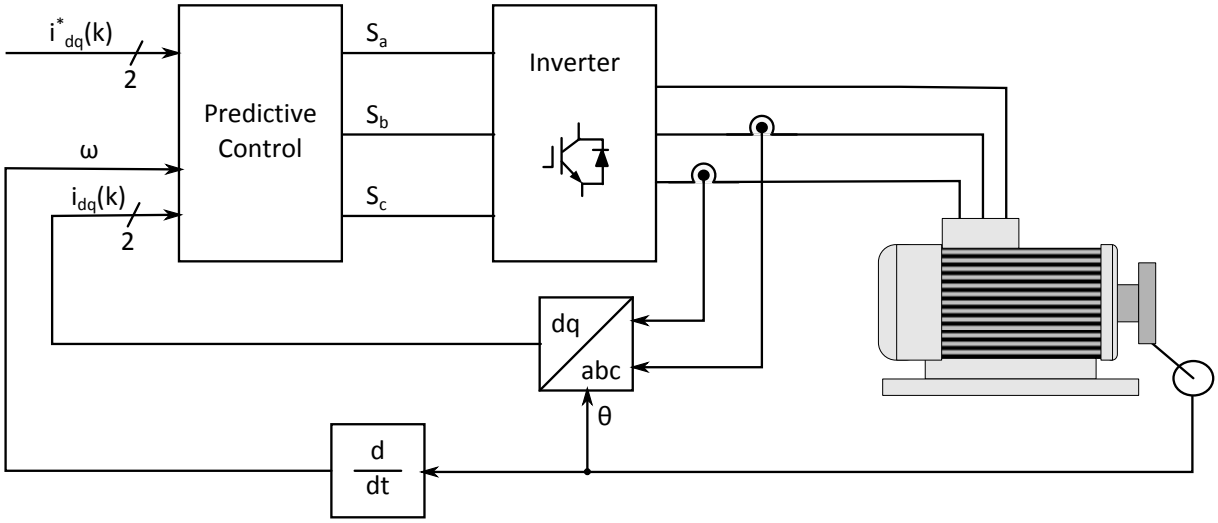
AALBORG UNIVERSITY

STUDENT REPORT

Predictive Controller for PMSM Drive



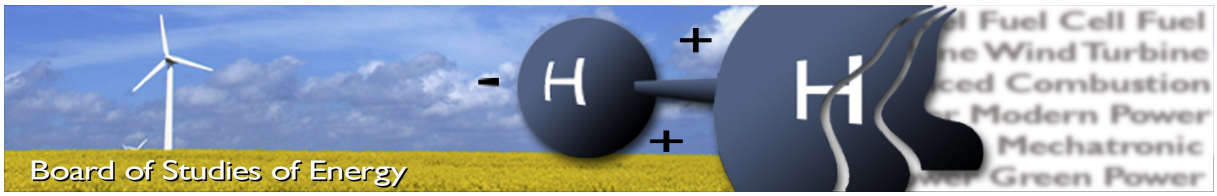
P10 PROJECT

GROUP PED10-1047

DEPARTMENT OF ENERGY TECHNOLOGY

AALBORG UNIVERSITY

JUNE 4TH, 2013



Title:	Predictive Controller for PMSM Drive
Semester:	10
Semester theme:	Master Thesis
Project period:	01.02.2013 to 04.06.2013
ECTS:	30
Supervisors:	Kaiyuan Lu
Project group:	PED4-1047

Judit Baños García

Sonny Quillo

SYNOPSIS:

The purpose of this project was to design and test a Predictive Current Controller (PCC) for fast torque response and high-performance operation of a permanent magnet synchronous machine (PMSMs). Different Predictive control methods have been analysed. Deadbeat and Model Predictive Control (MPC) are both designed and simulated. These methods are compared with the classical Field Oriented Control (FOC) in simulations with Matlab/Simulink. Inverter dead-time and actuation delay compensation are used to improve steady state error and current ripple for both predictive control methods. The deadbeat predictive controller is selected and implemented in the laboratory using a dSpace system. The experimental results show that deadbeat has an improved transient response compared with the FOC.

Copies:	4
Pages, total:	109
Appendix:	4
Supplements:	1 CD

By signing this document, each member of the group confirms that all participated in the project work and thereby all members are collectively liable for the content of the report.

Preface

This Master Thesis is written by group PED10-1047 of two 10^{th} semester students at the Department of Energy Technology at Aalborg University in 2013. The theme for this semester is Power Electronic and Drives.

Reader's guide. The denotation of equations is made by the following principle; the notation (X.Y) suggests that the equation is the Y^{th} in the X^{th} chapter. The principle also applies to tables and figures. This report uses SI units, as described by The International System of Units. The appendices contains the laboratory methodology for obtaining the PMSM drive electrical parameters, the Simulink models of the implemented controllers and the code transcriptions from the matlab function block used. The appendices are included as a supplement to the report, if the reader should need knowledge of the methods used to achieve the results described in the report.

The report is written in LaTeX and the layout is as follows; the bibliography is at the end of the report and the specified details are: Author, year of publishing, url/ISBN/publication. In the report the references are denoted by [surname of the author, year of publication].

A CD-ROM containing the models made in Matlab and Simulink, the articles used, and a PDF version of the report created during the project is attached to the report.

Abstract

Predictive control comprises a large family of controllers. Recent development in microprocessors and Digital Signal Processors (DSP), has encouraged the research of predictive controllers for power electronics and drives applications.

In this Master Thesis different methods of Predictive Current Control (PCC) for a permanent magnet synchronous machine (PMSMs) are investigated. Deadbeat and Model Predictive Control (MPC) are suggested as the most advantageous according simplicity and flexibility.

These proposed predictive control methods are compared in simulation using Matlab/Simulink with the classical and widely used Field Oriented Control (FOC). Given inverter dead-time and actuation delay are influential in the predictive controllers, compensation is suggested and implemented in simulations. The results are discussed and deadbeat was chosen as the best solution because it provides fast dynamic response and low current ripple at a sampling frequency implementable in the given micro-controller.

Deadbeat and FOC were implemented in the laboratory using dSpace DS1103 platform. As a result, deadbeat is found to be a new, intuitive control alternative to the traditional FOC control scheme offering an improved torque response and dynamic performance for PMSMs control.

Contents

Nomenclature

B	Viscous friction coefficient [Nm·s]
J	Motor moment of inertia [kgm^2]
J_L	Load moment of inertia [kgm^2]
K_i	Controller integral gain [-]
K_p	Controller proportional gain [-]
L_d	Direct-axis synchronous inductance [H]
L_d	Quadrature-axis synchronous inductance [H]
L_s	Synchronous inductance [H]
OS	Desired overshoot [%]
P	Instantaneous power [W]
$S_{a,b,c}$	Phase a,b or c switching function [-]
T	Electrical rotor torque [Nm]
T_D	System time delay [s]
T_L	Load torque [Nm]
T_s	Sampling time [s]
$u_{dq}^{\bar{D}T}$	Dead-time voltage vector in d-q reference frame[V]
λ_{PM}	Rotor permanent magnet flux [wb]
λ_{SW}	Weighting factor for switching term [-]
$\lambda_{a,b,c}$	Phase a,b or c flux linkage [Wb]
$\lambda_{d,q}$	Direct or quadrature axis flux [wb]
ω	Electrical angular velocity [rad/s]
ω_m	Mechanical angular velocity [rad/s]

ω_n Natural undamped frequency [rad/s]

θ Electrical angular position [rad]

ζ Damping ratio [-]

$e_{a,b,c}$ Phase a, b or c back EMF [V]

$f_{a,b,c}$ Stator quantity (voltage, current or flux) [-]

$f_{d,q,0}$ Direct, quadrature or zero-sequence quantity (voltage, current or flux) [-]

g Cost function [-]

$i_{a,b,c}$ Phase a,b or c current [A]

$i_{d,q}$ Direct or quadrature axis current [A]

i_{dq}^{p} Predicted direct or quadrature axis current [A]

p Number of pole pair [-]

t_{DT} Dead-time [μs]

$u_{a,b,c}$ Phase a,b or c voltage [V]

$u_{d,q}$ Direct or quadrature axis voltage [V]

u_{dc} DC link voltage [V]

Introduction 1

Power electronics and drives is a hot topic of study, with a great attempt to further increase their performance and efficiency, for many different industrial areas.

In the search for high performance, fast transient response and good control flexibility, various control techniques have been developed in recent years [Lipo, 1996]. Different interesting variables such as speed, position, torque and power can be controlled.

The most widespread and verified control technique is the classical Field Oriented Control (FOC) with Space Vector Modulation (SVM) technique [Irwin et al., 2002]. This control method uses Proportional-Integral (PI) controllers, and controls the current in a dq rotating reference frame. This allows to control the machine torque and magnetic field independently. In order to secure a good performance of a FOC controller, the proportional and integral gains Kp and Ki must be carefully tuned, normally guided by classical control theory.

Direct torque control (DTC) has also been extensively studied because of its simplicity in implementation and fast dynamics. DTC controls the torque directly without using any inner current loop. It regulates the torque by controlling the angle between the stator flux and the rotor flux. The rotor position information is needed in order to locate the rotor flux position. A proper voltage vector from the eight possible voltage vectors offered by a two-level inverter will be chosen, according to the desire to increase the torque (and/or the flux). The main disadvantages of DTC are that the needed switching frequency is high and is normally not fixed. As a consequence, the torque ripple will be relatively large when compared to e.g. FOC. In order to overcome this disadvantage, among many possible, SVM-DTC has been proposed in literature [Swierczynski and Kazmierkowski, 2002], [Ye and Zhang, 2010] as a promising solutions. Both achieve lower switching frequencies, which lead to a reduction of the torque ripple, with a almost similar dynamic response as a conventional DTC scheme.

New control schemes are being studied in the last decade in order to improve especially the transient performances of electrical drives. Predictive control is a promising method that has been developed for this purpose for different applications [Rodriguez and Cortes,

2012]. Some of the new emerging predictive controllers require a high sampling frequency and consume large computation power of Digital Signal Processors (DSP). But thanks to technological development of microprocessors and digital signal processors, these new types of predictive controllers can be implemented in modern DSPs with a reasonably low price for power electronics and electrical drive applications. Among many different control schemes, the predictive controllers are believed to offer a good dynamic response and are advantageous towards simplicity and flexibility.

1.1 Predictive control overview

The main principle of predictive control is to use the drive system model to predict the future behaviour of the variables to be controlled (e.g current). There are different types of predictive controllers as summarized in [Kennel et al., 2008] depending on the criterion used to obtain the optimal performance. Basically, there are four different types: Hysteresis-based, Trajectory-based, Deadbeat and Model-based predictive control (MPC) [Kennel et al., 2008].

The Hysteresis-based controller maintains the control variable within a given hysteresis boundary and the controller in its simplest form may be referred to as a bang-bang controller. The advantages of the hysteresis controller are its simple implementation and fast response. But it is characterised because it has a non-fixed and high switching frequency [Moon et al., 2003]. In trajectory-based control the variable is forced to track a predefined trajectory [Kennel et al., 2008].

In deadbeat, the current reference is to be reached in the next sampling period by using a model of the system to calculate the required output voltage to be generated. This controller also gives a fast dynamic response but it is often sensitive to model parameter errors, system delay and the inverter non-linearities which will reduce th system performance. Different compensation methods, such as delay and dead-time compensation, can be implemented in the deadbeat controller to improve system performance.

Model-based predictive control uses a cost function that needs to be minimised. Furthermore, MPC can be implemented as continuous or finite control set. The finite control set is also called Direct Predictive Control (DPC) [Kouro et al., 2009]. DPC does not need a modulator like SVM unlike the continuous version. It considers a finite set of actuations that correspond to the eight possible switching statues (voltage vectors) of the inverter. All these eight possible voltage vectors are evaluated for each switching period, and the one which minimises the cost function is applied for one switching period. A drawback of the DPC is that the switching frequency is generated directly by the controller cost function may vary. A variable frequency hinders, for example, a proper passive EMI filter design.

