the angular velocity at $t = k$. In addtion note that **A**, **B**, **C** and **G** are the discretized system matrices.

The opitmization problem does not show how many times the linearization of the $i_d(t)$- and $i_q(t)$-dynamics happens. However, as mentioned earlier, the model has to be linearized and discretized a sufficient amount of times such that the controller utilize an accurate model. This can for instance be at every fourth, tenth, 100th, and so on, call to the predictive controller.

The format of the optimization problem is supported by a framework named Yalmip.

6.3 CONTROLLER WITH 3 STATES

This version of the predictive controller consider the angular velocity to be varying in the prediction horizon. The benefit of doing such is that the predictive controller then a more accurate representation of the current dynamics. The system then needs to considers a system with three states and two inputs. As in the previous section a linear model has been used. By linearization around i_d^0, i_q^0 and ω_m^0, the system can be written as:

$$\frac{d\vec{x}(t)}{dt} = \mathbf{A}_c\vec{x}(t) + \mathbf{B}_c\vec{u}(t) + \mu_c^0 \tag{39a}$$

$$\vec{y}(t) = \mathbf{C}_c\vec{x}(t) \tag{39b}$$

with $\vec{x}(t) = [i_d(t) \quad i_q(t)]^T$, $\vec{u}(t) = [V_d(t) \quad V_q(t)]^T$, $\vec{y}(t) = [i_d(t) \quad T_e(t)]^T$ and the matrices $\mathbf{A}_c$, $\mathbf{B}_c$, $\mathbf{C}_c$ and μ_c^0 as

$$\mathbf{A}_c = \begin{bmatrix} -\frac{R_s}{L_d} & \frac{L_q}{L_d}Z_p\omega_m^0 & \frac{L_q}{L_d}i_q^0 \\ -\frac{L_d}{L_q}Z_p\omega_m^0 & -\frac{R_s}{L_q} & -(\frac{L_d}{L_q}i_d^0 + \frac{\phi_{mg}}{L_q}) \\ 0 & \frac{K_t}{J_m} & -\frac{B_v}{J_m} \end{bmatrix} \tag{40a}$$

$$\mathbf{B}_c = \begin{bmatrix} \frac{1}{L_d} & 0 \\ 0 & \frac{1}{L_q} \\ 0 & 0 \end{bmatrix} \tag{40b}$$

$$\mathbf{C}_c = \begin{bmatrix} 1 & 0 & 0 \\ 0 & K_t & 0 \end{bmatrix} \tag{40c}$$

$$\mu_c^0 = \begin{bmatrix} -\frac{L_q}{L_d}\omega_m^0 i_q^0 \\ \frac{L_d}{L_q}\omega_m^0 i_d^0 \\ -\frac{T_L}{J_m} \end{bmatrix} \tag{40d}$$

where K_t is given by equation 27.

By discretizing the system, one could use the following optimization problem

$$\min_{\substack{x_0,\dots,x_N \\ u_0,\dots,u_{N-1}}} J = \min_{\substack{x_0,\dots,x_N \\ u_0,\dots,u_{N-1}}} ||(\mathbf{P}_{\vec{y}}(\vec{y}_{k+N} - \vec{r}_k)||^2 + \sum_{i=0}^{N-1} (||(\mathbf{Q}_{\vec{y}}(\vec{y}_{k+i} - \vec{r}_k)||^2 + ||\mathbf{R}_{\Delta\vec{u}}\Delta\vec{u}_{k+i}||^2) \tag{41a}$$

$$s.t \quad \vec{x}_{k+i+1} = \mathbf{A}\vec{x}_{k+i} + \mathbf{B}\vec{u}_{k+i} + \mu_k \tag{41b}$$

$$\vec{y}_{k+i} = \mathbf{C}\vec{x}_{k+i} \tag{41c}$$

$$\mathbf{A}_{\vec{u}_{k+i}}\vec{u}_{k+i} \leq \mathrm{b}_{\vec{u}_{k+i}} \tag{41d}$$

$$\mathbf{A}_{\vec{x}_{k+i}}\vec{x}_{k+i} \leq \mathrm{b}_{\vec{x}_{k+i}} \tag{41e}$$

$$\mathrm{b}^{\mathrm{lo}}_{g(\vec{x}_{k+i},\vec{u}_{k+i})} \leq g(\vec{x}_{k+i},\vec{u}_{k+i}) \leq \mathrm{b}^{\mathrm{up}}_{g(\vec{x}_{k+i},\vec{u}_{k+i})} \tag{41f}$$

$$\text{with } \vec{x}_k \text{ and} \vec{u}_{k-1} \text{ given and } i = 0,1,\dots N-1 \tag{41g}$$

where equation 41d is the octagonal approximation of the voltage constraint (equation 13a-13b), equation 41e is the constraint on the angular velocity (see equation 21) and equation 41f is the battery current restriction (equation 20). Notice that k is the present step at which the predictive controller is called at while i is the steps in the prediction horizon. μ_k is the currents, velocity and torque load at $t = k$. As the torque load is slowly varying, a constant torque load in the prediction horizon is a valid assumption.

The opitmization problem does not show how many times the linearization of the $i_d(t)$- and $i_q(t)$-dynamics happens. However, as mentioned earlier, the model has to be linearized and discretized a sufficient amount of times such that the controller utilize an accurate model. This can for instance be at every fourth, tenth, 100th, and so on, call to the predictive controller. If the demand of new linearization points is significantly less than for "Controller with 2 states", this predictive controller might be more suitable for an embedded target. As told before, there are many factors that influence the computational effort. In this paper the implementation does not offer any direct answer on this.

The format of the optimization problem is supported by a framework named Yalmip.

Note: equation 41e is not part of "Controller with 2 states" as that controller does not consider a varying $\omega_m(t)$ in the prediction horizon. However, this is not the case here. Therefore, the constraint can be included here for increased robustness and to prevent having unbounded states. However as seen in results, some tuning is yet required when the constraint is included. For instance if the constraint is in the future implemented as a soft constraint, the corresponding slack variable has to added to the objective function.

Part III

RESULTS

7

TEST CASE

This chapter presents the results obtained when running the test case scenario described in section 4.2. This is shown by using the the controllers presented in chapter 5 and chapter 6. Some versions of the predictive controllers are also presented. This is done to show the influence of changing some of the parameters or adding some additional constraints in the controllers.

The results are first presented in each section with figures to illustrate how the motor behaves with respect to the different controllers. The results are thereafter summarized in tables in the end of this chapter, where the tables presents the simulation time and the overall Root Mean Squared Error (RMSE) between the signals and the references. The reader is therefore suggested to look at the respective figures for the desired controller together with the tables. The results are discussed in the next chapter.

Note: The figures obtained with IPOPT as solver are omitted from this thesis. The reason for doing so is that they gave almost identical responses to the figures where FMINCON was used. The mismatch were roughly less then 3 %.

In addition it is important to point out that the Modified Controller has been used as benchmark in one of the tables presented in this chapter. The reason for this is that the Traditional Controller does not handle the battery current constraint, while the predictive controllers are currently not implementable on an embedded target. The latter is due to the framework used in this paper.

The reader is also suggested to read the sections in a sequential order as some of the analyses of the responses are repeatable, and therefore referred to in the respective comment section.

7.1 TRADITIONAL CONTROLLER

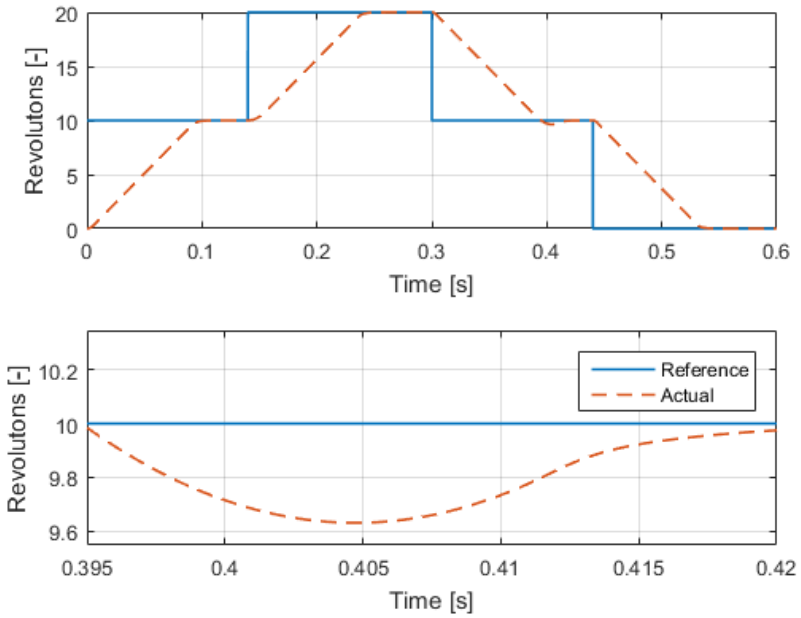

Figure 23.: Traditional Controller: Angular Position

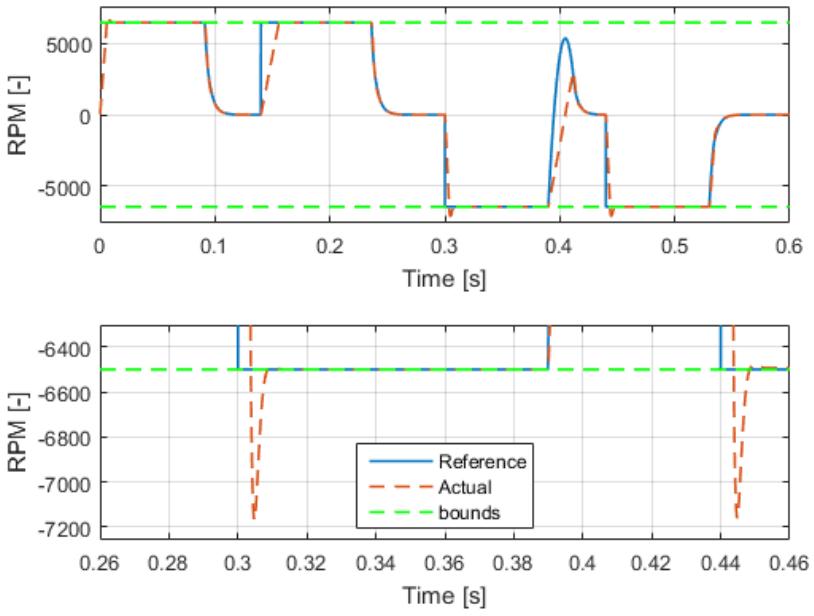

Figure 24.: Traditional Controller: Angular Velocity

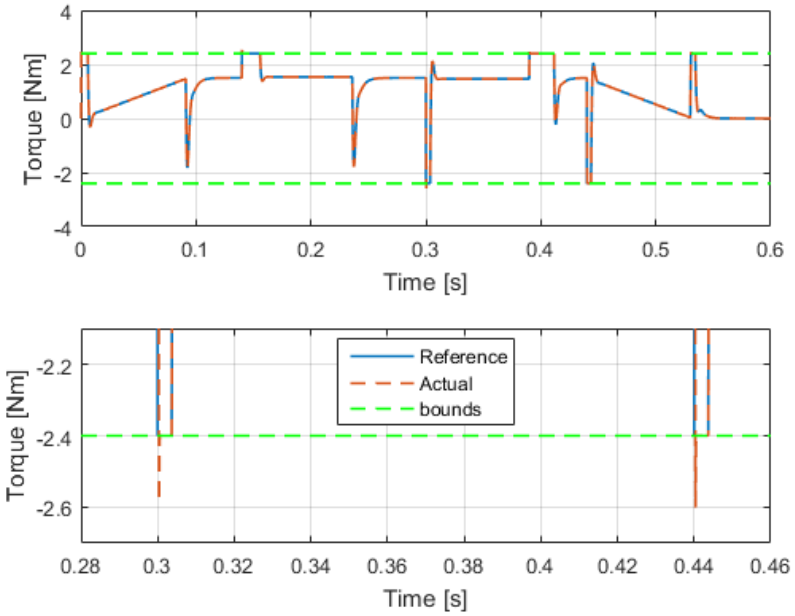

Figure 25.: Traditional Controller: Torque

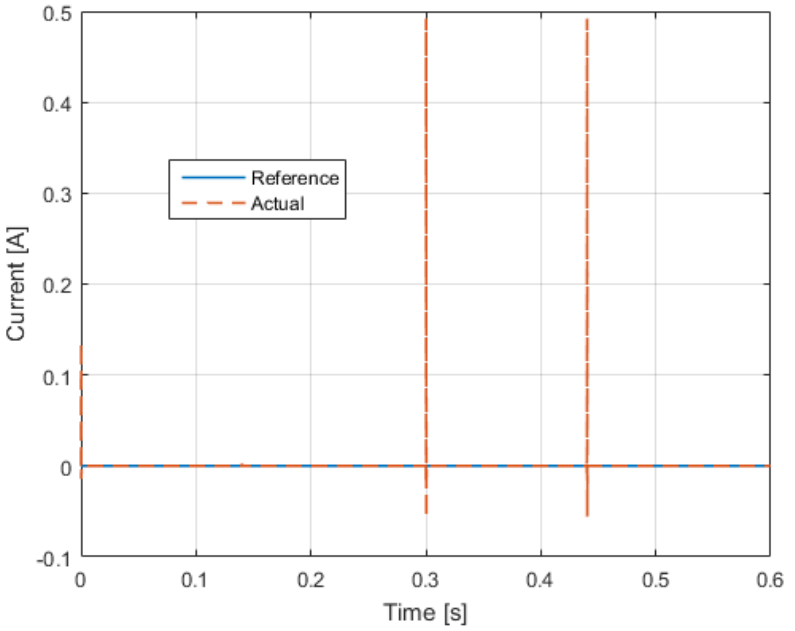

Figure 26.: Traditional Controller: Current in d-direction

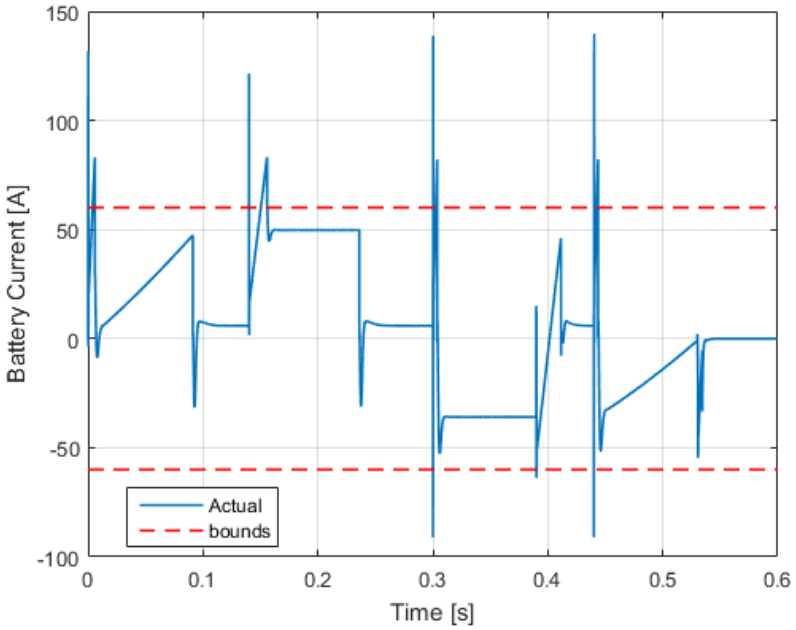

Figure 27.: Traditional Controller: Battery Current

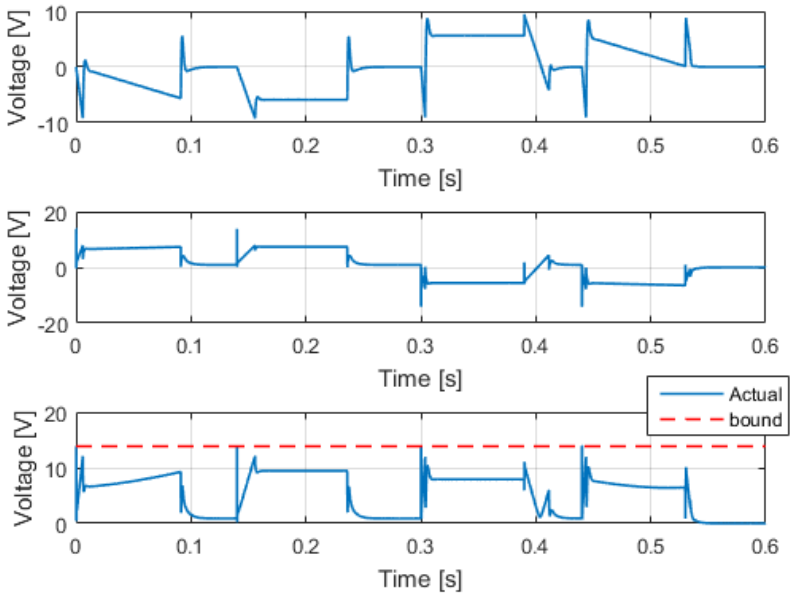

Figure 28.: Traditional Controller: Voltages. The top shows $V_d(t)$, the middle $V_q(t)$ and the bottom is the length of these two voltages.

7.1.1 COMMENTS

Overall the results for the given test case is as expected when using the Traditional Controller. One can clearly see that angular position is well followed, as the actual value reaches the reference and stabilizes at $t \approx 0.10[s]$, $t \approx 0.24[s]$ and $t \approx 0.53[s]$. See Figure 23. The overshoot that occurs at $t \approx 0.395[s]$ is due to the saturation on the torque, $T_e(t)$, which prevents the motor from utilizing the required angular velocity in that time domain. In other words, the motor does not have the required change in angular velocity available, due to saturation on the torque, to stop at the position reference.

As one can see in Figure 24 the angular velocity controller is tuned to give maximum speed to reach the angular position references as fast as possible. This is seen in terms of the steps in the reference signal for the velocity, which occurs at the same time as the steps in the position reference. The overshoots in the angular velocity are acceptable as the biggest overshoot observed here is roughly 11 %. This is well within the requirements given by Kongsberg Automotive AS.

The torque reference is very well followed. It is seen from Figure 25 the torque reaches its bounds in those intervals the angular velocity differs from its reference signal. In those intervals where the angular velocity is equal to the reference signal, the required torque to counteract the torque load and kinematic friction is provided. The overshoots in the torque are acceptable as the biggest overshoot observed here is roughly 8.5 %. This is well within the requirements given by Kongsberg Automotive AS.

From Figure 26 one can clearly see that this controller manages to keep $i_d(t)$ at its reference value, even though some spikes are observed at $t = 0[s]$, $t = 0.30[s]$ and $t = 0.44[s]$. As the $i_d(t)$ is nonzero for a very short amount of time, the influence of $i_d(t)$ in terms of energy and heat increase are negligible. This is also seen in terms of the RMSE-value for $i_d(t)$ in Table 4.

The battery current is as expected. As algorithm 1 does not handle the battery current constraint, the violations are as expected. See Figure 27.

It can from Figure 28 be seen that algorithm 1 satisfies the voltage constraint. It is also seen that the sign of $V_d(t)$ and $V_q(t)$ are opposite of the terms they want to compensate for. For instance, it is observed that the sign of $V_d(t)$ is the opposite of $Z_p\omega_m(t)L_q i_q(t)$, which is the bigger term the voltage has to compensate for to obtain $\frac{di_d(t)}{dt} \approx 0$. See equation 6a.

7.2 MODIFIED CONTROLLER

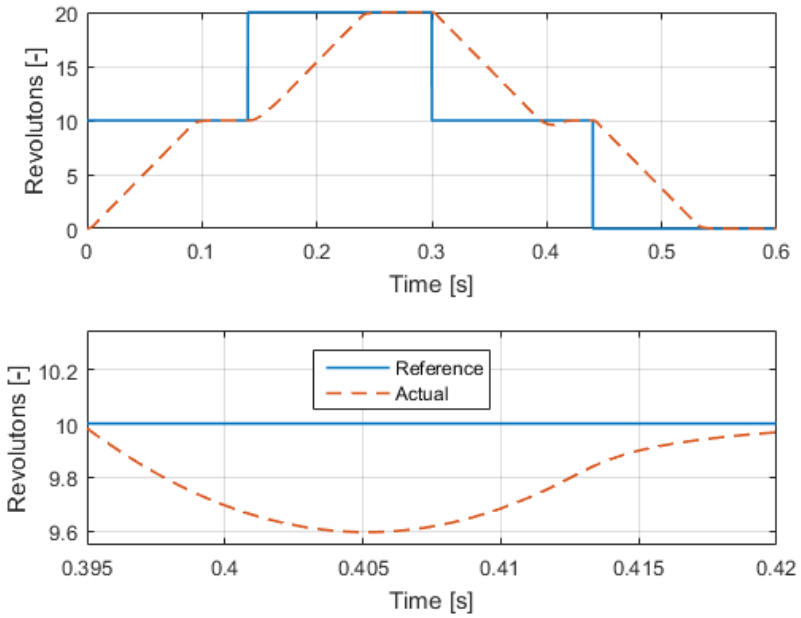

Figure 29.: Modified Controller: Angular Position

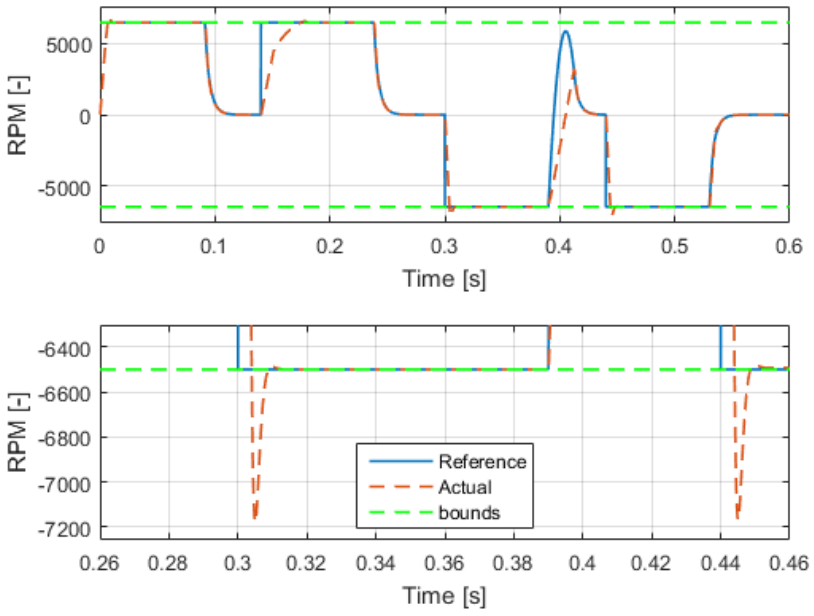

Figure 30.: Modified Controller: Angular Velocity

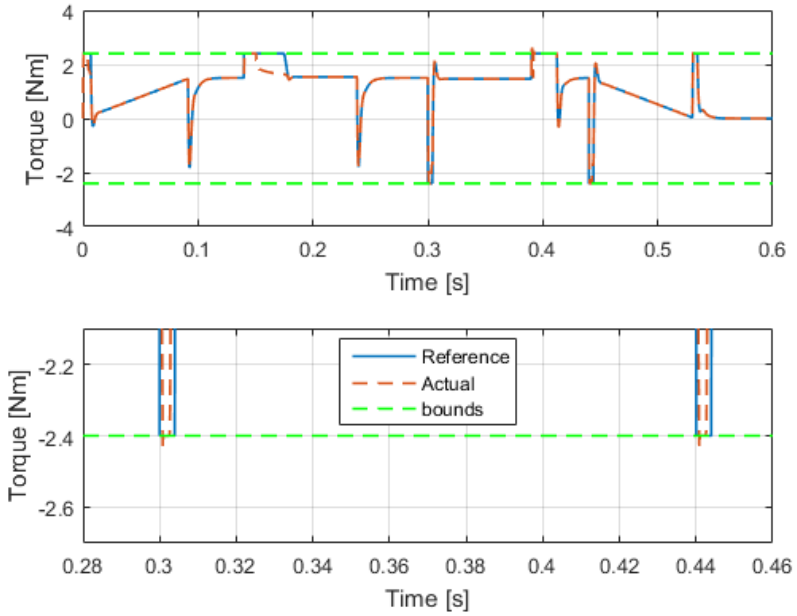

Figure 31.: Modified Controller: Torque

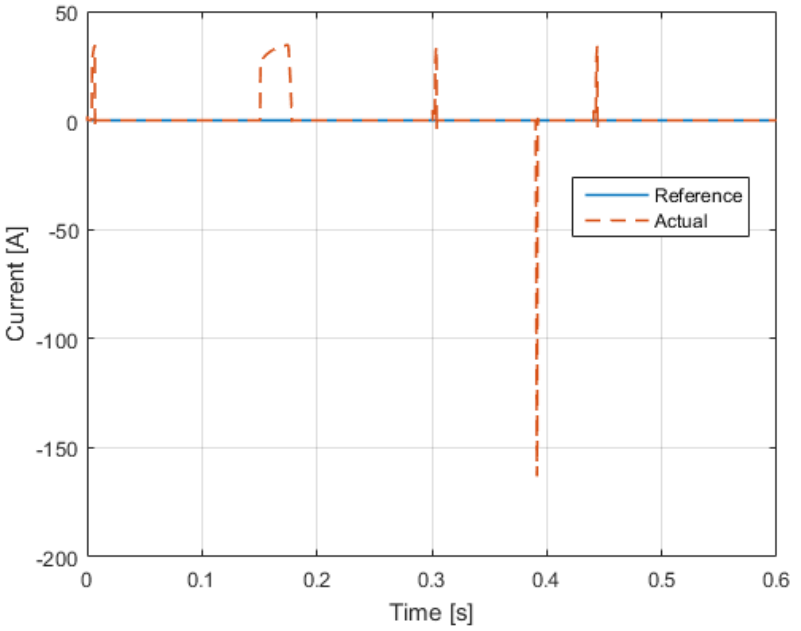

Figure 32.: Modified Controller: Current in d-direction

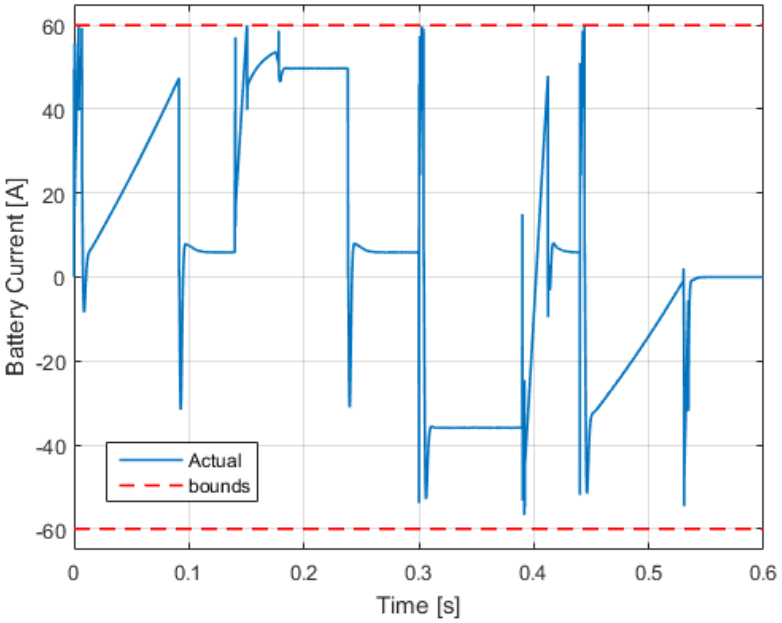

Figure 33.: Modified Controller: Battery Current

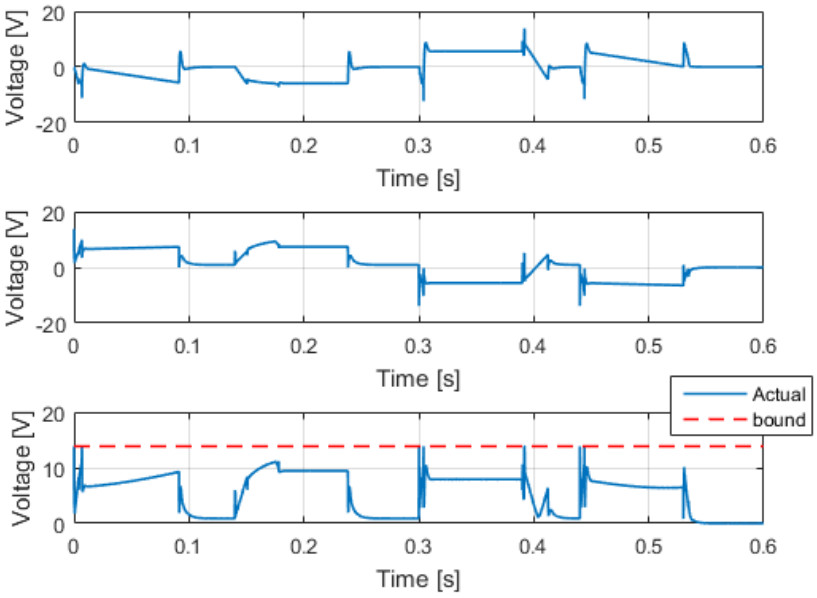

Figure 34.: Modified Controller: Voltages. The top shows $V_d(t)$, the middle $V_q(t)$ and the bottom is the length of these two voltages.