Deadbeat control and MPC have been the most widely investigated control methods in recent years, with respect performance and parameters sensitivity evaluations. Furthermore, they offer an intuitive basis, simplicity, flexibility and enable good performance according to [Morel et al., 2009]. Therefore, all further focus on this project will be on the deadbeat and MPC control methods.

On one hand, MPC is one of the most used in practical applications [Kennel et al., 2008]. System constraints are included in both. continuous and finite set, MPC types, such as maximum output voltage limits from the inverter, maximum current and etc. Besides this, the cost functions can be designed to fulfil different goals, like for example, minimising power losses, switching frequency or common mode voltage [Cortes et al., 2009]. This can be done by adjusting the weighting factor introduced in the designed cost function. The cost function is the most important and complex part of MPC. However, it should be taken into account that a more complex controller also means that even huge computation power is needed.

The strategy chosen for the MPC cost function design would depend on the application. Two main cases are found for the case of PMSM. In the case of high power rating, the switching frequency reduction is of importance so that the switching losses may be required to be reduced to improve the efficiency. Properly chosen cost function in MPC, may ensure the minimum number of switch changes in a steady state while the control error would be another subject to minimise in the transient period to ensure the system stability. Another case is for low power servo drives, where the goal is to obtain high dynamics, avoid oscillations, etc.

On the other hand, Deadbeat control needs less computation power than MPC and it uses a modulator to produce a fixed switching frequency like in the case for FOC. Also, it does have higher stress levels on the inverter than MPC, even without optimising the the cost function to minimise switching frequency [Morel et al., 2009]. However, deadbeat can include constraints or non-linearities from the real system. Also deadbeats simple control scheme could get even more complex in order to improve its robustness for parameter changes. In [Li et al., 2012] they present such a method, for a predictive current controller.

In the Table 1.1 at the end of this section, the different predictive controller types are compared according to: model parameter sensitivity, stress level on inverter, computation requirement and if the controller produces a fixed frequency or not.

Table 1.1. Summary of predictive controller types

	Model Predictive Control (MPC)				
	Continuous Control Set	Direct Predictive Control (DPC)	Dead-beat Control	Hysteresis-based Predictive control	Trajectory-based Predictive control
Fixed switching frequency:	Yes	No	Yes	No	No
Sensitivity of model parameters:	Medium	Bad	Good	No data	No data
Stress level on the inverter:	Medium	Good	Bad	No data	No data
Computation requirement:	Bad	Bad	Good	No data	No data
Constraints implementation:	Yes	Yes	Yes	No	No

1.2 Goals

The main focus of this project is the design and implementation of a high performance predictive control for PMSM so that its performance is evaluated and compared with the traditional FOC.

Predictive control is chosen from the different control possibilities because it is easy and coherent, with a comparable simple controller implementation. It is supposed to improve the torque response and dynamic performance compared with the classical control schemes.

The torque is directly dependent on the stator current, therefore, an instantaneous stator currents control with high accuracy and a short transient period, together with a high dynamic performance and low current ripple, can potentially improve the torque performance.

Other goals pursued in this project are to mitigate the computation delay, compensate the deviation caused by the inverter dead-time and study the predictive controllers sensitivity to parameter uncertainties.

Different predictive control schemes could fulfil these goals. Taking into account the simplicity of the implementation and the flexibility for including the different constraints deadbeat and Model Predictive Control (MPC) are the focus of investigation. Both predictive controllers and FOC will be modelled in Simulink/Matlab and compared in simulations.

The simulation results together with the system limitations, from the available set-up, will be evaluated in order to choose the predictive controller which can fulfil the presented goals in practice.

The chosen PCC and FOC will be designed and implemented in the laboratory using dSpace. The simulation models will be verified with the experiments. Based on the experiment the proposed PCC will be verified as an alternative to the classic FOC for high performance control of PMSMs.

1.3 Project limitations

One of the characteristics of predictive control is the large computation time required. The lack of DSPs able to carry on all the required calculations fast enough for power electronics and drives applications, impede the development of the predictive techniques in this area. Computation time is still a limitation. The code should be optimised taking into account that the computation time for practical implementation is limited by the given dSpace DS1103 platform micro-controller.

It should be noticed that the results obtained are very dependent of the given system,

these are, the DC-link voltage is not regulable, the PMSM has a low inductance which benefits the high current ripples and the built inverter imposes a maximum switching frequency. For this project the inverter components are considered ideal, no voltage drop in transistors or diodes are considered in simulations.

Predictive control comprises a broad class of controllers. Because of time constraint, it is not possible to preformed an extensive investigation and complete implementation of all the aspects that could improve the performance of the different methods. A parallel study of the proposed predictive controllers starting from the most simpler implementation and investigating inverter dead-time and computation delay.

1.4 Report structure

In this report, Predictive Control for PMSM is investigated. Firstly, a review of the different control has been done. The test system has been described and the PMSM and Inverter have been modelled in Chapter 2.

FOC and Deadbeat predictive control have been compared and both are described and simulated in Chapter 3 and Chapter 4 respectively. Finite Set Model Predictive Control (MPC) was also implemented in Chapter 5, so a more complete comparison within the predictive control types has been undertaken. The Matlab/Simulink models are introduced in the dSpace digital controller and the system has been validated in the test set-up. Simulation and experimental results respectively are presented at the end of Chapter 3 and 4 for each one of the developed control schemes. MPC is only analysed based on simulations. In Chapter 5 the three control schemes are compared in terms of their sensitibity to model parameter variations. Conclusions and future work are presented in Chapter 6.

Problem Analysis 2

In this chapter, the hardware for the project is presented starting with a general description of the whole system. Knowledge about the system and its components is important to be able to model and analyse it correctly. For modelling and simulating the PMSM, the electrical and mechanical equations of the motor are presented in this chapter. The electrical part of the model will be discretised so the predictive controllers can be derived later for both deadbeat and MPC. In order to control the motor in simulations an inverter model is developed as well.

2.1 System Description

The configuration of one test bench in the Flexible Drive Systems Laboratory (FDSL) is show in Figure 2.1.

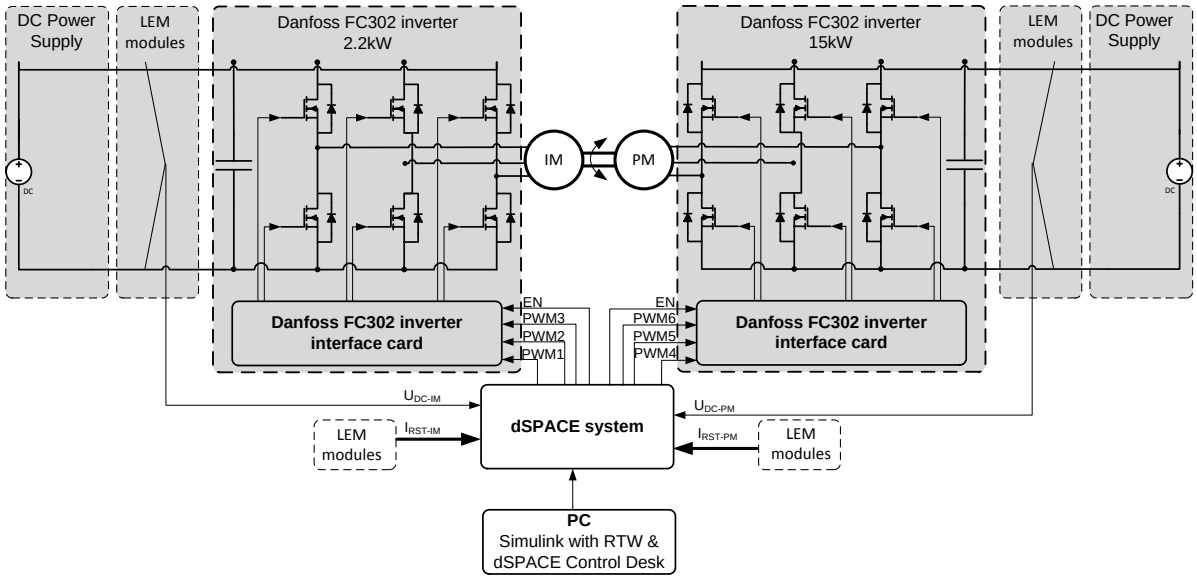


Figure 2.1. Schematic of one test setup in the Flexible Drive Systems Laboratory

The test setup consists of two AC motors. One is a 3-phase induction machine from ABB (M2AA100LA) and the other is a PMSM from Siemens (ROTEC 1FT6084-8SH7). For

this project the PMSM is used and its specifications are listed in Table 2.1. The inductance and resistance for the PMSM are obtained from measurements. The resistance includes the cable resistance. A more detailed description of how the resistance and inductance values were obtained is given in Appendix A. The induction motor is controlled in speed mode, serving as the load torque for the PMSM. The load inertia J_L of the induction motor is 0.069 kg·m^2 and should be added to the inertia of the PMSM. This give a total inertia of 0.0117 kg·m^2.

Table 2.1. Specification of the PMSM

Rated power	9.4 kW	Pole pair (p)	4
Rated torque	20 Nm	Stator resistance (R_s)	0.203 Ω
Maximum torque	65 Nm	Inductance ($L_d = L_q$)	2.10 mH
Rated current	24.5 A	Inertia (J)	0.0048 kg·m^2
Maximum current	86 A	Permanent magnet flux (λ_{PM})	0.123 Wb
Rated frequency	300 Hz	Torque constant	1.01 Nm/A

A Danfoss FC302 (15 kW) 2-level frequency inverter controls the PMSM motor. The inverter specifications are listed in Table 2.2. In the lab the inverter is connected directly to a DC supply. The dead time for the inverter is 2.5 μs. This dead-time is fixed by the inverter hardware to prevent a shoot-through fault. It should be noticed that an inverter may behave non-linearly. For example, dead-time, IGBT voltage drop and turn on/off time will cause voltage error at the inverter output [Jones et al., 2009] and [Summers and Betz, 2004].

Table 2.2. Specification of frequency inverter

	Output	Input
Rated voltage	0-V_{in}	380-500 V
Rated current	32 A	29 A
Rated power	15 kW	
Frequency	0-1 kHz	

dSpace system (DS1103 PPC) is a real time system with high computation power, with many analog and digital I/O ports, like a DSP. But the dSpace system features a direct connection with a PC where the software can compile Simulink models into usable c-code for the system. In order to increase the computational speed, the Simulink model can be defined in c-code by using s-funcion blocks. The dSpace system handles all the measured analog signals from the DC-link and the phase currents as shown in Figure 2.1. These measured values can be shown in a real time interface (RTI) on the PC control desk.

This control desk can contain sliders, scops and other user inputs. dSpace also provides PMW, set and reset signals to the inverter through optic fibers. Some of the hardware specifications for the dSpace system are listed here:

- Motorola PowerPC 750GX running at 1 GHz.
- Slave DSP TI's TMS320F240 Subsystem
- 16 channels (4 x 4ch) ADC, 16 bit, 4 μs,$\pm$ 10 V
- 4 channels ADC, 12 bit, 800 ns, $\pm$ 10V
- 8 channels (2 x 4ch) DAC, 14 bit, 6 μs, $\pm$ 10 V
- Incremental Encoder Interface with 7 channels
- 32 digital I/O lines, programmable in 8-bit groups
- Software development tools (Matlab/Simulink, RTI, Control Desk)

2.2 Permanent Magnet Synchronous Machine

In this chapter, the equations used to describe the Permanent Magnet Synchronous Machine (PMSM) are presented. Also the matrices for dq reference frame transformations are shown. The equations will be used to make a dq model of the motor in Simulink. This chapter is based on the literature from [Fitzgerald et al., 2002] and [Novotny and Lipo, 1996].