7.2.1 COMMENTS

Overall the results for the given test case is as expected when using the Modified Controller. One can clearly see from Figure 33 that the battery current does not violate the given bounds. This on expense of following the reference signals. A table that shows the increase in the RMSE can be seen in the section 7.5. The RMSE shows how well the respective signals followed their references. From the figures above and the tables in section 7.5, one can see that ensuring the battery current constraint influenced $i_d(t)$ and $T_e(t)$ the most, when comparing them to the ones with the Traditional Controller.

The analysis of the angular position and the velocity are similar to the ones with the Traditional Controller. The angular position is well followed, as the actual value reaches the reference and stabilizes at $t \approx 0.10[s]$, $t \approx 0.24[s]$ and $t \approx 0.53[s]$ as well. The overshoot that occurs at $t \approx 0.395[s]$ is also due to the saturation on the torque, $T_e(t)$. This prevents the motor from utilizing the required angular velocity in that time domain.

For the angular velocity, see Figure 30, it can however be seen the velocity is slightly slower in reaching the new reference given at $t = 0.14[s]$. This is terms of the slope of the signal being less steep when compared the one with the Traditional Controller. The reason for this is the mismatch between the required and the actual torque in that time period, which is due to the reduced voltages to satisfy the battery current constraint.

The torque reference is well followed. However some time periods with a mismatch between the required and the actual exist. A such time period is roughly $t \in [0.15, 0.18][s]$, see Figure 31. In this period $i_d(t)$ is nonzero which affects the $i_q(t)$, and thus $T_e(t)$ from reaching the reference. This is through the term $Z_p \omega_m(t) L_d i_d(t)$ in the $i_q(t)$-dynamics, see equation 6b.

From Figure 32 and Figure 33 one can clearly see that $i_d(t)$ is mostly nonzero in those intervals where the battery current is sufficiently close to its bounds. The negative spike in $i_d(t)$ is due to the overshoot in the angular position. Compared with the values obtained with the Traditional Controller for $i_d(t)$, the nonzero values in Figure 32 are much higher. In terms of energy and heat increase, the bursts in $i_d(t)$ might be negligible due to the short time intervals. However this is not the case in the time period $t \in [0.15, 0.18][s]$ and might have a significant influence on the system. This should be further investigated in future work with this thesis. See Table 4 for RMSE values.

The battery current is as expected. The algorithm 2 does handle the battery current constraint. See Figure 33.

It can from Figure 28 be seen that algorithm 2 satisfies the voltage constraint. Compared with the Traditional Controller, it is also seen that the signs of $V_d(t)$ and $V_q(t)$ are opposite of the terms they want to compensate for.

7.3 DRIVE SYSTEM WITH 2 STATES IN THE NMPC

Some small remarks on the the controllers presented in this section:

- "Original" uses the parameters given in the table below.
- "With lower sampling time on the NMPC" has $T_s = \frac{1}{18}[ms]$. Rest of the parameters are the same as the "Original".
- "With higher time duration between updating the model matrices in the NMPC" has $n = 100$. Rest of the parameters are the same as the "Original".

Table 3.: 2 states in NMPC - Some specific parameters

Prediction horizon N [rev]	7
Output weights $\mathbf{Q}_{\vec{y}}$ and $\mathbf{P}_{\vec{y}}$	$\begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{K_t} \end{bmatrix}$
Input rate weight $\mathbf{R}_{\Delta\vec{u}}$	$\begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}$
Sampling Time T_s	$\frac{1}{12}[ms]$
Control model updated at every n call to the controller, where n is	10

7.3.1 ORIGINAL NMPC

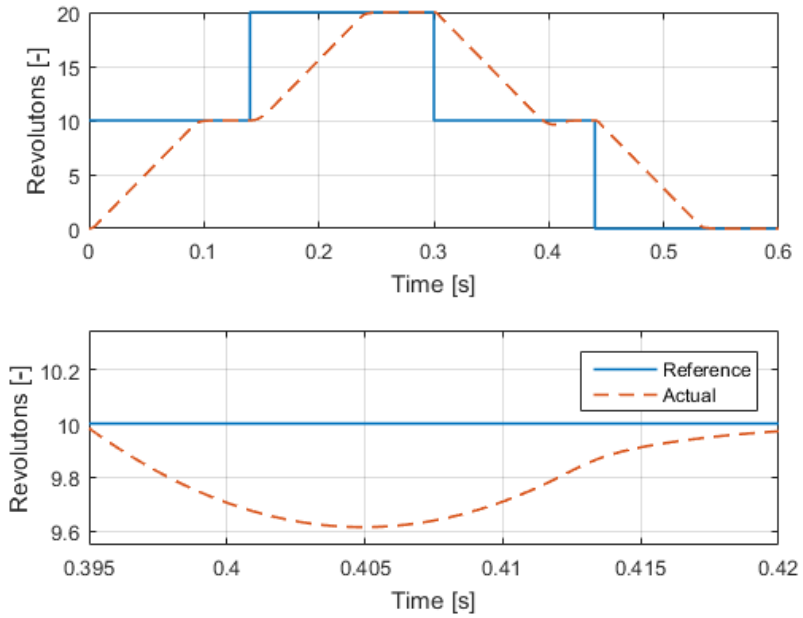

Figure 35.: Drive system with 2 states in the NMPC: Angular Position

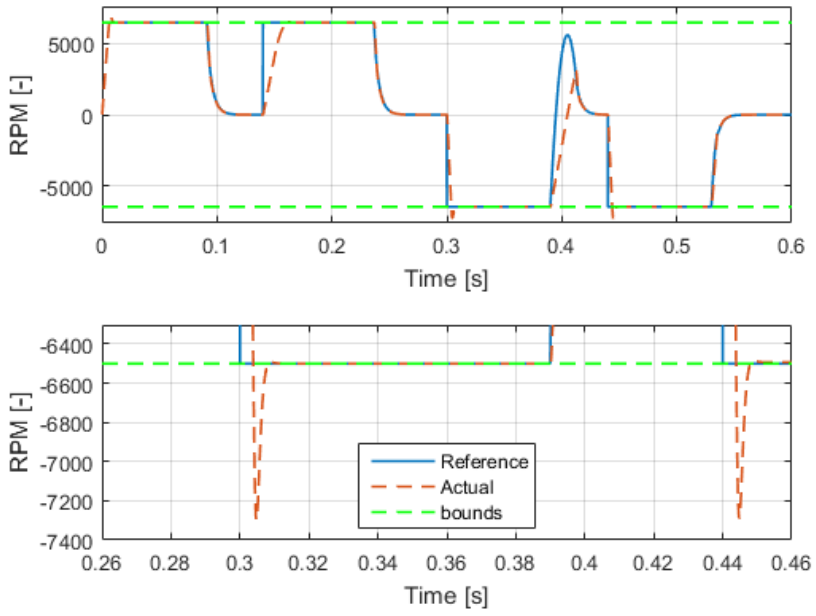

Figure 36.: Drive system with 2 states in the NMPC: Angular Velocity

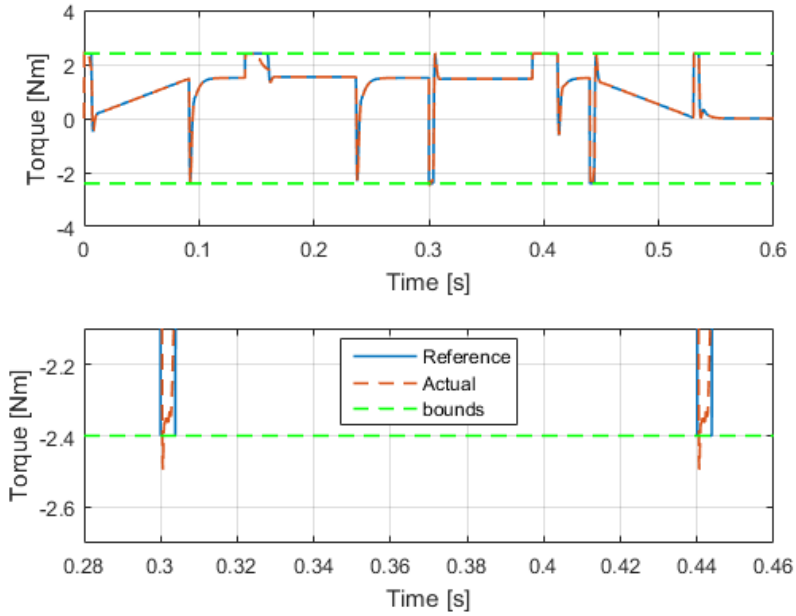

Figure 37.: Drive system with 2 states in the NMPC: Torque

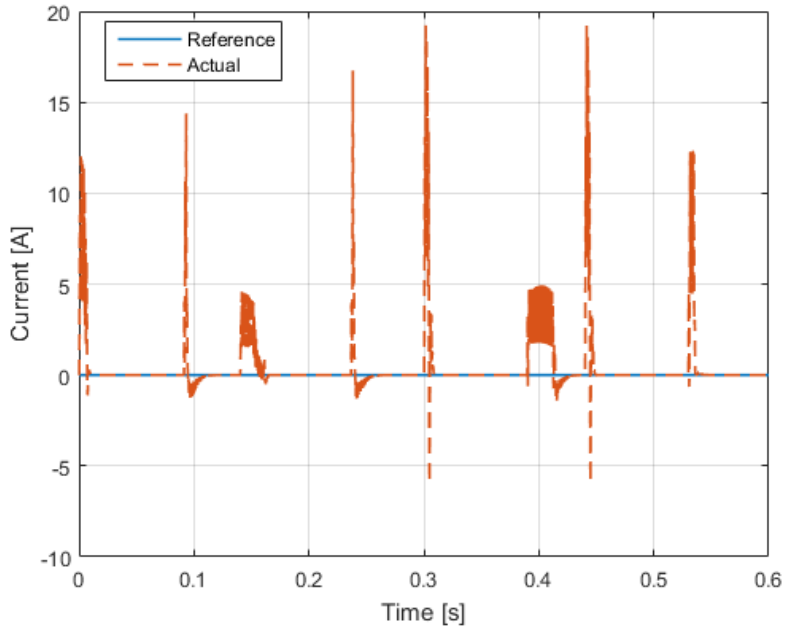

Figure 38.: Drive system with 2 states in the NMPC: Current in d-direction

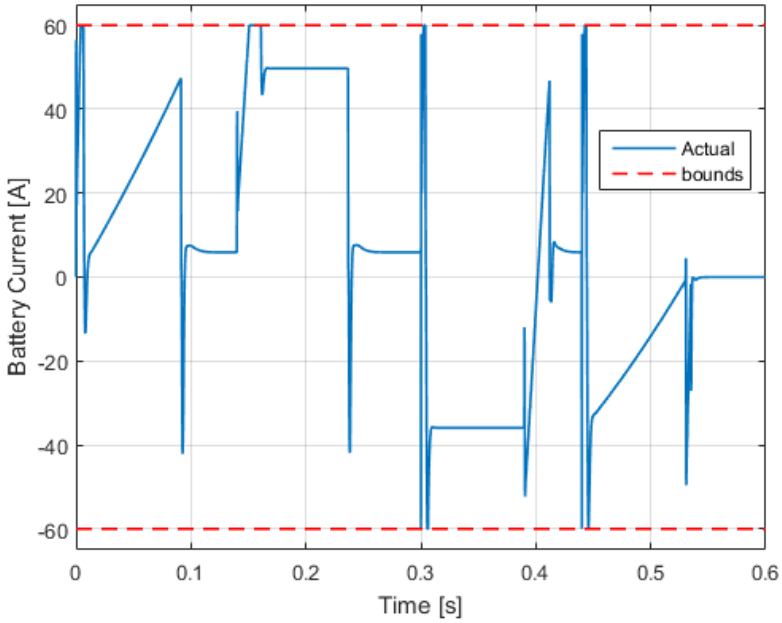

Figure 39.: Drive system with 2 states in the NMPC: Battery Current

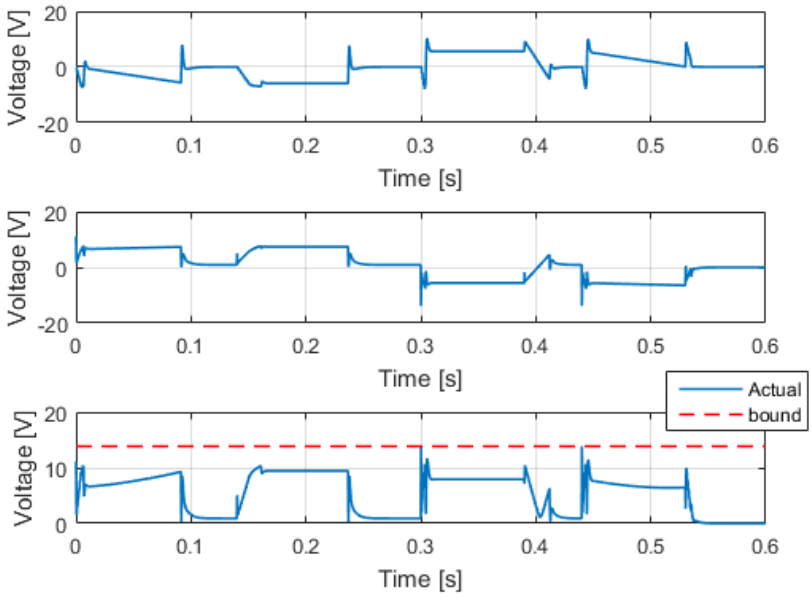

Figure 40.: Drive system with 2 states in the NMPC: Voltages. The top shows $V_d(t)$, the middle $V_q(t)$ and the bottom is the length of these two voltages.