The equivalent circuit for one phase of a synchronous machine with a non-salient rotor may be illustrated in Figure 2.2. Each phase is equally distributed with 120° between each other.

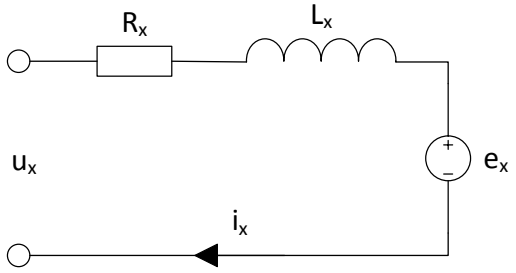


Figure 2.2. Equivalent circuit for a single phase of a motor

From Figure 2.2 the voltage equations for a PMSM are given by Equation 2.1.

$$\begin{aligned} u_a &= Ri_a + \frac{d\lambda_a}{dt} = Ri_a + L_s\frac{di_a}{dt} + e_a \\ u_b &= Ri_b + \frac{d\lambda_b}{dt} = Ri_b + L_s\frac{di_b}{dt} + e_b \\ u_c &= Ri_c + \frac{d\lambda_c}{dt} = Ri_c + L_s\frac{di_c}{dt} + e_c \end{aligned} \tag{2.1}$$

It is shown that the flux linkage does not only depend on the current in each phase. It also depends on the mutual flux created by the other phase currents and the rotor position. The complexity of the flux linkage calculations can be reduced with reference frame transformation. If the dq0 reference frame is aligned with the rotor magnetic field, it will remove the position dependent inductance components. Figure 2.3 illustrates a rotating vector with constant magnitude. The dq frame rotates with the rotors velocity, so that d-axis follows the rotor position θ. Thus the sinusoidal abc components are converted into DC values in steady state which is easier to control. Equation 2.2 shows the dq0 transformation in matrix form. The last row of the matrix corresponds to the zero-sequence component. It is assumed that the three phases in the motor are balanced. Therefore the zero-sequence component in observed from 2.2 is zero.

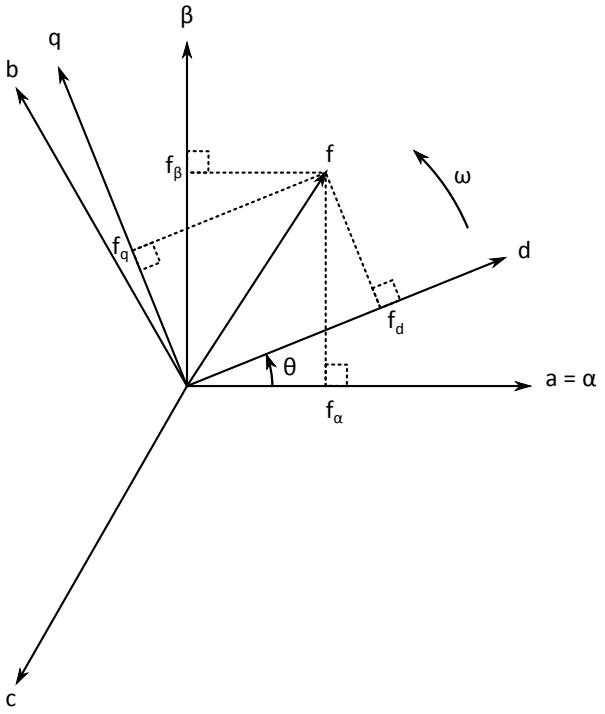


Figure 2.3. dq reference frame transformation

$$\begin{bmatrix} f_d \\ f_q \\ f_0 \end{bmatrix} = \frac{2}{3} \begin{bmatrix} \cos(\theta) & \cos(\theta - \frac{2\pi}{3}) & \cos(\theta + \frac{2\pi}{3}) \\ -\sin(\theta) & -\sin(\theta - \frac{2\pi}{3}) & -\sin(\theta + \frac{2\pi}{3}) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} f_a \\ f_b \\ f_c \end{bmatrix} \tag{2.2}$$

Equation 2.3 and 2.4 are the motor voltage equations in dq reference frame after transformation.

$$u_d = Ri_d + \frac{d\lambda_d}{dt} - \omega\lambda_q \tag{2.3}$$

$$u_q = Ri_q + \frac{d\lambda_q}{dt} + \omega\lambda_d \tag{2.4}$$

Where

$$\lambda_d = L_d i_d + \lambda_{PM} \tag{2.5}$$

$$\lambda_q = L_q i_q \tag{2.6}$$

Equation 2.7 describes the instantaneous power in a three phase motor. By combining Equation 2.2 and 2.7, the instantaneous power using dq components is as given in Equation 2.8. In Equation 2.8 the zero component is also assumed to be zero.

$$P = u_a i_a + u_b i_b + u_c i_c \tag{2.7}$$

$$P = \frac{3}{2}(u_d i_d + u_q i_q) \tag{2.8}$$

The electrical torque produced by the motor is explained with Equation 2.9. If the i_d current for example is set to zero the torque can be directly controlled by the i_q current. This is a simple and widely used method for controlling the torque in the motor. There are two fundamental types of PMSM, with interior rotor permanent magnets or with surface mounted rotor magnets. The magnets are treated as air region in inductance determination, for surface mounted magnets the air-gap distribution is uniform and $L_d = L_q$ (non-salient). Equal inductances in the d-q axes will simplify Equation 2.10 even more and make the torque solely dependent on the i_q current.

$$T = \frac{3}{2}p(\lambda_d i_q - \lambda_q i_d) \tag{2.9}$$

$$\Updownarrow$$

$$T = \frac{3}{2}p\left((L_d - L_q)i_d i_q + \lambda_{PM} i_q\right) \tag{2.10}$$

For a synchronous machine the relationship between the mechanical angular velocity ω_m and the electrical angular velocity ω is described by Equation 2.11. The mechanical angular velocity is simply multiplied by the number of pole pairs. For the torque equilibrium in Equation 2.12 the mechanical angular velocity is used. Equation 2.12 can be rearranged to describe the mechanical dynamic of the motor like the speed if, the electrical torque given by Equation 2.9 and the load torque is known. If the angular

speed is integrated the position of the rotor can be obtained, which is needed for the dq transformations.

$$\omega = p\omega_m \tag{2.11}$$

$$T = (J + J_L)\frac{d\omega_m}{dt} + B\omega_m + T_L \tag{2.12}$$

2.2.1 Motor model discretisation

To predict the current at sampling time k+1 with the measured position, speed and currents at sampling time k, a discrete model of the motor is needed. Equations 2.13 and 2.14 are a rearranging of Equation 2.3 and 2.4, where the derivative of the current have been isolated on the left side of the equal sign.

$$\frac{di_d(t)}{dt} = \frac{1}{L_d}u_d(t) - \frac{R}{L_d}i_d(t) + \frac{\omega(t)L_q}{L_d}i_q(t) \tag{2.13}$$

$$\frac{di_q(t)}{dt} = \frac{1}{L_q}u_q(t) - \frac{R}{L_q}i_q(t) - \frac{\omega(t)L_d}{L_q}i_d(t) - \frac{\omega(t)\lambda_{PM}}{L_q} \tag{2.14}$$

The derivative can also be expressed with Newton's difference quotient as shown in Equation 2.15 if the time steps are approaching zero. In this case the time step Δt will be the sampling time of the controller. If the sampling time is small, this will be a valid approximation for the derivative.

$$\frac{df(t)}{dt} = \lim_{\Delta t \to 0} \frac{f(t + \Delta t) - f(t)}{\Delta t} \tag{2.15}$$

A Taylor series like Equation 2.16 can also be applied instead of Equation 2.15. If only a first order Taylor series is used, it gives the same result as with Equation 2.15.

$$f(t + \Delta t) = f(t) + \Delta t f'(t) + \frac{1}{2!}\Delta t^2 f''(t) + ... \tag{2.16}$$

Equation 2.15 or the first order Taylor expansion from Equation 2.16 are substituting the current derivative which gives Equation 2.17 and 2.18.

$$\frac{i_d(t + \Delta t) - i_d(t)}{\Delta t} = \frac{1}{L_d}u_d(t) - \frac{R}{L_d}i_d(t) + \frac{\omega(t)L_q}{L_d}i_q(t) \tag{2.17}$$

$$\frac{i_q(t + \Delta t) - i_q(t)}{\Delta t} = \frac{1}{L_q}u_q(t) - \frac{R}{L_q}i_q(t) - \frac{\omega(t)L_d}{L_q}i_d(t) - \frac{\omega(t)\lambda_{PM}}{L_q} \tag{2.18}$$

If Δt is equal to a fixed sampling period T_s which is small enough to neglect the changes of angular velocity $\omega(k)$, then the otherwise non-linear back-EMF terms, can be considered

as constants [Morel et al., 2009]. This means that Equation 2.17 and 2.18 can be discretised as shown in Equation 2.19 and 2.20.

$$i_d(k+1) = \frac{Ts}{L_d}u_d(k) + \left(1 - \frac{RTs}{L_d}\right) i_d(k) + \frac{\omega(k)L_qTs}{L_d}i_q(k) \tag{2.19}$$

$$i_q(k+1) = \frac{Ts}{L_q}u_q(k) + \left(1 - \frac{RTs}{L_q}\right) i_q(k) - \frac{\omega(k)L_dTs}{L_q}i_d(k) - \frac{\omega(k)\lambda_{PM}Ts}{L_q} \tag{2.20}$$

Here in Equation 2.21 the predicted $\bar{i}_{dq}(k+1)$ current is presented in matrix form.

$$\begin{bmatrix} i_d(k+1) \\ i_q(k+1) \end{bmatrix} = \begin{bmatrix} 1 - \frac{RTs}{L_d} & \frac{\omega(k)L_qTs}{L_d} \\ -\frac{\omega(k)L_dTs}{L_q} & 1 - \frac{RTs}{L_q} \end{bmatrix} \begin{bmatrix} i_d(k) \\ i_q(k) \end{bmatrix} + \begin{bmatrix} \frac{Ts}{L_d} & 0 \\ 0 & \frac{Ts}{L_q} \end{bmatrix} \begin{bmatrix} u_d(k) \\ u_q(k) \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{\omega(k)\lambda_{PM}Ts}{L_q} \end{bmatrix}$$

$$\Downarrow$$

$$\begin{bmatrix} i_d(k+1) \\ i_q(k+1) \end{bmatrix} = \mathbf{A}(k) \begin{bmatrix} i_d(k) \\ i_q(k) \end{bmatrix} + \mathbf{B} \begin{bmatrix} u_d(k) \\ u_q(k) \end{bmatrix} + \mathbf{C}(k)$$

$$\Downarrow$$

$$\bar{i}_{dq}(k+1) = \mathbf{A}(k)\bar{i}_{dq}(k) + \mathbf{B}\bar{u}_{dq}(k) + \mathbf{C}(k) \tag{2.21}$$

Now a general discretisation of the motor model is obtained. This can be applied to both a model predictive controller and a deatbeat controller.

2.3 Inverter Model

The Danfoss FC302 is a full bridge Voltage Source Inverter (VSI). Figure 2.4 is a simple inverter schematic where each IGBT is considered as an ideal switch in this Chapter. The load impedance representing the PMSM is Y-connected as illustrated in Figure 2.4. The inverter output line-to-line voltage can be described with Equation 2.22. This equation can easily be derived from Figure 2.5 which shows the resulting line-to-line voltage according to the switching states.

$$\begin{bmatrix} V_{ab} \\ V_{bc} \\ V_{ca} \end{bmatrix} = V_{DC} \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} S_a \\ S_b \\ S_c \end{bmatrix} \tag{2.22}$$

The vector $\boldsymbol{S}$ is the switching function with the value 1 when upper leg semiconductor is on and 0 when it is off. Since the two transistors in one inverter leg cannot be turned on at the same time, it will result in 8 different switching stages where two of those are zero voltage. The zero voltage vectors correspond to the switching states 111 and 000, when the three upper switches are either on or off simultaneously and hence short-circuit the DC-link capacitor. Figure 2.5 illustrate the eight different switching states and the resulting line-to-line output voltage from the inverter.