7.3.2 WITH LOWER SAMPLING TIME ON THE THE NMPC

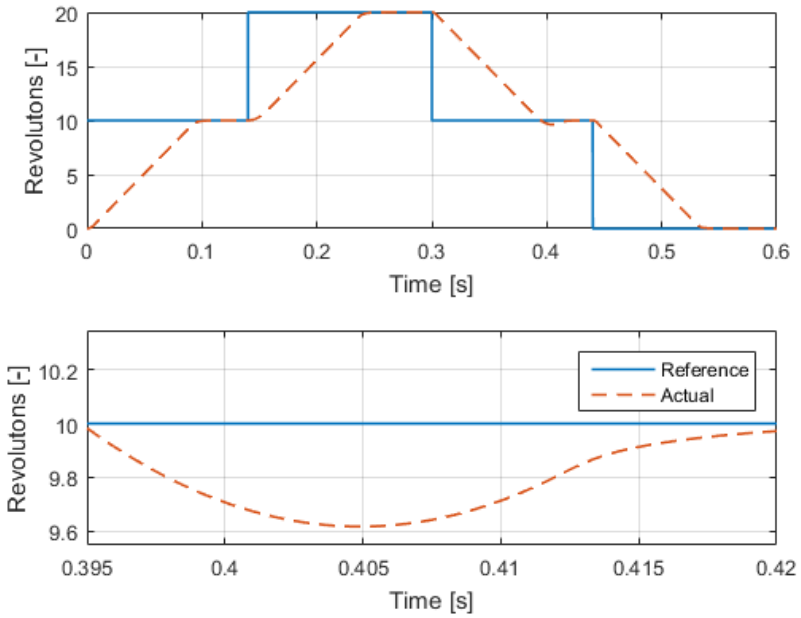

Figure 41.: Drive system with 2 states in the NMPC: Angular Position

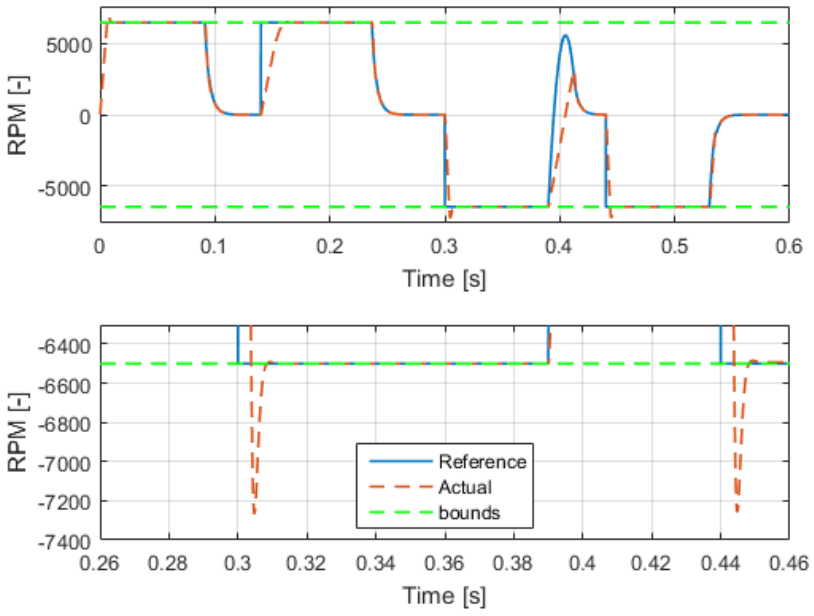

Figure 42.: Drive system with 2 states in the NMPC: Angular Velocity

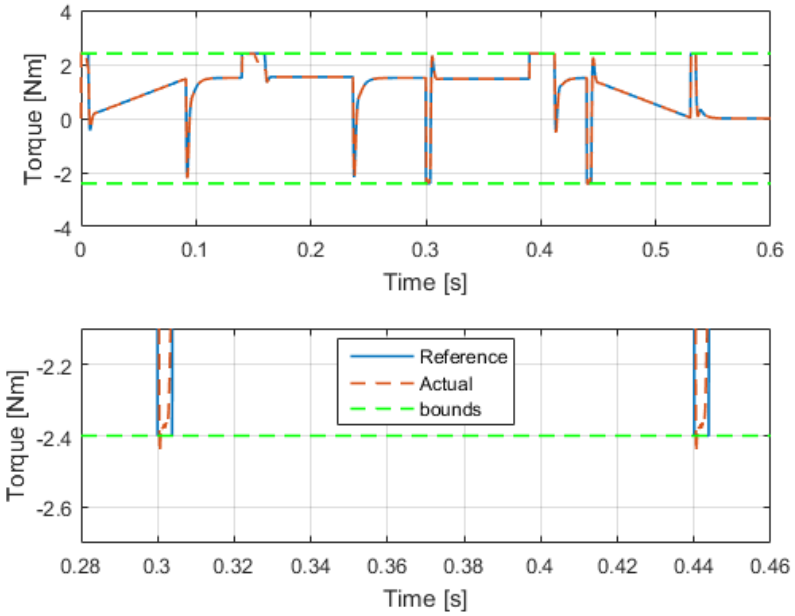

Figure 43.: Drive system with 2 states in the NMPC: Torque

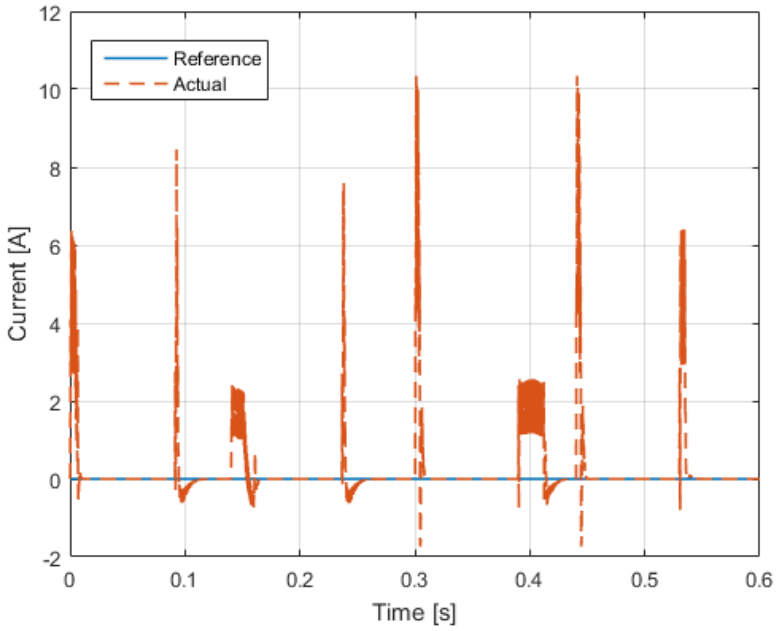

Figure 44.: Drive system with 2 states in the NMPC: Current in d-direction

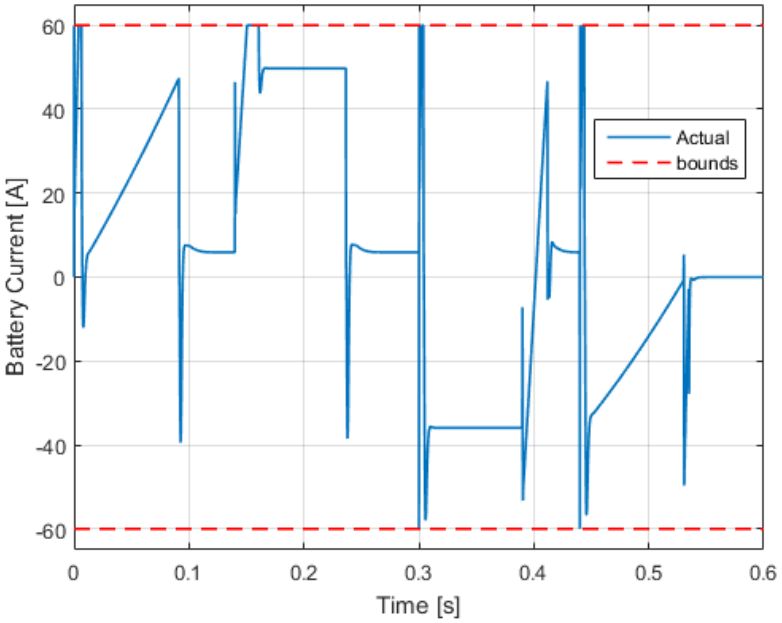

Figure 45.: Drive system with 2 states in the NMPC: Battery Current

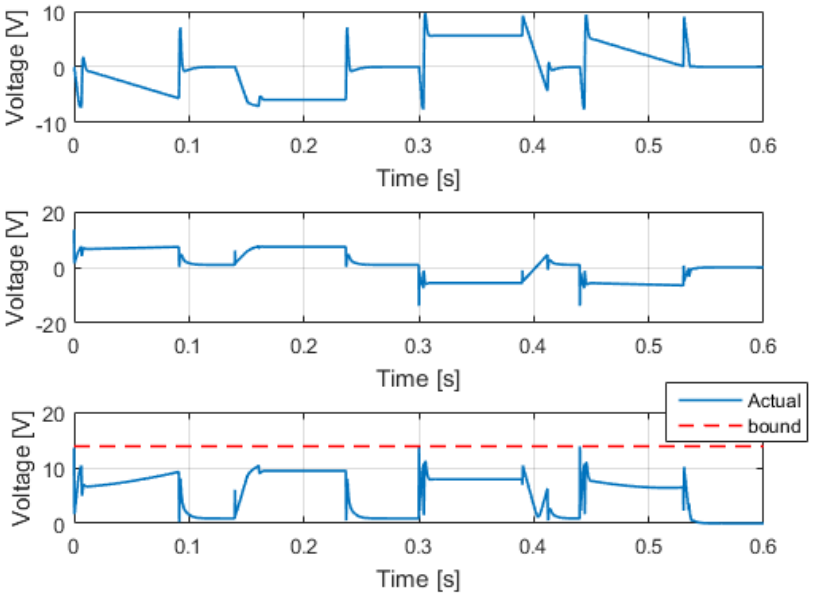

Figure 46.: Drive system with 2 states in the NMPC: Voltages. The top shows $V_d(t)$, the middle $V_q(t)$ and the bottom is the length of these two voltages.

7.3.3 WITH HIGHER TIME DURATION BETWEEN UPDATING THE MODEL MATRICES IN THE NMPC

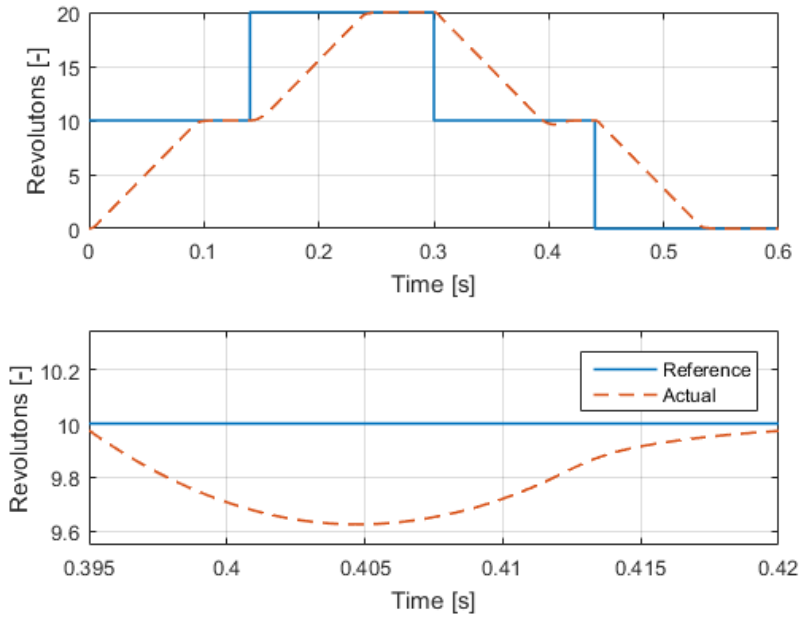

Figure 47.: Drive system with 2 states in the NMPC: Angular Position

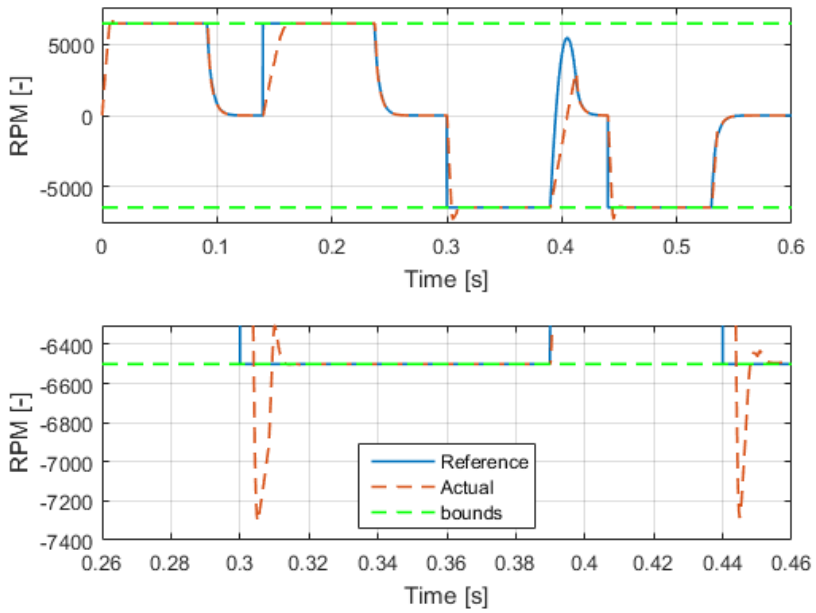

Figure 48.: Drive system with 2 states in the NMPC: Angular Velocity

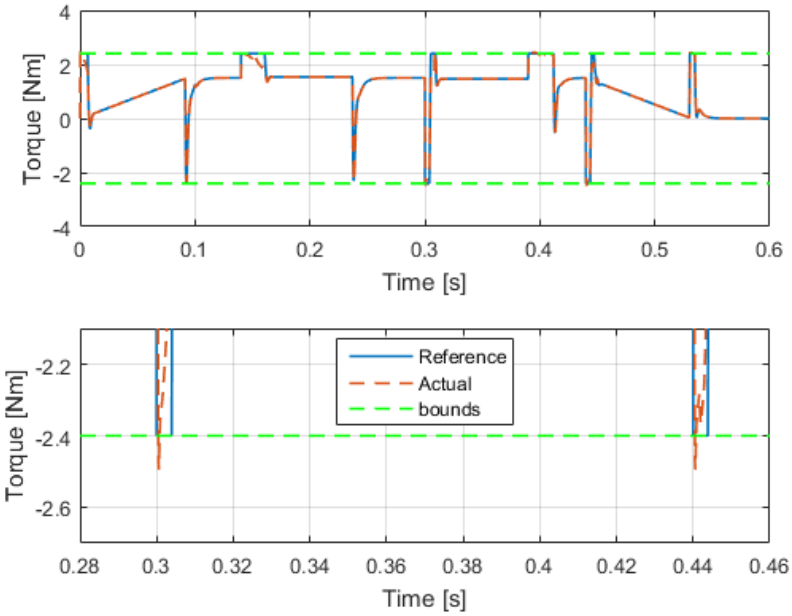

Figure 49.: Drive system with 2 states in the NMPC: Torque

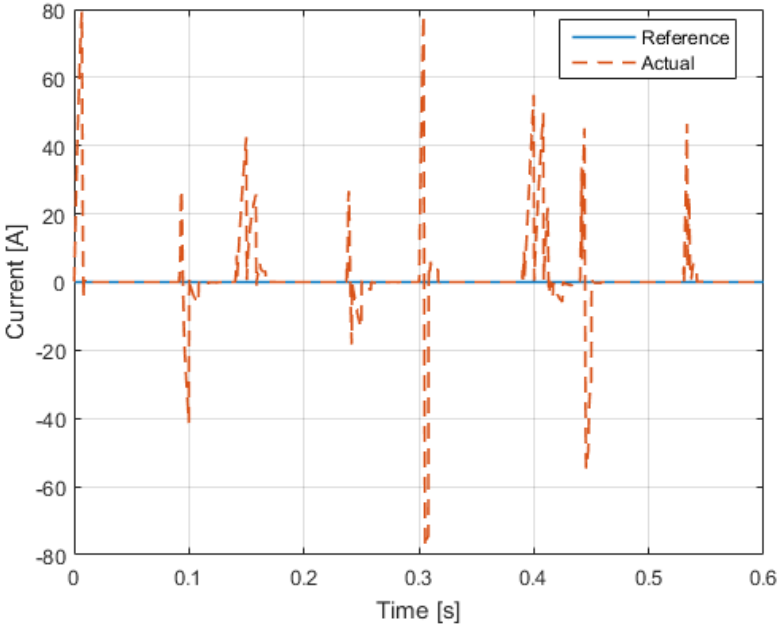

Figure 50.: Drive system with 2 states in the NMPC: Current in d-direction

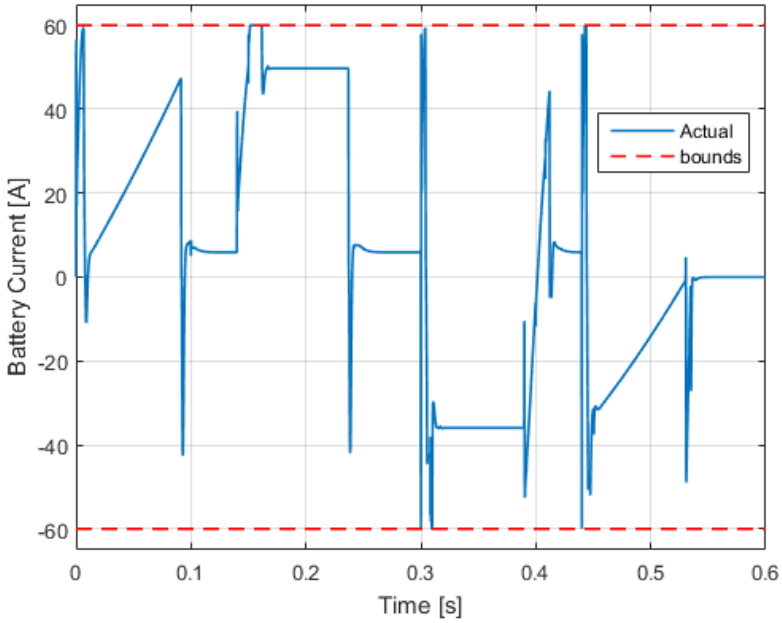

Figure 51.: Drive system with 2 states in the NMPC: Battery Current

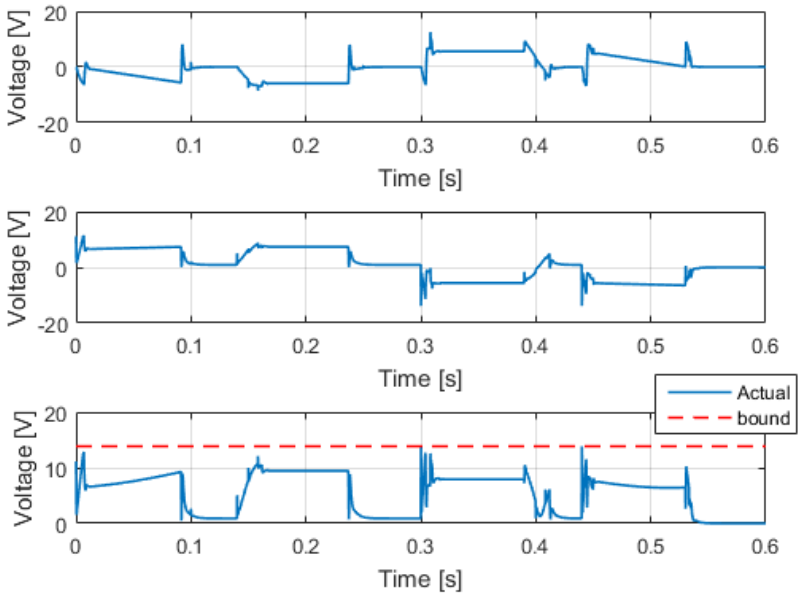

Figure 52.: Drive system with 2 states in the NMPC: Voltages. The top shows $V_d(t)$, the middle $V_q(t)$ and the bottom is the length of these two voltages.

7.3.3.1 COMMENTS

The three controllers have, with support of the tables in section 7.5, overall better responses than the Modified Controller in terms of RMSE-values. This is with exception to the controller where the state matrices are updated fewer times during the simulation, namely "With higher time duration between updating the model matrices in the NMPC". This is with respect to the RMSE-value for the current in d-direction. This makes sense as the controller utilizes state matrices with a higher mismatch from the actual plant. Thus, it is expected that this controller has worse performance than for instance the "Original" and the "With lower sampling time" controller.

That the controller, "With higher time duration between updating the model matrices in the NMPC", has better value in terms of RMSE for the velocity, when compared to the Modified controller, could be an indication that the current controllers in the Modified Controller could have been tuned better. It can for instance be seen from Figure 48 that the controller has some trouble settling at the reference in the period roughly given by $t \in (0.30, 0.32)[s]$. This is not the case for the two other versions of the predictive controllers or the Modified Controller.

Overall the figures in this section speaks for themselves and the analyses is quite similar to the ones presented in the previous comment sections. One can clearly see that the angular position and velocity are well followed. For the figures containing the torque-responses, one can see that updating the model matrices more often results in lower deviations from the reference signal.

From the figures containing the $i_d(t)$ responses, one can observe that the current is nonzero in some specific periods. These periods are given by those intervals where the battery current is sufficiently close to its bounds, and where a "sudden" change in the torque reference occurred. The "sudden" change in the torque reference is a result of the angular position and angular velocity. As stated before the position and speed controllers are tuned to give maximum input to reach the reference signals. Thus the "sudden" changes in the torque reference occurs when a change in the position reference occurs or when the position reference reaches its reference. As seen by the results of the three 2 states NMPC versions, the parameters in Table 3 plays an important role. To minimize the $i_d(t)$ further, one could consider having a higher weight on $i_d(t)$ in the objective function. This is realized by increasing the value of $i_d(t)$ in $\mathbf{Q}_{\bar{y}}$. This should be further investigated in future work with this thesis.

7.4 Drive System with 3 states in NMPC

Some small remarks on the the controllers presented in this section:

- Both controllers uses the parameters given in Table 3.
- One of the two controllers consider a constraint on the angular velocity in the NMPC. This is the same constraint that is used to saturate the velocity reference from the position controller, namely equation 21.