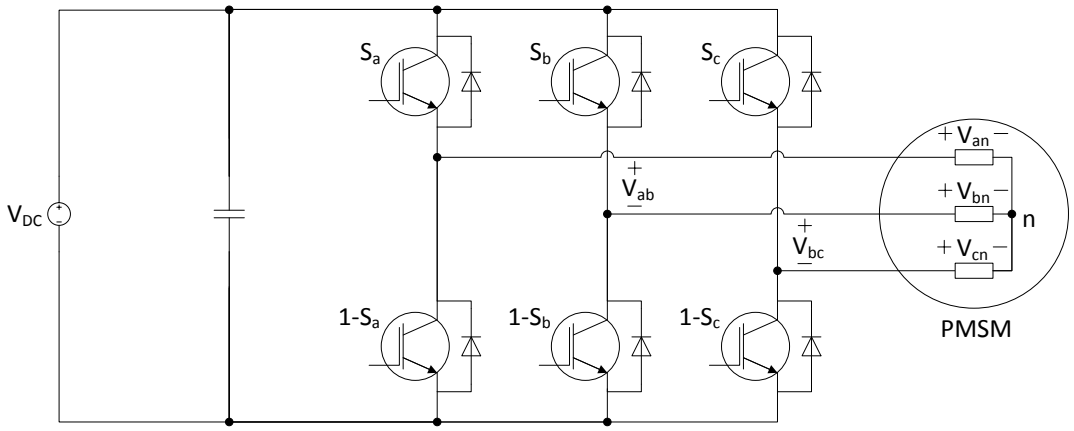


Figure 2.4. A full bridge three-phase voltage source inverter

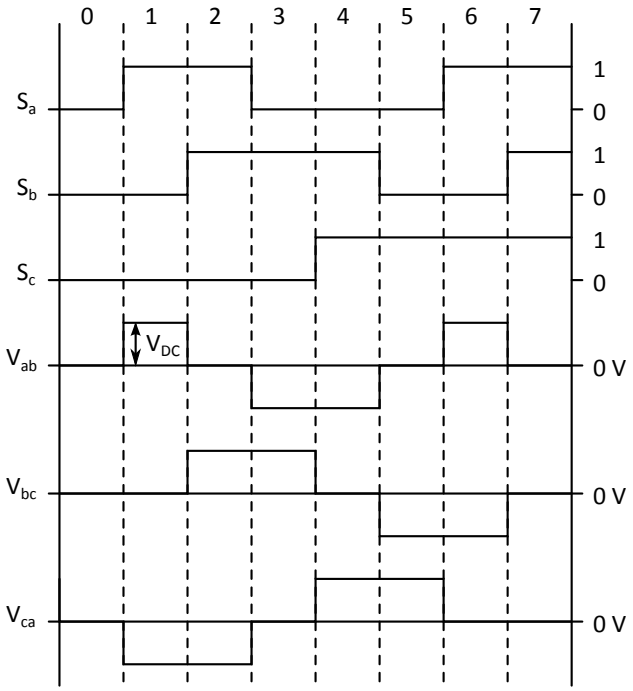


Figure 2.5. Switching functions and the resulting output line-to-line voltages from a full bridge inverter

The terminal voltage in the dq model of the PMSM will be line-to-neutral for a Y-connected motor. Therefore the line-to-neutral voltages are needed. For a balanced system the line-to-neutral voltages are described with Equation 2.23 as a function of $\boldsymbol{S}$.

$$\begin{bmatrix} V_{an} \\ V_{bn} \\ V_{cn} \end{bmatrix} = \frac{V_{DC}}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} S_a \\ S_b \\ S_c \end{bmatrix} \tag{2.23}$$

To describe the inverter switching voltages directly in the dq-reference frame Equation 2.23 and 2.2 can be combined. This combination is shown in Equation 2.24 where the two

matrices have been multiplied together. This equation can be used directly with the dq motor model, where only the switching states for each inverter leg have to be supplied.

$$
\begin{bmatrix} V_d \\ V_q \end{bmatrix} = \frac{2}{3}\frac{V_{DC}}{3} \begin{bmatrix} \cos(\theta) & \cos(\theta - \frac{2\pi}{3}) & \cos(\theta + \frac{2\pi}{3}) \\ -\sin(\theta) & -\sin(\theta - \frac{2\pi}{3}) & -\sin(\theta + \frac{2\pi}{3}) \end{bmatrix} \cdot \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} S_a \\ S_b \\ S_c \end{bmatrix}
$$

$$
\Downarrow
$$

$$
\begin{bmatrix} V_d \\ V_q \end{bmatrix} = \frac{2}{3}\frac{V_{DC}}{3} \begin{bmatrix} 3\cos(\theta) & 3\cos(\theta - \frac{2\pi}{3}) & 3\cos(\theta + \frac{2\pi}{3}) \\ -3\sin(\theta) & -3\sin(\theta - \frac{2\pi}{3}) & -3\sin(\theta + \frac{2\pi}{3}) \end{bmatrix} \cdot \begin{bmatrix} S_a \\ S_b \\ S_c \end{bmatrix}
$$

$$
\Downarrow
$$

$$
\begin{bmatrix} V_d \\ V_q \end{bmatrix} = \frac{2V_{DC}}{3} \begin{bmatrix} \cos(\theta) & \cos(\theta - \frac{2\pi}{3}) & \cos(\theta + \frac{2\pi}{3}) \\ -\sin(\theta) & -\sin(\theta - \frac{2\pi}{3}) & -\sin(\theta + \frac{2\pi}{3}) \end{bmatrix} \cdot \begin{bmatrix} S_a \\ S_b \\ S_c \end{bmatrix} \tag{2.24}
$$

Field oriented control 3

Field oriented control (FOC) is a classical approach to control ac-machines. FOC is implemented in this project in order to validate the performance of the proposed predictive control through a fair comparison. In this chapter, the FOC and space vector modulation principles are explained. The PI current controller is designed and the whole system is simulated and implemented for experimental testing.

In Figure 3.1 the general structure of the current vector control of the PMSM is shown.

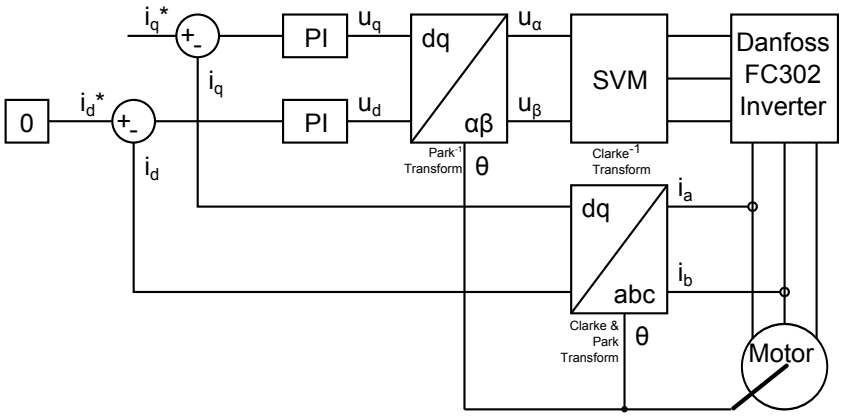


Figure 3.1. General structure for PMSM with a PI current controller

The main principle of FOC is that it uses a coordinate system that aligns with the rotor flux, it is call d-q rotating reference frame. In Section 2.2 the transformation from the 3 phase non-rotating reference frame (abc) to dq reference frame for a PMSM was described. In that frame, it can be approached an independent control of the electrical torque and the magnitude of the d-axis flux, that is by controlling i_q and i_d respectively.

According to Equation 2.10, when L_d=L_q, the torque is only dependent on the i_q current. The flux is maintained constant by setting i_d^* to 0.

The measured stator currents i_d and i_q are subtracted from the reference signals i_d^*, i_q^* producing a error signal for the PI controllers, as illustrated in Figure 3.1. The PI controller generates a output voltage in the rotating reference frame.

The controller output voltages u_d and u_q are then transformed to the stationary reference frame α, β. The obtained u_α, u_β voltages is then applied at the motor by using the Danfoss FC302 Inverter, which is controlled by a Space Vector Modulation (SVM) technique. The electrical rotor position used in both reference frame transformations is obtained from an encoder mounted on the motor shaft.

The SVM technique and the controller design are explained in detail in the following sections.

3.1 Space Vector Modulation

Variable voltage and frequency can be applied from the Danfoss FC302 Inverter. There are several pulse width modulation (PWM) techniques that have been deeply studied and applied [da Silva et al., 2011]. The most common ones are carrier-based sinusoidal SPWM and space vector modulation (SVPWM). The SVPWM determines the switching pulse width and their position improving the DC-link utilization by 15.15% and facilitating the digital implementation [Iqbal et al., 2006]. Therfore SVPMW is chosen for this project, which corresponds to 1.5 times the switching period

The SVPWM uses complex voltage vectors for control. There are eight switch combination possibilities in a three-phase inverter. Six of these combinations apply an non-zero output voltage, defining the six active vectors. The two left are the zero vectors, these states corresponds with the short-circuiting of the top or bottom switches in the inverter. In Figure 3.2 the phase voltage space vectors are represented. The active vectors are marked from 1 to 6 dividing the plane into six sectors spaced 60 degrees. The maximum amplitude of them is the maximum voltage that the inverter can deliver $2/3V_{DC}$ The zero vectors are positioned at the center of the hexagon.

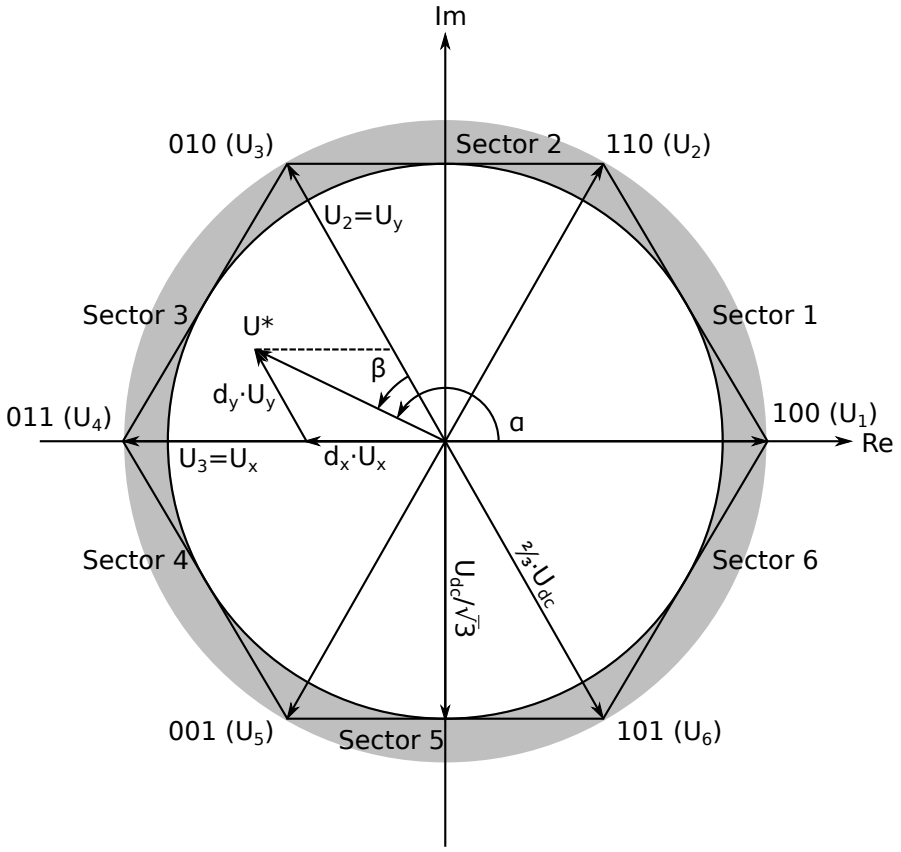


Figure 3.2. Space vector representation of inverter

U^* is the reference vector from the controller. The maximum magnitude of U^* is $U_{dc}/\sqrt{3}$, this is the radius of the circle which fits precisely inside the hexagon. If the reference vector $U^* > U_{dc}/\sqrt{3}$ then the inverter is operated in a nonlinear over modulated mode [Irwin et al., 2002].

The reference U^* represents the three-phase sinusoidal voltage that is generated by the SVPWM. U^* position places it in one of the six sectors on3.2. The sector defines the two actives vectors which, in combination with zero vectors will define the reference. For a general case where the active vectors are called U_x and U_y, and according the trigonometry on figure 3.2 the duty cycles can be defined as d_x, d_y and are obtained as follows:

$$\vec{U^*} = |U^*| \begin{bmatrix} cos(\beta) \\ sin(\beta) \end{bmatrix} = d_x \left|\vec{U_x}\right| \begin{bmatrix} 1 \\ 0 \end{bmatrix} + d_y \left|\vec{U_y}\right| \begin{bmatrix} cos(\frac{\pi}{3}) \\ sin(\frac{\pi}{3}) \end{bmatrix}$$

$$= d_x \frac{2}{3} U_{dc} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + d_y \frac{2}{3} U_{dc} \begin{bmatrix} cos(\frac{\pi}{3}) \\ sin(\frac{\pi}{3}) \end{bmatrix}$$

$$\Downarrow$$

$$d_x = \frac{\sqrt{3}\,|U^*|}{U_{dc}} \sin\left(\frac{\pi}{3} - \beta\right) \tag{3.1}$$

$$d_y = \frac{\sqrt{3}\,|U^*|}{U_{dc}} \sin\left(\beta\right) \tag{3.2}$$

The residual duty cycle, d_0, corresponds with the zero vectors and determines the magnitude of the reference vector.

$$d_0 = 1 - d_x - d_y \tag{3.3}$$

The switching sequence is implemented symmetrically where the residual duty cycle d_0 will be equally distributed between the two zero vectors.

As an example, the switching signal for each inverter leg for a reference situated in sector 1 is shown in the following Figure:

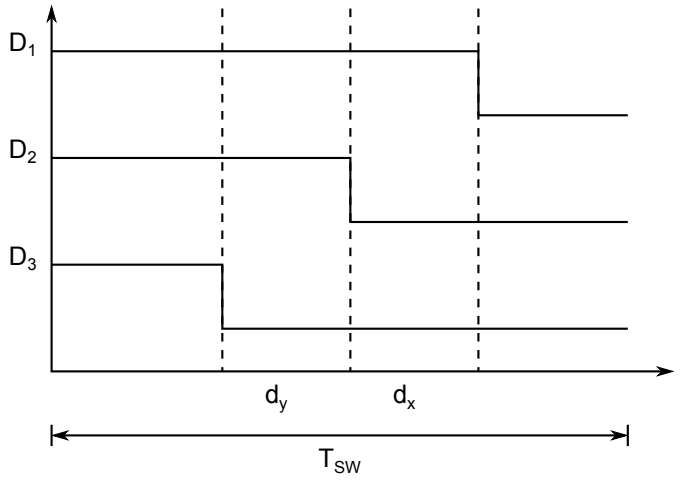


Figure 3.3. D_1, D_2 and D_3 are the output duty cycles for each inverter leg from the SVM which are used in the DSP

In this case, according Figure 3.3, the duty cycle for each inverter leg can be obtained from the calculated times.

$$\begin{bmatrix} 1 \\ d_x \\ d_y \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} D_1 \\ D_2 \\ D_3 \end{bmatrix}$$
$$\Downarrow$$
$$\begin{bmatrix} D_1 \\ D_2 \\ D_3 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & -1 & -1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ d_x \\ d_y \end{bmatrix} \tag{3.4}$$

In the same way similar matrices could be calculated for the rest of sectors.

3.2 PI Controller Design

The PMSM model equations 2.3 and 2.4 have been defined in Section 2.2. These equations may be represented in terms of the Laplace transform variable s as shown in Equation 3.5 and 3.6.

$$u_d(s) = (sL_d + R)i_d(s) - \omega L_q i_q(s) \tag{3.5}$$
$$u_q(s) = (sL_q + R)i_q(s) + \omega(L_d i_d(s) + \lambda_{PM}) \tag{3.6}$$

In the previous voltage motor equations, the second term in each of them is the back-EMF term induced by the opposite axis currents. Due to this term, the dq current control loops are not independent. In order to overcome this difficulty, a back-EMF decoupling term is added into the control loop as shown in Figure 3.4 for the case of i_q.

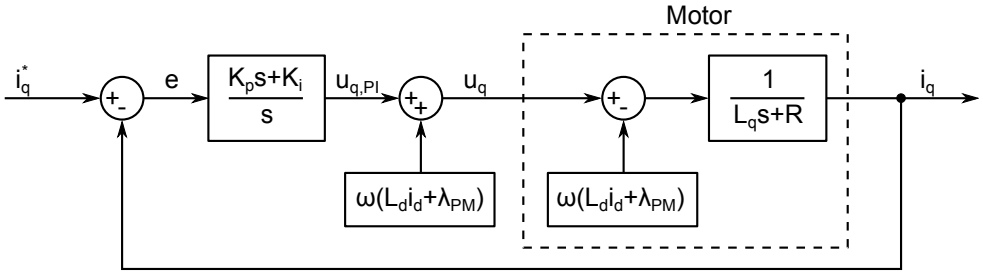


Figure 3.4. i_q current loop with an external disturbance and PI controller

This term is introduced after the PI controller as a disturbance with the same value but opposite sign of the back-EMF term included in the motor model. Therefore, the system can be simplified to a 2nd order system as shown in Figure 3.5 where i_q and i_d can be controlled independently.

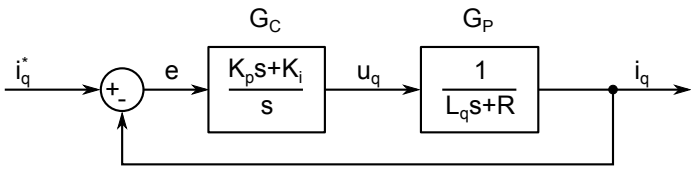


Figure 3.5. Simplified i_q current loop

It can be see that, for a non-salient pole machine where L_q=L_d, the current control loop in both axes would be identical. As shown in Table 2.1 this is the case for the used PMSM and the PI controller in both current loops are the same.

Time delay will occur in the system, caused by sensor delay, AD conversion and computation time. Therefore, in Figure 3.7 a 1st order time delay may be added to the control loop and this will increase the order of the whole system by one. The value of T_D is determined experimentally from the data presented in Figure 3.6. The delay time from the reference is applied by dSpace to the response is measured, is between 1-2 sampling periods of 0.2 ms. The reason for the uncertainty of the delay time is because the real reference could be applied at the beginning or at the end of a sampling time. Hence an average delay time for T_D=0.3 ms is chosen for this project.

If the PI controller is designed to have pole zero cancellation the system will behave as a 2nd order system again.

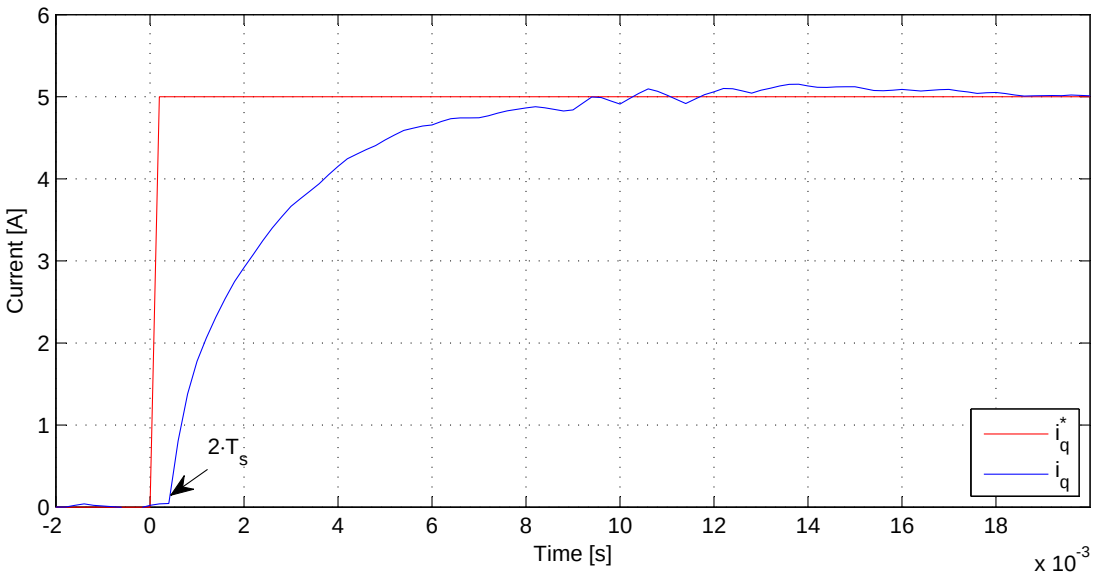


Figure 3.6. Measured current response with a 0.4 ms delay between the reference and the measure current

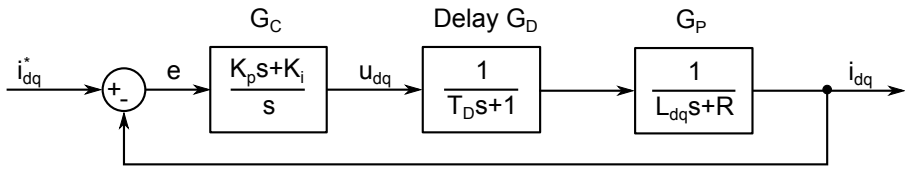


Figure 3.7. i_{dq} current loop with a 1. order delay added

The open loop transfer function is shown in Equation 3.7, where $K_i = K_p K_{ip}$. If $K_{ip} = \frac{R}{L}$, then the zero from the controller G_c will cancel the pole from the plant G_p reducing it to a 2nd order transfer function as shown in Equation 3.8.

$$G_{OL}(s) = \frac{K_p s + K_i}{s} \cdot \frac{1}{T_D s + 1} \cdot \frac{1}{Ls + R} = \frac{K_p}{T_D L} \cdot \frac{s + K_{ip}}{s} \cdot \frac{1}{s + \frac{1}{T_D}} \cdot \frac{1}{s + \frac{R}{L}} \tag{3.7}$$

$$G_{OL}(s) = \frac{K_p}{T_D L} \cdot \frac{1}{s\left(s + \frac{1}{T_D}\right)} \tag{3.8}$$

From the open loop transfer function 3.8, the closed loop transfer function is derived, as show in Equation 3.9, where $K = \frac{K_p}{T_D L}$. When comparing to the general equation for a 2nd order system as shown in 3.10, the natural frequency ω_n becomes $\sqrt{K}$ and the damping ratio ζ becomes $\frac{1}{2T_D\sqrt{K}}$.

$$G_{CL}(s) = \frac{G_{OL}(s)}{1 + G_{OL}(s)} = \frac{K}{s^2 + \frac{1}{T_D}s + K} \tag{3.9}$$

Where the general equation for a 2nd order system is:

$$H(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{3.10}$$

The K_p and the K_i values for the PI controller with pole zero cancellation in the current loop, can be calculated according to equation 3.11 and 3.12 respectively. These equations are valid for both the d- and q-axis current loops. With a desired overshoot of 2%, ζ have to be 0.78 according to Equation 3.13. Using the inductance and resistance value from Table 2.1 and the time delay of 0.3 ms, $K_p = 2.88$ and $K_i = 278$ are found.

$$K_p = \frac{L}{4\zeta^2 T_D} \tag{3.11}$$

$$K_i = K_p K_{ip} = K_p \frac{R}{L} \tag{3.12}$$

$$\zeta = \frac{\ln\left(\frac{1}{OS}\right)}{\sqrt{\ln\left(\frac{1}{OS}\right)^2 + \pi^2}} \tag{3.13}$$

The closed loop step response of the full system with time delay included, is illustrated in Figure 3.8. A 2% overshoot, as designed can be seen and a settling time of 1.69 ms is observed.