7.4.1 WITHOUT A CONSTRAINT ON THE ANGULAR VELOCITY IN THE NMPC

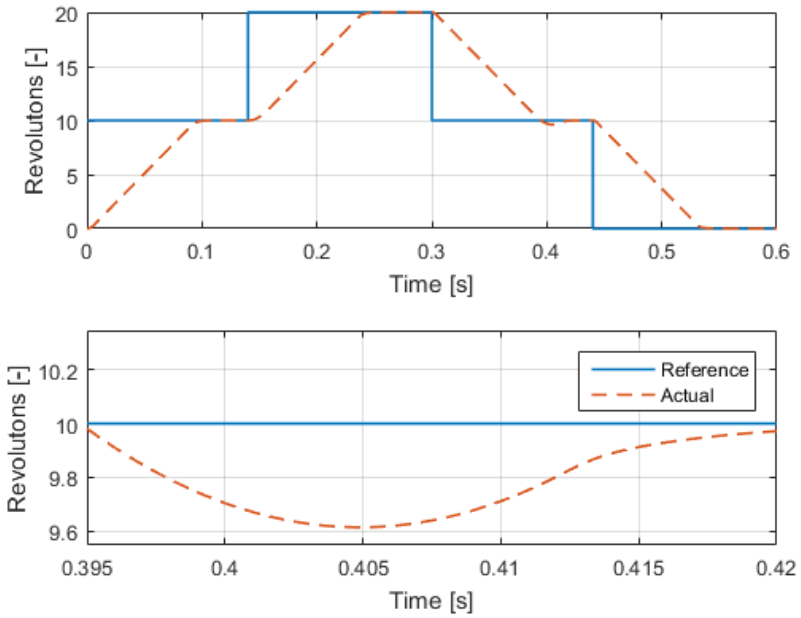

Figure 53.: Drive system with 3 states in the NMPC: Angular Position

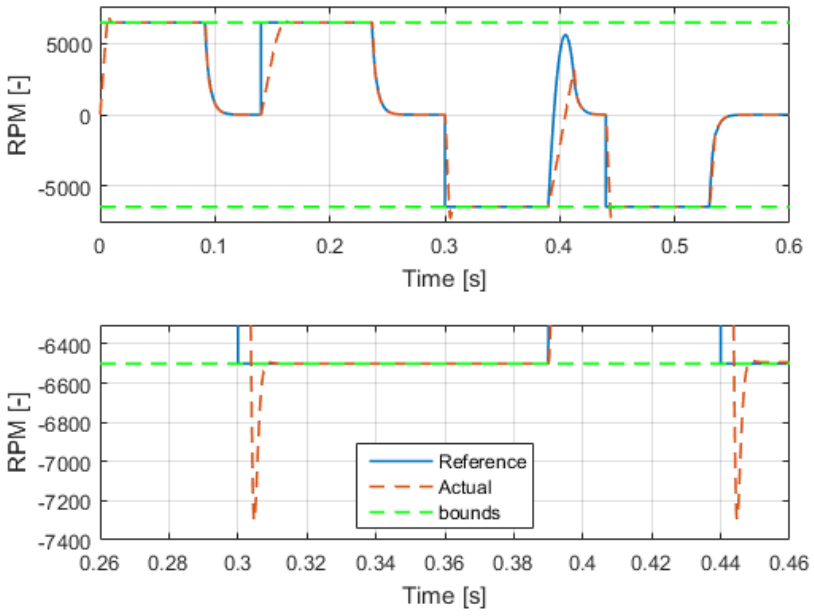

Figure 54.: Drive system with 3 states in the NMPC: Angular Velocity

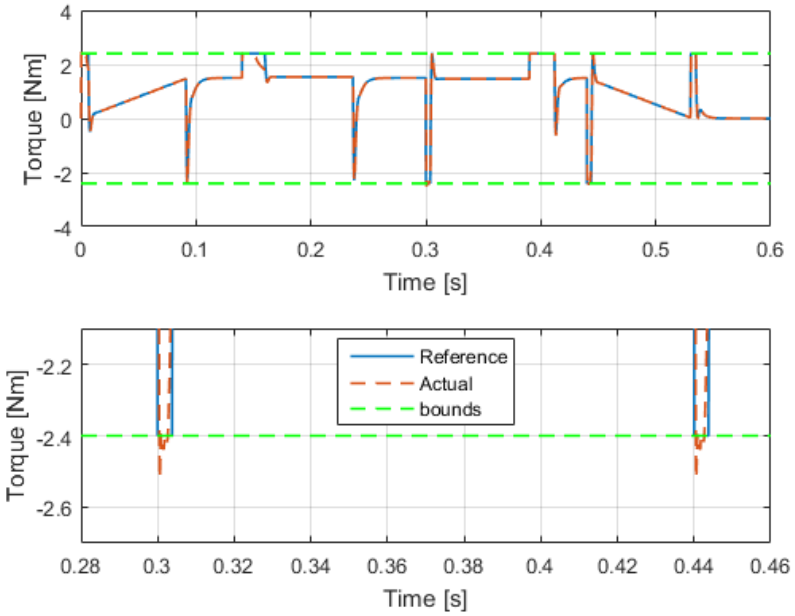

Figure 55.: Drive system with 3 states in the NMPC: Torque

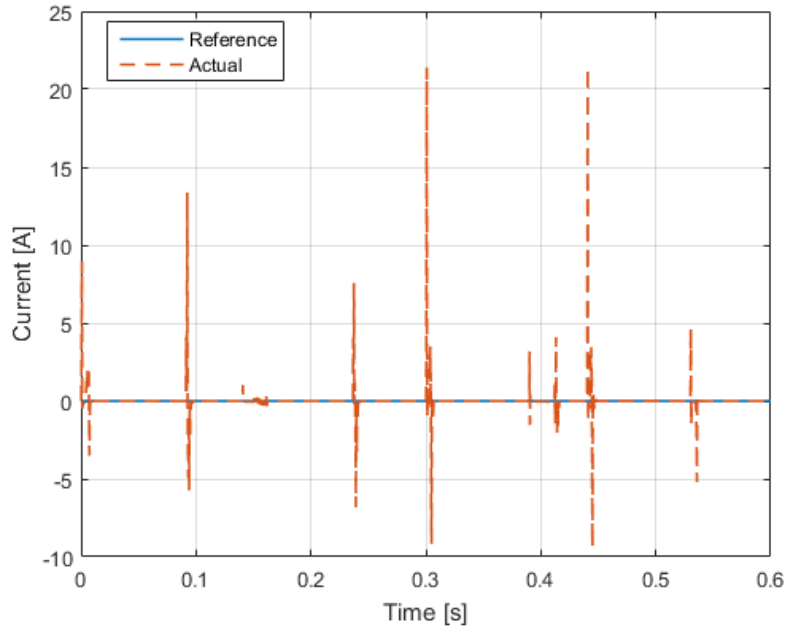

Figure 56.: Drive system with 3 states in the NMPC: Current in d-direction

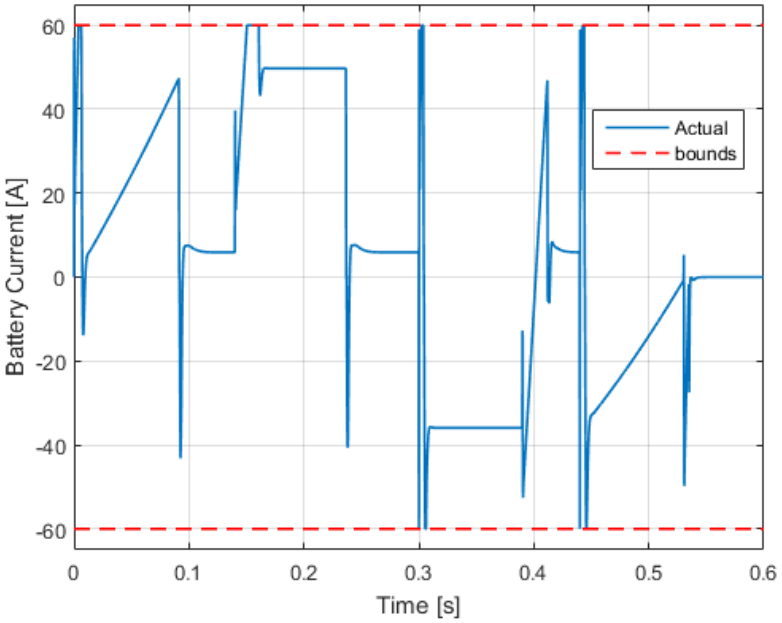

Figure 57.: Drive system with 3 states in the NMPC: Battery Current

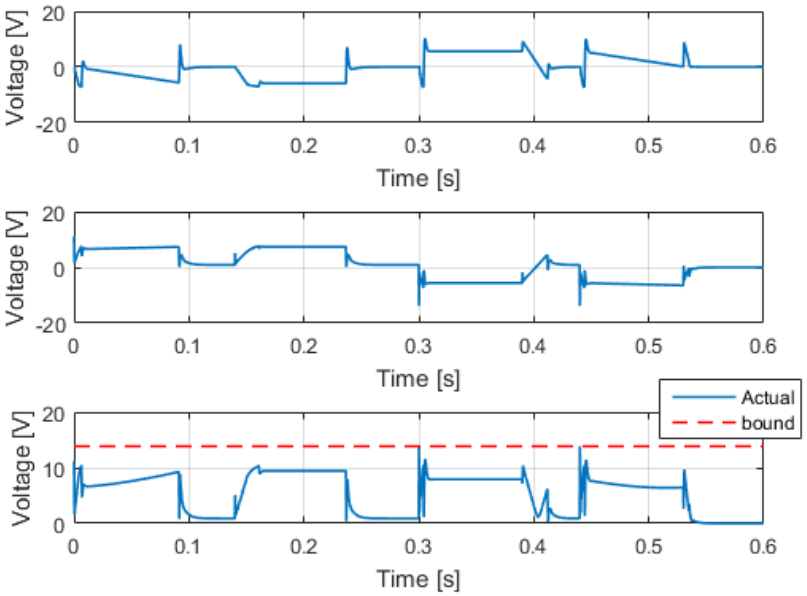

Figure 58.: Drive system with 3 states in the NMPC: Voltages. The top shows $V_d(t)$, the middle $V_q(t)$ and the bottom is the length of these two voltages.