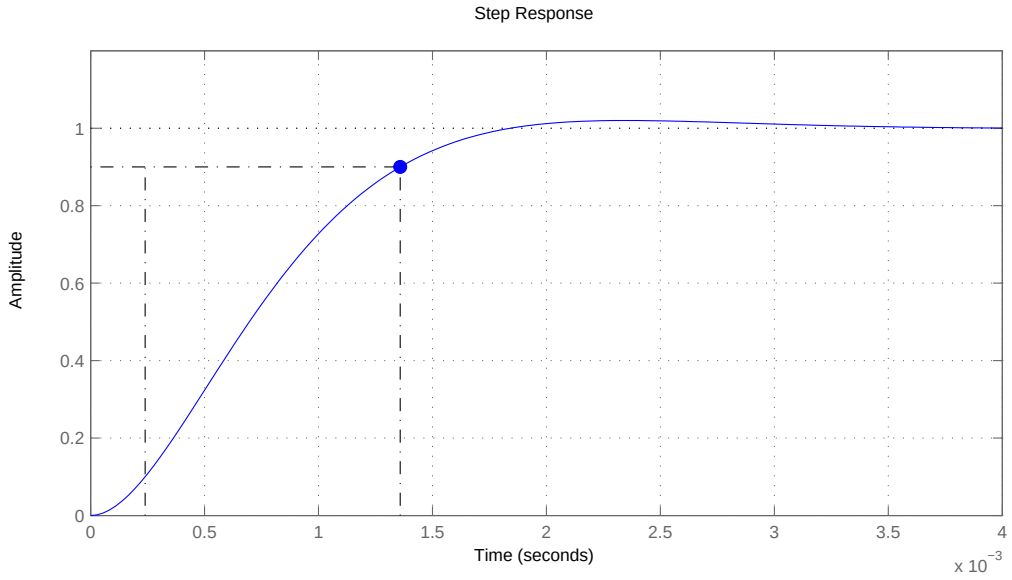


Figure 3.8. Closed loop step response of the system with time delay included

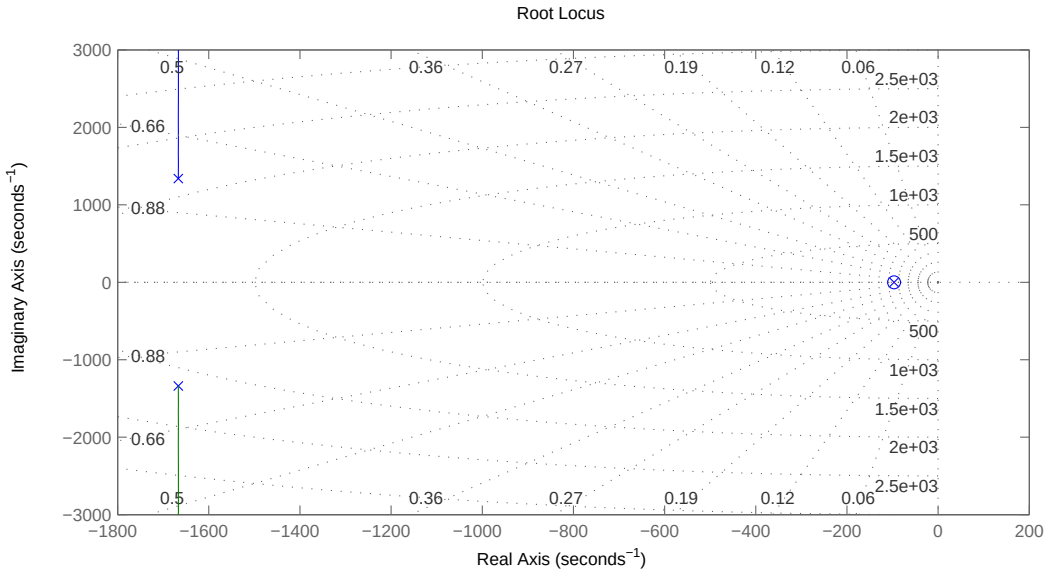


Figure 3.9. Closed loop root locus of the system with time delay included

From the root locus presented in Figure 3.9, it can be seen, that the pole zero cancellation is achieved with the designed PI controller. It should also be noticed that the complex pole gives a damping of 0.78 as designed.

A method to design the PI controller for an ideal system has be described. However, in real application, inverter nonlinearities, such as dead-time, conduction losses and threshold voltages of the power devices, will cause output voltage error. Therefore, when implementing these ideally designed K_p and K_i values in the laboratory, a different current response was observed compared with the simulation result as shown in Figure 3.10. The simulation shows a faster response which can be attributed to the voltage errors present in the experiment.

The measured current response from both dSpace and the oscilloscope is also compared in Figure 3.10. DSpace response fits exactly with the measurements from the oscilloscope, but it has less ripple due to a lower sampling frequency. Therefore the dSpace measurements are considered to be valid for all further experiments throughout the report. In cases where the current ripple has to be analysed, measurements from the oscilloscope will be used.

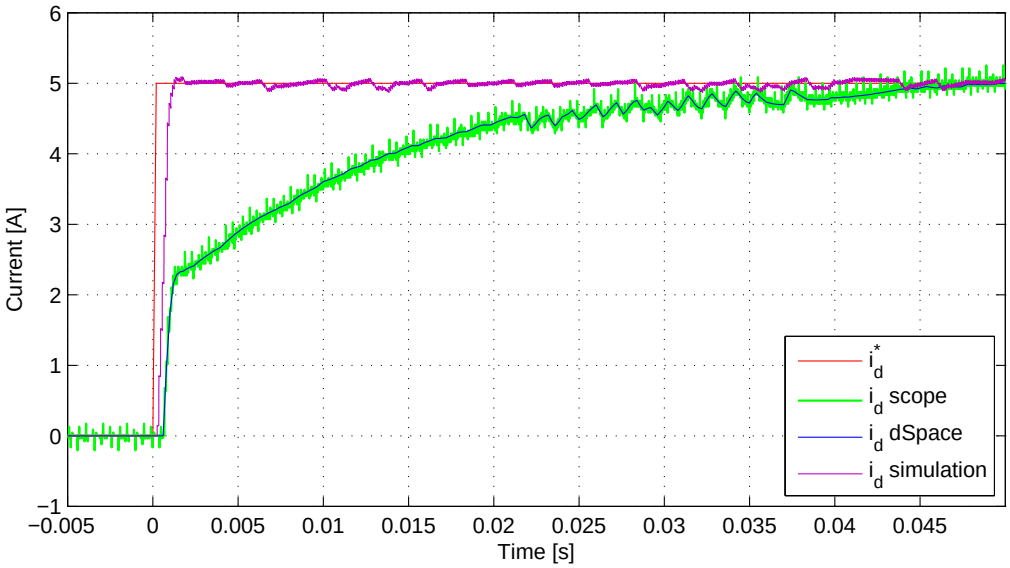


Figure 3.10. Simulation of an ideally system with a designed PI controller having $K_p = 2.88$ and $K_i = 278$, compared with the same controller implemented in the experimental setup

To improve model accuracy, voltage errors need to be implemented in the inverter design. The dead-time could cause the largest voltage error in the inverter, if the phase currents are low. Therefore, the effects of dead-time have been included in an updated model, made in PLECS Simulink. The comparative results are shown in Figure 3.11. Both the simulated and experimental step response is obtained with a PI controller having a $K_p = 2.88$ and a $K_i = 278$. From Figure 3.11 it is observed that the simulation fits well with the experimental result from dSpace. The current rises fast until it reaches 2.25 A where the curve break, and after that the current is increasing slow. This occurs because the controller is designed for an ideal system without any dead-time.

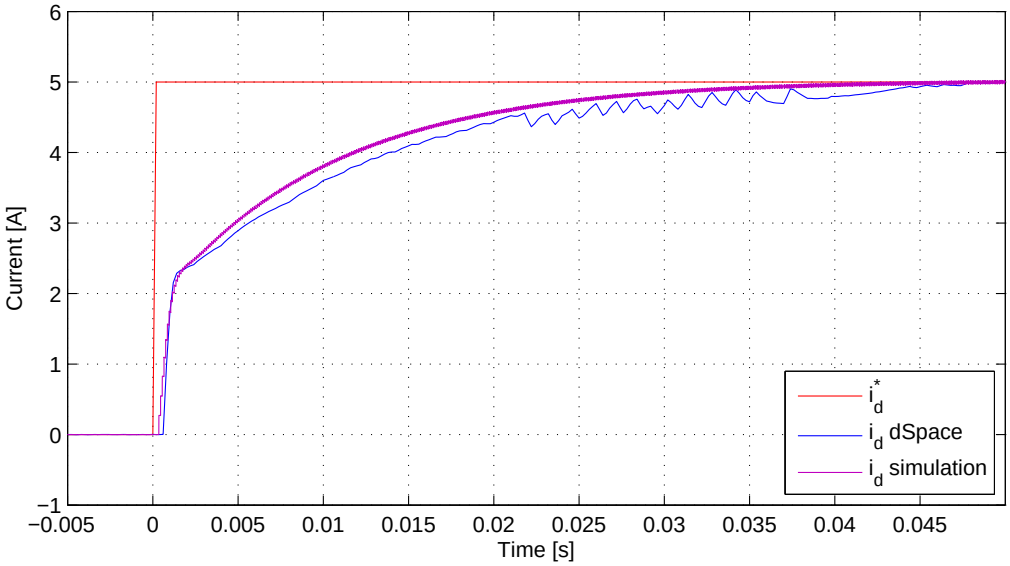


Figure 3.11. Simulation of a system with inverter dead-time and a designed PI controller having $K_p = 2.88$ and $K_i = 278$, compared with the same controller implemented in the experimental setup

Analytical design of the PI controller for FOC is not the main goal of this project. Therefore, a new controller is designed experimentally to ensure the fastest and most adequate response. The tuning of K_p and K_i starts by setting $K_i = 0$ and increasing K_p from 0 to a value where oscillations start. Then, K_p is kept constant and K_i is adjusted to obtain a minimised overshoot, under 2%. By using this method a new set of paremeter for the PI controller was determined. The proportional gain was found to be K_p=2 and the integral gain was supose to be K_i=1000. The measured rise time t_r is 4.2 ms and it is defined as illustrate in Figure 3.12. This definition of rise time is used in the whole report. No measurable overshoot and steady state is present in Figure 3.12, as designed.

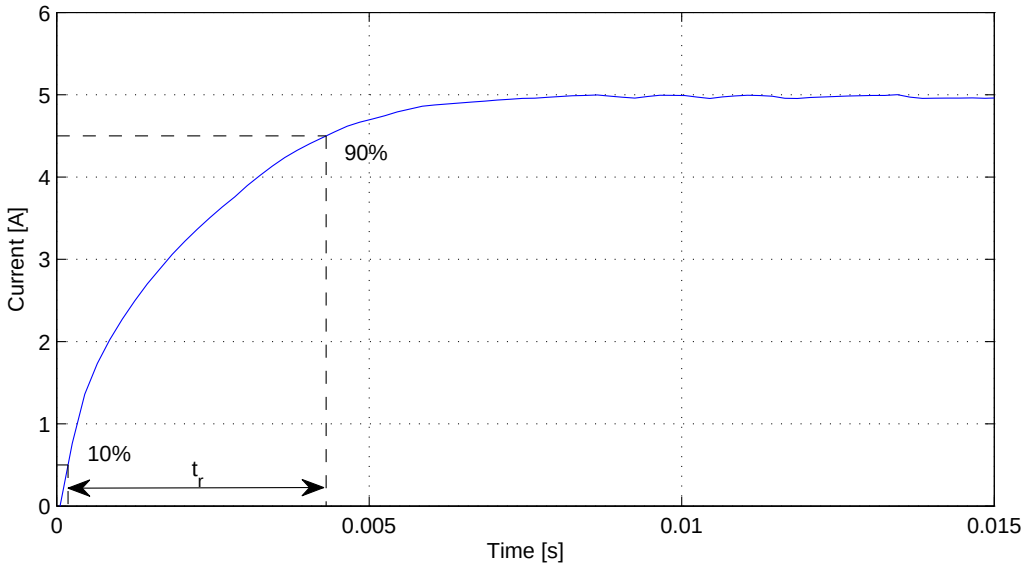


Figure 3.12. The measured d-axis current step response for the FOC with the PI parameters as follows: $K_p = 2$ and $K_i = 1000$

3.3 Simulation Results

In this section, the FOC simulation model is described and the result from the simulations is presented. An analysis of the simulated FOC performance characteristics regarding rise time, overshoot, ripple and steady state error is given at different operating conditions. This information serve as a comparison for the following section with the experiments implemented on the laboratory set-up, and in the following chapters with the proposed predictive controllers.

The simulations of the laboratory system with FOC are conducted in Matlab/Simulink with the following model:

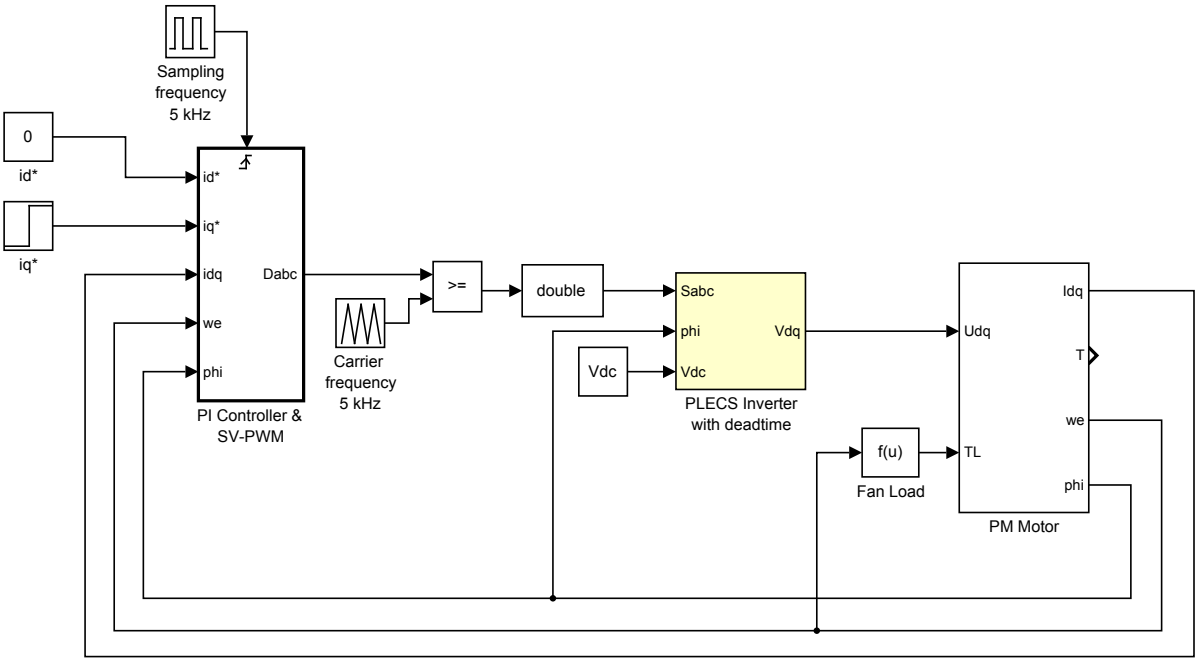


Figure 3.13. Simulink model for the FOC

In Figure 3.13 the first block is a triggered subsystem containing the two PI controllers and the SVM described earlier in this chapter. The PI uses the experimentally determined K_p = 2 and K_i = 1000 values and back EMF decoupling has been introduced. The triggered subsystem is supposed to emulate the discrete dSpace controller in the laboratory. According Figure 3.6, dSpace updates the actuation with one till two sampling delays, therefore, an unit delay was included in the model output. The triggering frequency (sampling frequency) of the subsystem is synchronized with the 5 kHz carrier frequency of the PWM generator. The original inverter block was based on Equation 2.24 described previously in this chapter. In order to include dead-time in the simulations, a new model of the inverter is created in PLECS, shown in Figure 3.14. The input for the inverter model is the switching function S and the output is the phase voltages expressed in the dq reference frame.

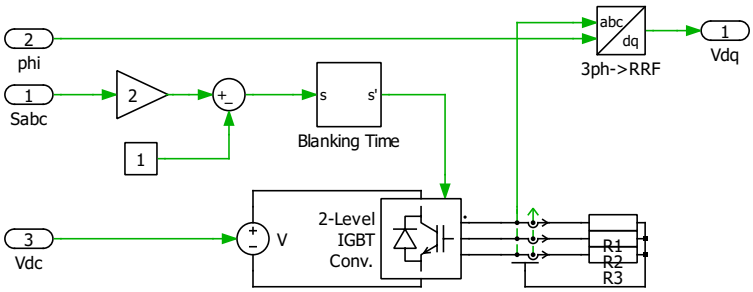


Figure 3.14. The Simulink/PLECS inverter model with dead-time implemented

The controller uses dq currents and therefore the PMSM model is directly expressed in dq reference frame so no further transformations have to be made for the feedback loop.

The full motor model is based on Equation 2.3, 2.4 and 2.12. Implementation of these equations in Simulink is illustrated in Figure 3.15. In the simulations, in order to facilitate the analysis on the current control, the load torque acting on the motor is a fan load, which offers a speed dependent load torque profile. So, the controller may focus on the current control performance only, and a fair comparison between different current control methods may be carried out For more detailed description of each model block, please refer to Appendix B.

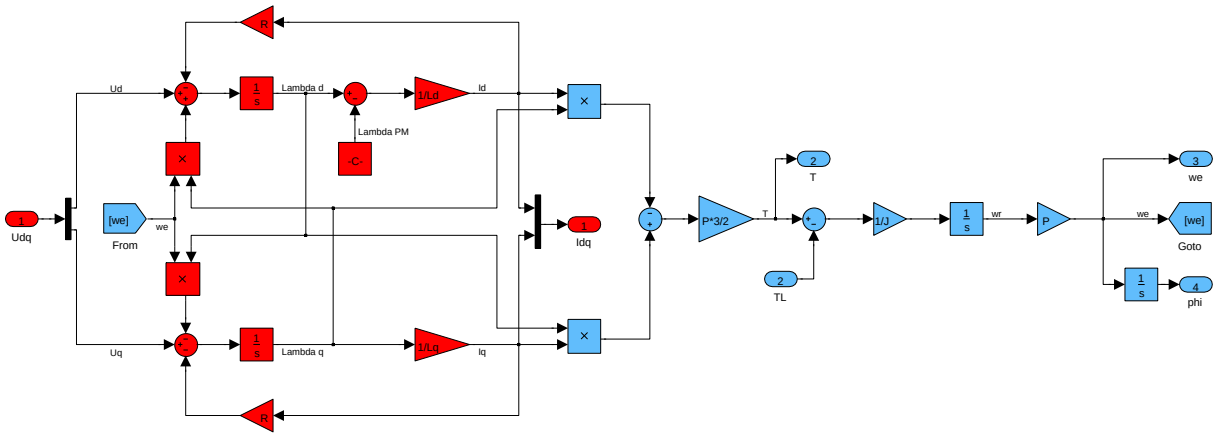


Figure 3.15. The Simulink motor model used for the simulations. The blue blocks are the mechanical part and the red are electrical.

In the following, the system response to different i_q^* steps are shown and analysed. Note that only the i_q and not i_d is displayed in the figures. This is because the torque response and ripple is directly proportional to i_q.

Start-up operation (zero speed) and 1000 rpm operation are both tested. The speed is determined by the fan load for the given i_q. Although, when the step happens the increase in i_q means an increase in the torque and therefore also an increase on the speed. It can be assumed that due to the inertia and the fast torque change than happens at the step instant, the speed will not change noticeably for the simulated time.

For characterising the transient, overshoot and rising time are measured. Where rising time is defined as the time taken from 10% to 90% of the step height as shown before in Figure 3.12. For the steady state, the error is presented as a percentage of the average current i_q and the current ripple is measured from peak to peak.

In Figure 3.16 an step from 0-5 A is made at zero speed. It can be observed that the simulated i_q current response has a rising time of 2.3 ms and an overshoot of 5%. This is over 2% which was the designed criteria in Section 3.2, however, the control parameters were designed experimentally and simulations uses ideal components with no losses. In steady state, the current ripple of i_q is only 64 mA with an small steady state error of 0.58%.

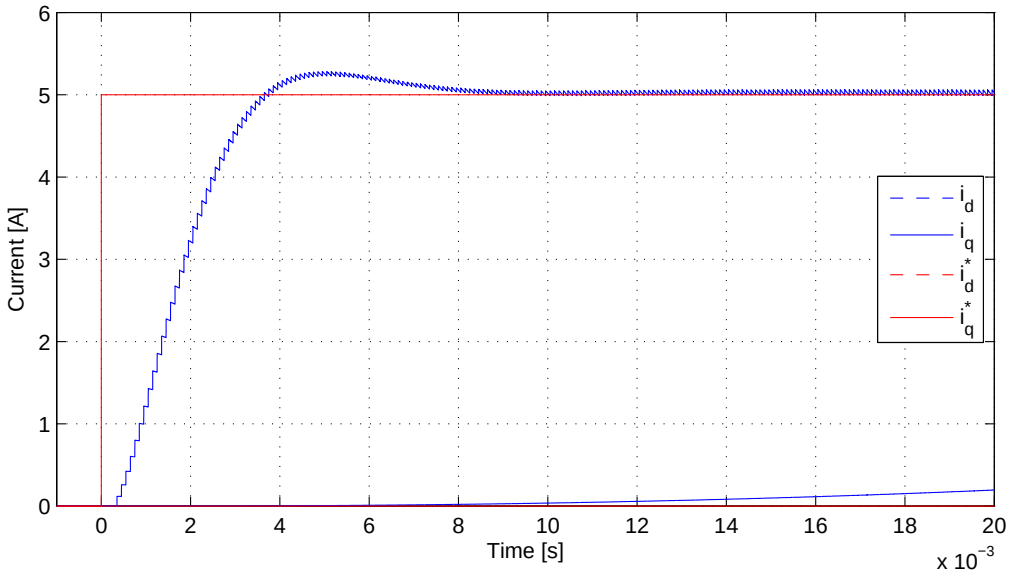


Figure 3.16. Motor start up current response starting from 0 rpm

When the speed starts to increase the ripple increases. This is because the back-EMF increases directly with the speed, and the controller increases the voltage reference to keep the i_q current constant. To produce a higher phase voltages, the SVM must produce wider switching pulses, hence increasing the current ripple. In 3.17, the motor is now running at 1000 rpm and the current ripple have increased to 2.04%. At t=0.015 s, an step current in i_q^* from 5 A to 10 A, it is applied.