7.4.2 WITH A CONSTRAINT ON THE ANGULAR VELOCITY IN THE NMPC

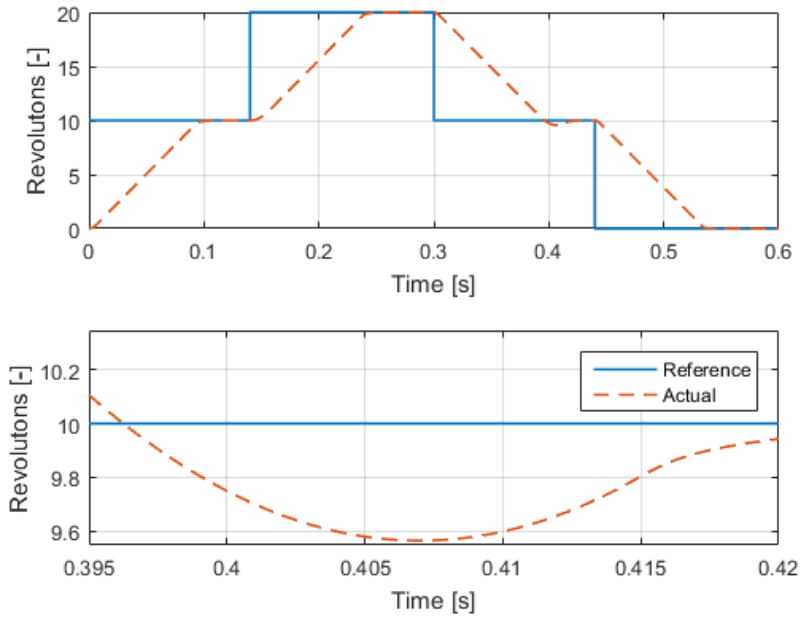

Figure 59.: Drive System with 3 states in the NMPC: Angular Position

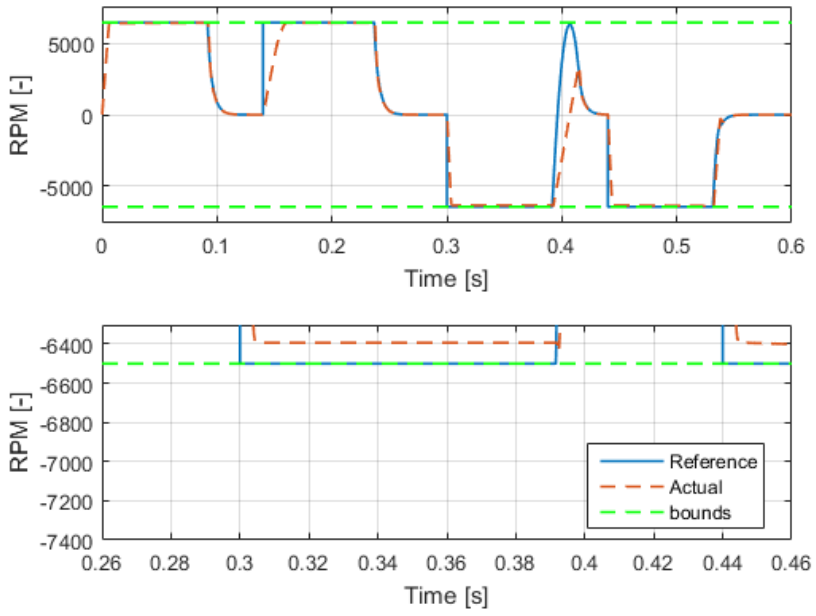

Figure 60.: Drive System with 3 states in the NMPC: Angular Velocity

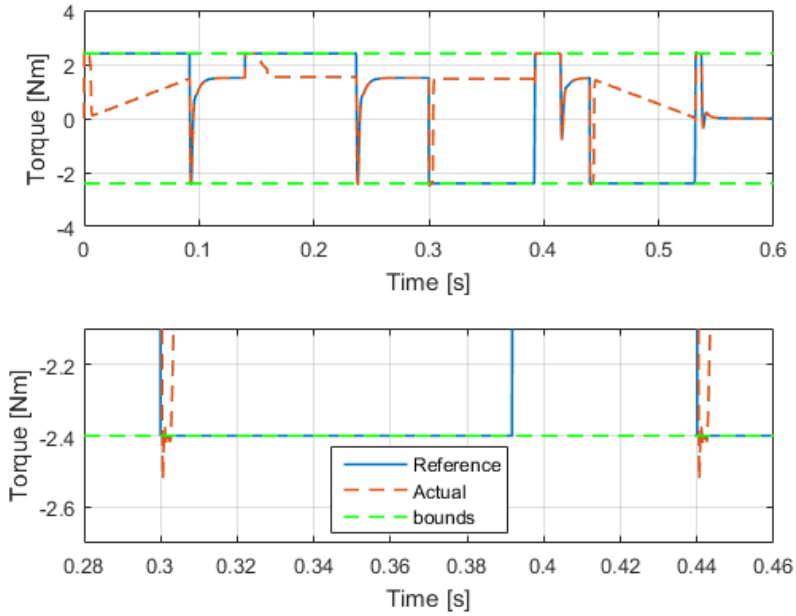

Figure 61.: Drive System with 3 states in the NMPC: Torque

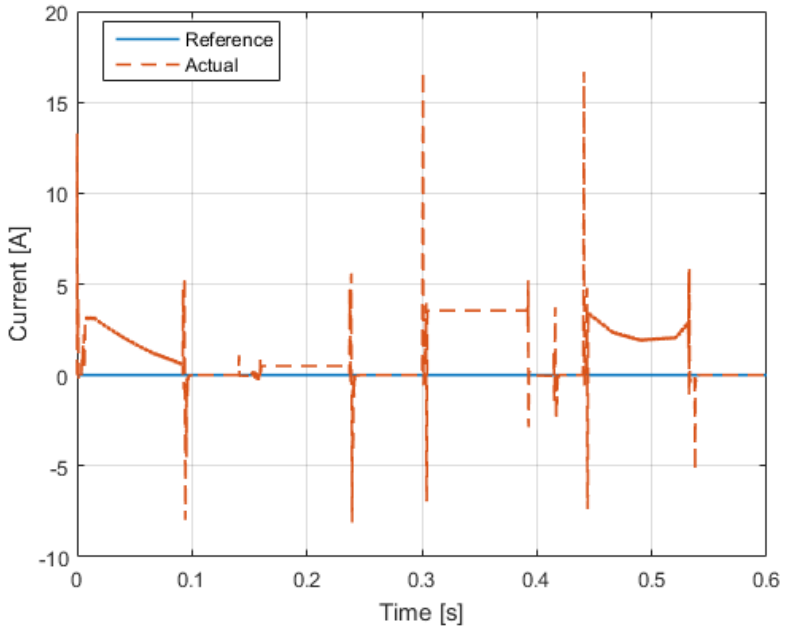

Figure 62.: Drive System with 3 states in the NMPC: Current in d-direction

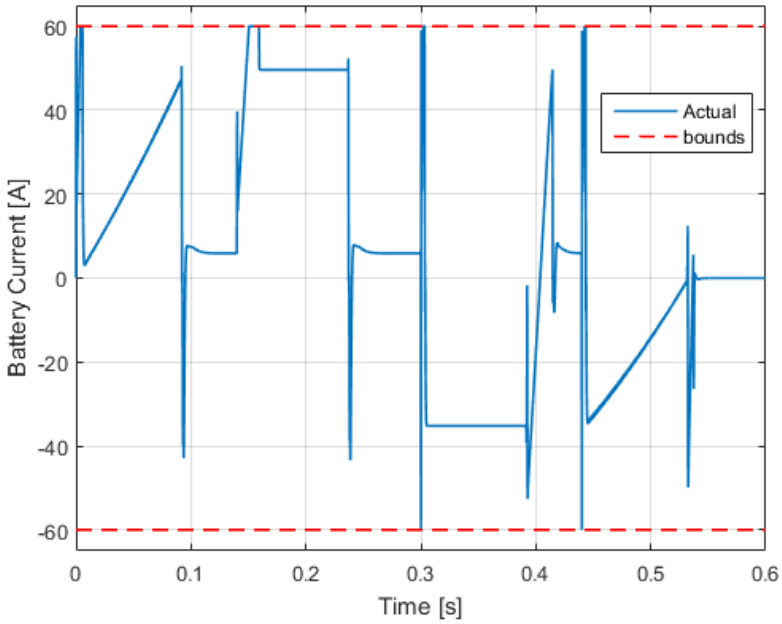

Figure 63.: Drive System with 3 states in the NMPC: Battery Current

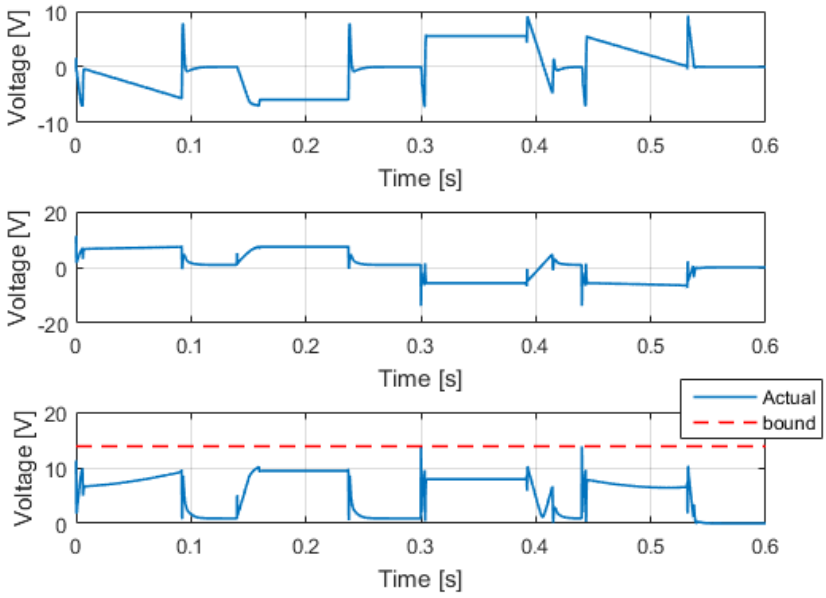

Figure 64.: Drive System with 3 states in the NMPC: Battery Current

7.4.3 COMMENTS

The controller without a constraint on the angular velocity in the NMPC has quite good results. From the tables in section 7.5, it yielded overall better responses than the Modified Controller in terms of RMSE-values. The figures, from Figure 53 to Figure 58, for this controller speaks for themselves. One can clearly it managed to follow the reference signals well, especially for $i_d(t)$ when compared to the controllers where 2 states where used in the NMPC. This makes sense as the controller utilizes a state space model where the angular velocity varies. This gives a better approximation of the real behavior in the prediction horizon.

The behavior of the signals for this controller is quite similar to the ones presented in the previous section. Here, $i_d(t)$ is nonzero due to the same reasons as presented in comment section where the angular velocity was assumed constant in the prediction horizon. It is seen that the nonzero values are given in the intervals where the battery current is sufficiently close to its bounds, and where a "sudden" change in the torque reference occurred.

For the controller with a constraint on the angular velocity in the NMPC the results presented here was not entirely as expected. This is mainly in terms of the torque response obtained. It can from the Figure 61 and Table 4 be seen that the deviation from the reference signal is quite large. A less aggressive torque response was somewhat expected as the NMPC, but only near the bound for the velocity. The reason for this is that due to the velocity constraint in the NMPC, the voltages, and thus the currents and the torque, are limited in a such manner that the velocity does not exceed its bounds. See Figure 60. This prevents the overshoot in the velocity which has been experienced in the others controllers in this paper. Thus, the deviation between the torque response and the reference in the time period $t \in [0, 0.3)[s]$ is as expected.

The unexpected behaviour in the torque response is in the time periods approximately given by $t \in [0.3, 0.39)[s]$ and $t \in [0.44, 0.53][s]$. It is seen there that the torque reference is negative with a value of $2.4[N]$, while the actual torque is positive. Although the angular velocity is close to the bounds in that those time intervals, the author of this paper finds the value of the given torque to be strange. The reason for this is that the torque in all the other controllers were quite close to the reference. A deviation of roughly $3[Nm]$ in average, in the interval $t \in [0.3, 0.39)[s]$, should have another influence on the velocity than what is experienced here. The author expected the torque to have the same sign as the reference with a small deviation when the velocity was close to its bounds.

The possible reasons for why the unexpected behaviour in the torque occurred are:

- The weight matrices are poor tuned. A deviation in the torque should greater importance than keeping the current in d-direction close to zero. This is achieved by increasing the value of $T_e(t)$ in $\mathbf{Q}_{\bar{y}}$.
- The angular velocity is currently implemented as a hard constraint, which means that the velocity in the NMPC cannot exceed the given bounds. This is giving some form of infeasibility that results in the obtained behaviour. A solution to this can be to treat the constraint as a soft constraint instead.
- Another reason could be a mistake in the implementation. However as the same implementation is used to generate the other results, this seem to be unlikely.

This should be further investigated in future work with this thesis.

7.5 SUMMARIZING TABLES

Table 4.: Root Mean Square Error: Following the references

Root Mean Squared Error ($\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}$)				
	Position [rev]	Velocity [rpm]	Torque [Nm]	Current in d-direction [A]
Field Oriented Control (FOC) with Position Control				
Traditional	4.8902	1368.9	0.1855	0.0191
Modified	4.9110	1458.6	0.2512	9.5822
Drive System with 2 states in NMPC				
Original, $T_s = \frac{1}{12}[ms]$ and $n = 10$	4.8116	1275.1	0.1762	1.9329
With lower sampling time, $T_s = \frac{1}{18}[ms]$	4.8019	1261.3	0.1522	1.0591
With higher duration between updating system matrices, $n = 100$	4.8252	1268.4	0.1901	13.0426
Drive System with 3 states in NMPC				
Without constraint on the angular velocity	4.8144	1273.5	0.1752	0.8909
With constraint on the the angular velocity	4.8166	1369.7	2.0382	1.9797

Table 5.: Root Mean Square Error: Following the references - Percentage deviation from Modified Controller

Deviation calculated as $(y_i - y_{\textbf{Modified}})/(y_{\textbf{Modified}}) \cdot 100$				
	Position (%)	Velocity (%)	Torque (%)	Current in d-direction (%)
Field Oriented Control (FOC) with Position Control				
Traditional	-0.4222	-6.1497	-26.1414	-99.8009
Modified	0	0	0	0
Drive System with 2 states in NMPC				
Original, $T_s = \frac{1}{12}[ms]$ and $n = 10$	-2.0242	-12.5858	-29.8690	-79.8282
With lower sampling time, $T_s = \frac{1}{18}[ms]$	-2.2222	-13.5253	-39.4219	-88.9472
With higher duration between updating system matrices, $n = 100$	-1.7471	-13.0430	-24.3410	36.1133
Drive System with 3 states in NMPC				
Without constraint on the angular velocity	-1.9673	-12.6894	-30.2423	-90.7020
With constraint on the the angular velocity	-1.9219	-6.0982	711.3886	-79.3396

Table 6.: Simulation Time

Simulation Time by using tic-toc in MATLAB			
	Without Yalmip [s]	FMINCON [s]	IPOPT [s]
Field Oriented Control (FOC) with Position Control			
Traditional	1.801052	-	-
Modified	10.779881	-	-
Drive System with 2 states in NMPC			
Original, $T_s = \frac{1}{12}[ms]$ and $n = 10$	-	600.160462	1316.001670
With lower sampling time, $T_s = \frac{1}{18}[ms]$	-	1028.267617	3096.820545
With higher duration between updating system matrices, $n = 100$	-	681.077781	1438.499695
Drive System with 3 states in NMPC			
Without constraint on the angular velocity	-	733.867338	1972.020323
With constraint on the the angular velocity	-	907.738119	1597.526253

Part IV

CONCLUDING REMARKS AND SUGGESTION ON FUTURE WORK

8

CONCLUDING REMARKS AND SUGGESTION ON FUTURE WORK

This chapter summarizes what has been done in the thesis and discuss the results obtained. In addition this chapter presents the future work the author suggest could be of interest to look at.

The results presented in the previous chapter showed the performance of the different controllers for the given test case. There, it was of interest to see if the proposed controllers managed to follow the given references while satisfying the given constraints.

A controller that did not satisfy the battery current constraint was used to show the aim of thesis. This was is in the previous chapter referred to as the "Traditional Controller". To satisfy the battery current constraint, two approaches were implemented. One of them was by modifying the Traditional Controller in a such manner that it did not violate the battery current constraint anymore. This was done by using algorithm 2. The other approach in this paper was by using a predictive control methodology. In this paper this was in form of a NMPC as the optimization problem included a nonlinear constraint. Based on the highlighted MPC strategies from the literature, two versions of the NMPC has been presented. One of them assumed constant velocity in the prediction horizon, while the other did not. In addition some sub-versions of the predictive controllers are presented as well where effects of the sampling time, how often the matrices are updated and an additional constraint has been shown.

From the results it is seen that the two approaches indeed managed to keep the battery current within the given bounds, while satisfying the other constraints, and thus achieving the aim of this thesis. This was however on expense on one or more of the responses in terms of following the reference signals, especially with respect to $i_d(t)$. See the summarizing tables in section 7.5.

It was in the results shown that the predictive controllers yielded better responses

than the Modified Controller. This is with respect to the overall performance in terms of the RMSE values. This was the case for both versions of the predictive controllers, which indicates that the assumption of constant velocity could be sufficient to obtain satisfactory performance. However as including a varying velocity in the NMPC resulted in almost the same performance at 12[kHz] as the 18[kHz] for constant velocity, it could be beneficial to include it. As the sampling time plays a vital role for the computational effort, this is something that should be further investigated when using a framework that supports c-code generation.

As the predictive controllers were implemented with a framework that does not support c-code generation, it cannot from this paper alone be given an answer on which version of the NMPC is the way to go. As mentioned before there are several considerations that could influence the computational effort when implementing the predictive controllers on an embedded target. These are for instance which solver is used and how optimized the the framework used is with concern on generating efficient c-code. Another aspect could be if the state matrices are pre-compiled on the embedded target, or if they have to be compiled during the operation.

Further work on this paper the author primarily suggests to see if the Modified Controller yields satisfactory results for the given application. This is for instance with respect to the response of $i_d(t)$. As mentioned earlier, a nonzero value over time would cause an unnecessary increase in heat. If the values are not acceptable with the present tuning, it could be of interest to try to tune again.

If it is of interest to rather utilize a predictive control strategy, a framework that support c-code generation should be looked at. This could for instance be by using framework named ACADO. If the predictive controller is rather placed on the position and velocity controller instead, such that a linear optimization problem can be achieved, toolboxes as JMPC toolbox [24] could rather be utilized. This is a toolbox optimized for embedded targets where one have linear optimization problems. Note that this would mean that the battery current is handled by algorithm 2.

It should also be noted that some further investigation on why the unexpected behaviour in the "3 states in the NMPC" occurred is needed. This is with concern to the torque response when a constraint on the angular velocity was added. According to the author this is might be a result insufficient tuning and the velocity being hard constrained in the NMPC.

Part V

APPENDICES AND BIBLIOGRAPHY

A

MATLAB-CODE

A.1 2 STATES IN THE NMPC

```
%% Author: Shiv Jeet Rai

%% Description: 2-states nonlinear MPC
% Objective function:   only penalizes deviations in id and iq ...
    + rate of change
% Note: the conversion from Te to iq is done outside this function

function [out] = mpcFile_2_states(currentx,currentr,currentw,t)

persistent  Controller Ts N nu nx Ac Bc Gc u0 counter Ad1 Bd1 Gd1
if isempty(u0)
    u0 =[0;0];
end

if t == 0
    % Avoid explosion of internally defined variables in YALMIP
    yalmip('clear')

    Qy = diag([1,1]);       %Weight for tracking id and iq
    Py = Qy;                %Terminal weight
    Ru = 100*diag([1,1]);   %Weight on rate change in u
    Vmax = 24/sqrt(3);      %Voltage/Input limit
    Pmax = 24*60;           %Battery Current / Active Power ...
        Constraint
    %% Setup the optimization problem
    nu = 2; nx = 2;         %Number of inputs and states
    N  = 7;                 %Prediction horizon
    Ts = 1/18e3;            %Sampling time of the NMPC (Same ...
        as on function in Simulink)

    % Want to minimize with respect to x and u
```

```
29      u = sdpvar(repmat(nu,1,N),repmat(1,1,N));       %2 inputs
30      x = sdpvar(repmat(nx,1,N+1),repmat(1,1,N+1));   %2 states
31      r = sdpvar(2,1); %2 references, assumed constant in ...
           prediction horizon.
32      w = sdpvar(1);   %Disturbance, we, assumed constant in ...
           ---||---
33      pastu = sdpvar(nu,1); %Previous input, used in objective ...
           function
34
35      % State matrices, 'full' is used to prevent the framework from
36      % assuming symmetric matrix
37      Ad = sdpvar(2,2,'full');
38      Bd = sdpvar(2,2,'full');
39      Gd = sdpvar(2,1);
40
41      %Retrieve Motor Parameters (Ac,Bc,Gc has to be discretized)
42      [Ac,Bc,Gc,C] = getMotorMatrices();
43
44      constraints = [];  %Used to add the relevant constraints
45      objective   = 0;   %Used to calculate the cost
46
47      [A_oct,b_oct] = octConstraints(); %Retrieve octagonal ...
           constraints (syntax: Az≤b)
48      for k = 1:N     %N is prediction horizon (Terminal cost is ...
           at N+1)
49          if k == 1
50              objective = objective + (r-C*x{k})'*Qy*(r-C*x{k}) ...
51                  + (u{1}-pastu)'*Ru*(u{1}-pastu);
52          else
53              objective = objective + (r-C*x{k})'*Qy*(r-C*x{k}) ...
54                  + (u{k}-u{k-1})'*Ru*(u{k}-u{k-1});
55          end
56          % State space model:
57          constraints = [constraints, x{k+1} == ...
               Ad*x{k}+Bd*u{k}+Gd*w];
58          % Voltage Constraint:
59          constraints = [constraints, A_oct*u{k}≤ b_oct*Vmax];
60
61          % Power/Battery Current Constraint
62          constraints = [constraints,...
63              -Pmax≤(3/2)*(x{k}(1)*u{k}(1)+x{k}(2)*u{k}(2)) ≤ Pmax];
64
65      end
66      objective = objective + (r-C*x{N+1})'*Py*(r-C*x{N+1});  ...
           %Terminal Cost
67
68      %Define an optimizer object which solves the problem for a ...
           particular
69      %initial state and reference
70      %options = sdpsettings('solver','ipopt');   %Ipopt Solver used
```

```
71      options = sdpsettings('solver','fmincon'); %fmincon Solver
72      parameters_in = {x{1},r,w,Ad,Bd,Gd,pastu}; %Parameters ...
            given to the controller at each call
73      solutions_out = {[u{:}],[x{:}]};%Output after each call to ...
            NMPC
74
75      %Controller:
76      Controller = optimizer(constraints,objective,options,...
77          parameters_in,solutions_out);
78  end
79
80  %% Discretization
81  if (isempty(counter) || counter == 10) %update matrices at ...
        initialization
82                                          %or when counter == value
83      counter = 0;
84      [Ad1,Bd1] = c2d(Ac+[0,currentw;-currentw,0],Bc,Ts);
85      [¬,Gd1] = c2d(Ac+[0,currentw;-currentw,0],Gc,Ts);
86  end
87
88  %% Obtain solution
89  [solution,infeasible_flag] = ...
        Controller(currentx,currentr,currentw,...
90          Ad1,Bd1,Gd1,u0);
91  U = solution{1};
92  X = solution{2};
93
94  %% Interpreted MATLAB-func, dont like cells or matrices as ...
        function output
95  % Thus the U and X have been modified to vectors, sorted by ...
        prediction
96  % vec = [values for k = 1, values for k=2,etc].
97  U_vec = reshape(U,N*nu,1);
98  X_vec = reshape(X,nx*(N+1),1);
99
100 %% Debugging and Answer
101 u=U_vec(1:2); %First input given by the controller
102 xout = Ad1*currentx+Bd1*u+Gd1*currentw; %Next output, C is ...
        here eye(2)
103
104
105 % First input and the predicted response is sent out
106 out=[u;xout];
107
108
109 % % For debugging
110 % X;
111 % U;
112 % xout;
113 % pause(2) % A Pause is added for easier reading of terminal
```

```
114
115 %% Update counter and u0
116 u0 = u;
117 counter = counter + 1;
```

A.2 3 STATES IN THE NMPC

```
1  %% Author: Shiv Jeet Rai
2
3  %% Description: 3-states nonlinear MPC
4  % Objective function:   only penalizes deviations in id and iq ...
       + rate of change
5  % Note: the conversion from Te to iq is done outside this function
6
7  function [out] = mpcFile_3_states(currentx,currentr,currentTl,t)
8
9  persistent  Controller Ts N nu nx Ac Bc dc u0 counter Ad1 Bd1 dd1
10
11 if isempty(u0)
12     u0 =[0;0];
13 end
14
15 if t == 0
16     % Avoid explosion of internally defined variables in YALMIP
17     yalmip('clear')
18
19     Qy = diag([1,1]);        %Weight for tracking id and iq
20     Py = Qy;                 %Terminal weight
21     Ru = 100*diag([1,1]);    %Weight on rate change in u
22     Vmax = 24/sqrt(3);       %Voltage/Input limit
23     Pmax = 24*60;            %Battery Current / Active Power ...
           Constraint
24     Wemax = 6500*5;          %Angular Velocity Constraint
25     %% Setup the optimization problem
26     nu = 2; nx = 3;          %Number of inputs and states
27     N  = 7;                  %Prediction horizon
28     Ts = 1/12e3;             %Sampling time of the NMPC (Same ...
           as on function in Simulink)
29
30     % Want to minimize with respect to x and u
31     u = sdpvar(repmat(nu,1,N),repmat(1,1,N));        %2 inputs
32     x = sdpvar(repmat(nx,1,N+1),repmat(1,1,N+1));    %3 states
33     r = sdpvar(2,1); %2 references, assumed constant in ...
           prediction horizon. ...
                                                                  ...
           %1 disturbance, assumed constant in the prediction
```

```
34      pastu = sdpvar(nu,1); %Previous input, used in objective ...
            function
35
36      % State matrices, 'full' is used to prevent the framework from
37      % assuming symmetric matrix
38      Ad = sdpvar(3,3,'full');
39      Bd = sdpvar(3,2);
40      dd = sdpvar(3,1);
41
42
43      %Retrieve Motor Parameters (Ac,Bc,Gc has to be discretized)
44      [Ac,Bc,dc,C] = getMotorMatrices();
45
46      constraints = [];  %Used to add the relevant constraints
47      objective   = 0;   %Used to calculate the cost
48
49      [A_oct,b_oct] = octConstraints();  %Retrieve octagonal ...
            constraints (syntax: Az≤b)
50      for k = 1:N  %N is prediction horizon (Terminal cost is at ...
            N+1)
51          if k == 1
52              objective = objective + (r-C*x{k})'*Qy*(r-C*x{k}) ...
53                  + (u{1}-pastu)'*Ru*(u{1}-pastu);
54          else
55              objective = objective + (r-C*x{k})'*Qy*(r-C*x{k}) ...
56                  + (u{k}-u{k-1})'*Ru*(u{k}-u{k-1});
57          end
58          % State space model:
59          constraints = [constraints, x{k+1} == Ad*x{k}+Bd*u{k}+dd];
60          %Voltage Constraint:
61          constraints = [constraints, A_oct*u{k}≤ b_oct*Vmax];
62          % Power/Battery Current Constraint
63          constraints = [constraints,...
64              -Pmax≤(3/2)*(x{k}(1)*u{k}(1)+x{k}(2)*u{k}(2)) ≤ Pmax];
65          %Angular Velocity Constraint (Comment in/out)
66          %constraints = [constraints,...
67          %    -Wemax≤x{k}(3)≤Wemax];
68      end
69      objective = objective + (r-C*x{N+1})'*Py*(r-C*x{N+1}); ...
            %Terminal Cost
70
71      %Define an optimizer object which solves the problem for a ...
            particular
72      %initial state and reference
73     % options = sdpsettings('solver','ipopt');   %Ipopt Solver used
74      options = sdpsettings('solver','fmincon'); %fmincon Solver
75
76      parameters_in = {x{1},r,Ad,Bd,dd,pastu}; %Parameters given ...
            to the controller at each call
```

```
77      solutions_out = {[u{:}],[x{:}]}; %Outputs obtained after ...
            each call to NMPC
78
79      %Controller:
80      Controller = optimizer(constraints,objective,options,...
81          parameters_in,solutions_out);
82  end
83
84  %% Terms due to linearization (added to Ac and the disturbance ...
        term dc) see figure in MPC subsystem in Simulink
85  id0 = currentx(1); iq0 = currentx(2); we0 = currentx(3); Tl = ...
        currentTl;
86
87  Ac_mod = Ac+[0,we0,iq0;-we0,0,-id0;0,0,0];
88  dc_mod = dc.*[-we0*iq0;we0*id0;-Tl]; %elementwise
89
90  %% Discretization
91  if (isempty(counter) || counter == 100) %update matrices at ...
        initialization or when counter == value
92      counter = 0;
93      [Ad1,Bd1] = c2d(Ac_mod,Bc,Ts);
94      [¬,dd1] = c2d(Ac_mod,dc_mod,Ts);
95  end
96
97  %% Obtain solution
98  [solution,infeasible_flag] = ...
        Controller(currentx,currentr,Ad1,Bd1,dd1,u0);
99  U = solution{1};
100 X = solution{2};
101
102 %% Interpreted MATLAB-func, dont like cells or matrices as ...
        function output
103 % Thus the U and X have been modified to vectors, sorted by ...
        prediction
104 % vec = [values for k = 1, values for k=2,etc].
105 U_vec = reshape(U,N*nu,1);
106 X_vec = reshape(X,nx*(N+1),1);
107
108 %% Debugging and Answer
109 u=U_vec(1:2); %First input given by the controller
110 xout = Ad1*currentx+Bd1*u+dd1; %Next states
111
112 % First input and the predicted response is sent out
113 out=[u;xout];
114
115 % % For debugging
116 % currentr;
117 % X;
118 % U;
119 % xout;
```

```
120 % pause(2) % A Pause is added for easier reading of terminal
121
122 %% Update counter and u0
123 u0 = u;
124 counter = counter + 1;
```

BIBLIOGRAPHY

[1] J. Lemmens, P. Vanassche, and J. Driesen. Pmsm drive current and voltage limiting as a constraint optimal control problem. IEEE Journal of emerging and selected topics in power electronics, 03, 2015.

[2] P. Pillay and R. Krishnan. Modeling of permanent magnet motor drives. IEEE Transactions on industrial electronics, 35, 1988.

[3] T. Sebastian, G. Slemon, and M. Rahman. Modelling of permanent magnet synchronous motors. IEEE Transactions on Magnetics, 22, 1986.

[4] M. S. Merzoug and F. Naceri. Comparison of field-oriented control and direct torque control for permanent magnet synchronous motor (pmsm). World Academy of Science in Engineering and Technology, 45, 2008.

[5] J. Nocedal and S. Wright. Numerical Optimization 2nd editon. Springer-Verlag New York, 2006.

[6] M. Preindl and S. Bolognani. Comparison of direct and pwm model predictive control for power electronic and drive systems. IEEE Applied Power Electronics Conference and Exposition (APEC), 2013.

[7] G. Cimini, D. Bernardini, A. Bemporad, and S. Levijoki. Online model predictive torque control for permanent magnet synchronous motors. IEEE International Conference on Industrial Technology (ICIT), 2015.

[8] S. Bolognani and L. Peretti. Design and implementation of model predictive control for electrical motor drives. IEEE Transactions on industrial electronics, 56, 2009.

[9] L. Wang, S. Chai, D. Yoo, L. Gan, and K. Ng. PID and Predictive Control of Electric Drives and Power Converters using MATLAB/Simulink. Wiley, 2015.

[10] The acado code generation tool - nonlinear model predictive control and moving horizon estimation. `http://acado.github.io/cgt_overview.html`. Last accessed: 28.05.2017.

[11] M. Allen. Single phase vs. three phase power: What you need to know? https://www.datacenters.com/news/information-technology/149-single-phase-vs-three-phase-power-what-you-need-to-know. Last accessed: 18.05.2017.

[12] Drives and mechanisms - elements of cnc machine tools: Electric motors. http://nptel.ac.in/courses/112103174/module4/lec1/3.html. Last accessed: 18.05.2017.

[13] Park, inverse park and clarke, inverse clarke transformations, mss software implementation- user guide. https://www.microsemi.com/document-portal/doc_view/132799-park-inverse-park-and-clarke-inverse-clarke-transformations-mss-software-implementation-user-guide. Last accessed: 18.05.2017.

[14] What is an igbt? http://www.futureelectronics.com/en/transistors/igbt-transistor.aspx. Last accessed: 02.03.2017.

[15] The MathWorks Inc. SimPowerSystems version 6.4 (R2015b). Natick, Massachusetts, United States, 2015.

[16] D. P. Bertsekas. Nonlinear Programming 2nd editon. Athena Scientific, 1999.

[17] Nonlinear programming. https://neos-guide.org/content/nonlinear-programming. Last accessed: 28.05.2017.

[18] A. Bemporad. Automatic control 2 - anti-windup techniques. Lecture Notes (pdf) used at Univerisity of Toronto in 2010-2011. url: https://www.researchgate.net/file.PostFileLoader.html?id=58bbfa484048541d26040a4a&assetKey=AS%3A468515938410496%401488714312138. Last accessed: 30.05.2017.

[19] J. Löfberg. Model predictive control - basics. https://yalmip.github.io/example/standardmpc/. Last accessed: 28.05.2017.

[20] Welcome to the ipopt home page. https://projects.coin-or.org/Ipopt. Last accessed: 30.05.2017.

[21] J. Löfberg. Solvers. https://yalmip.github.io/allsolvers/. Last accessed: 30.05.2017.

[22] Model predictive control. https://de.wikipedia.org/wiki/Model_Predictive_Control. Last accessed: 30.05.2017.

[23] H. Khalil. Nonlinear Control, Global Edition. Pearson, 2015.

[24] J. Currie. jmpc toolbox. http://www.i2c2.aut.ac.nz/Resources/Software/jMPCToolbox.html#. Last accessed: 07.06.2017.