The rising time is 0.95 ms, and a 36.8% overshoot occurs in Figure 3.17. The integrator part of the PI controller tries to minimise the steady state error, which in this case reaches 3.15%.

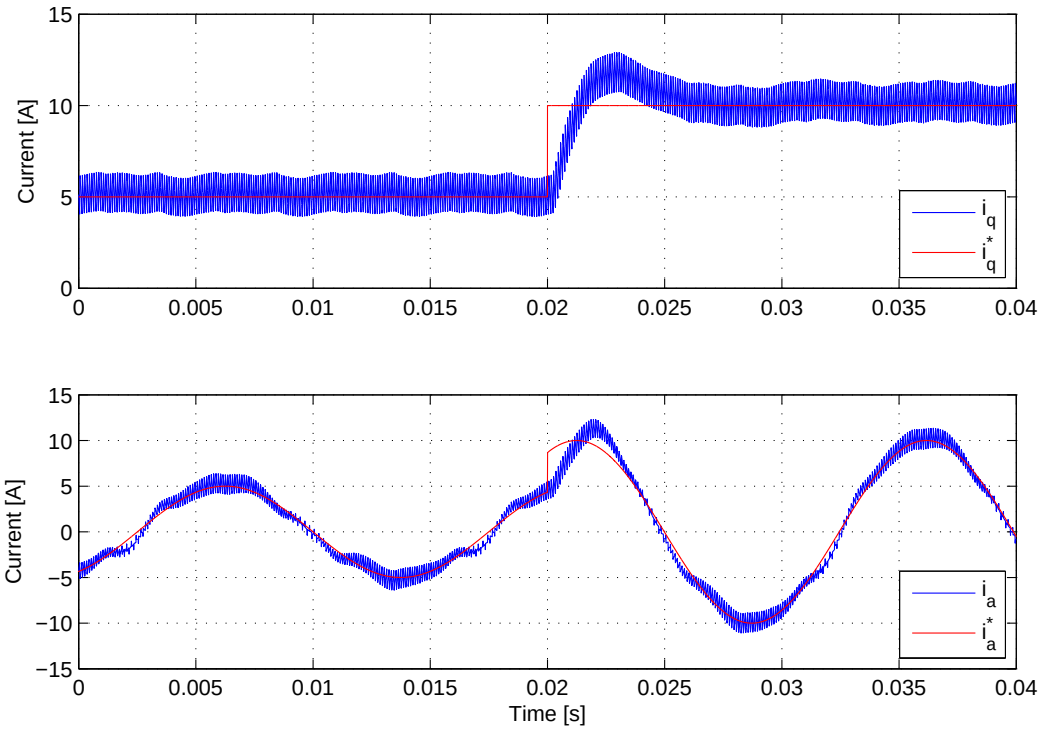


Figure 3.17. Simulated current step response from 5 A to 10 A at 1000 rpm. The red lines are the input reference for the controller

Figure 3.18 shows a motor rotational direction change, commanding a i_q^* current step from 5 A to -5 A, again at a constant speed of 1000 rpm. A similar transient response to the previous figure is measured, with a rising time of 0.96 ms and a overshoot of 33.2%. The steady state error is invariably 3.15% and the i_q current ripple is 2.09 A.

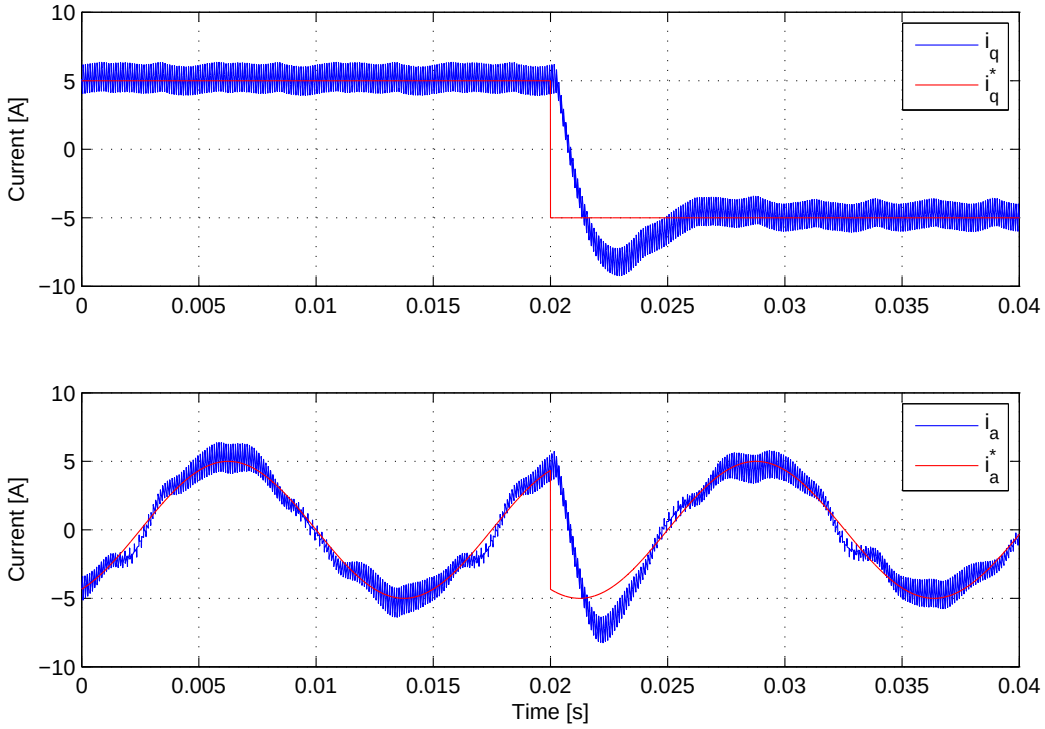


Figure 3.18. Simulated current step response from 5 A to -5 A at 1000 rpm. The red lines are the input reference for the controller

3.4 Experimental Results

Using the laboratory system described in Section 2.1, the PMSM is operated in current control mode using the following parameters for the PI controller, $K_p = 2$ and $K_i = 1000$. The IM is in speed control mode, running at a constant speed of 1000 rpm. Similar test to the ones described for the simulations in the previous section, are now conducted with the laboratory set-up and the presented data is captured with dSpace. Both reference, and current response are plotted for i_q and i_a.

The experimental result presented in Figure 3.19 is the response to an step from 0 to 5 A at 0 rpm. The rising time is measured as 5 ms. As designed, no overshoot is present. A small steady state error of 0.14% are measured.

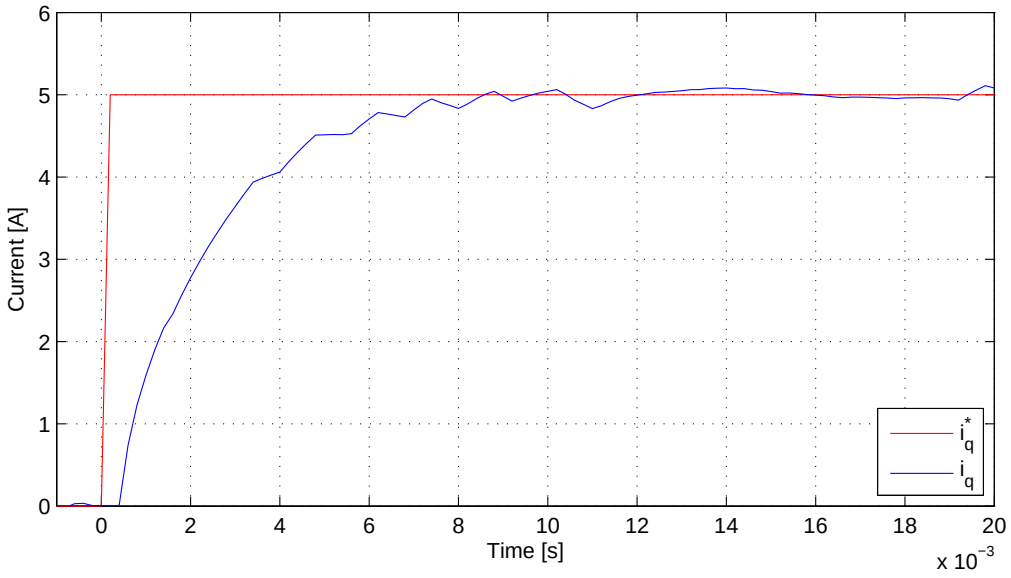


Figure 3.19. Measured current step response from 0 A to 5 A at 0 rpm

A new step was applied now from 5 A to 10 A at constant speed of 1000 rpm provided by the IM. The result is presented in Figure 3.20. The rising time is measured as 1 ms, very closed to the value of 0.96 ms obtained from the simulations. The overshoot is 19.6% and a steady state error before and after the step from 0.015 to 0.13%.

Since the sampling and the switching frequency is the same for the dSpace system, the measured currents will appear as a filtered signal with no switching ripple. Therefore, measurements of the current ripple, which can not be observed in the figures, were made with an oscilloscope. The result for the current ripple at 1000 rpm was measured to be 2.08 A, very close to the value of 2.04 A obtained from the simulations.

Finally, a step command from 5 A to -5 A is applied and the measured result is presented in Figure 3.21. The rising time is now 5 ms and where no measurable overshoot is occurring. The steady state error have increased a bit, measuring 0.61% before and 1.05% after the step command.

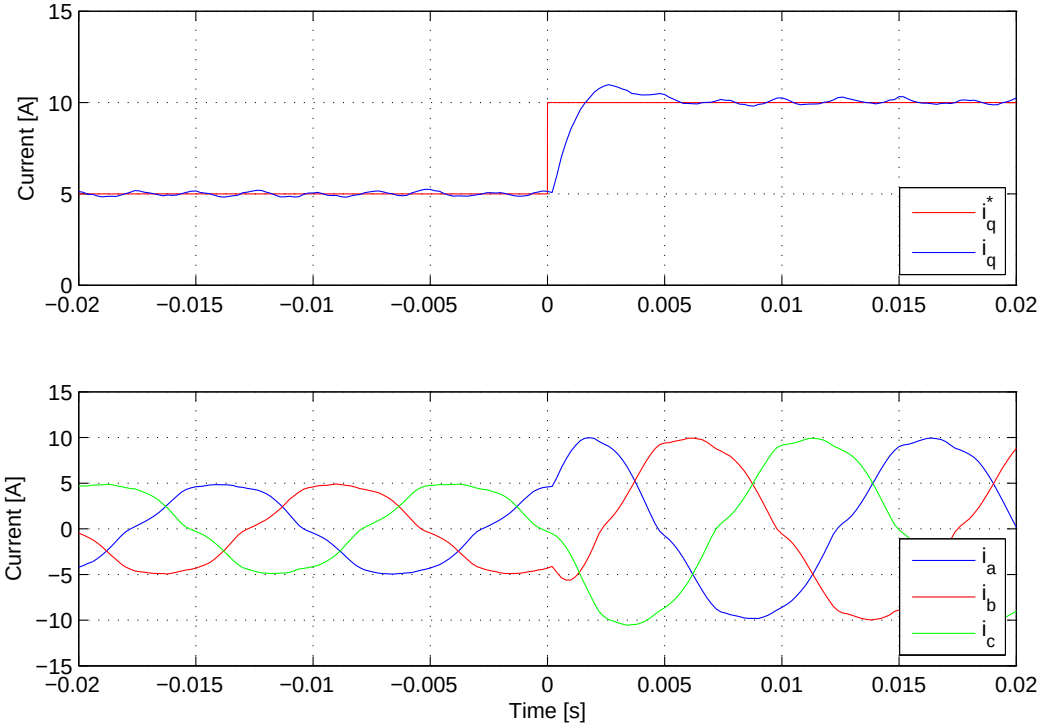


Figure 3.20. Measured current step response from 5 A to 10 A at 1000 rpm

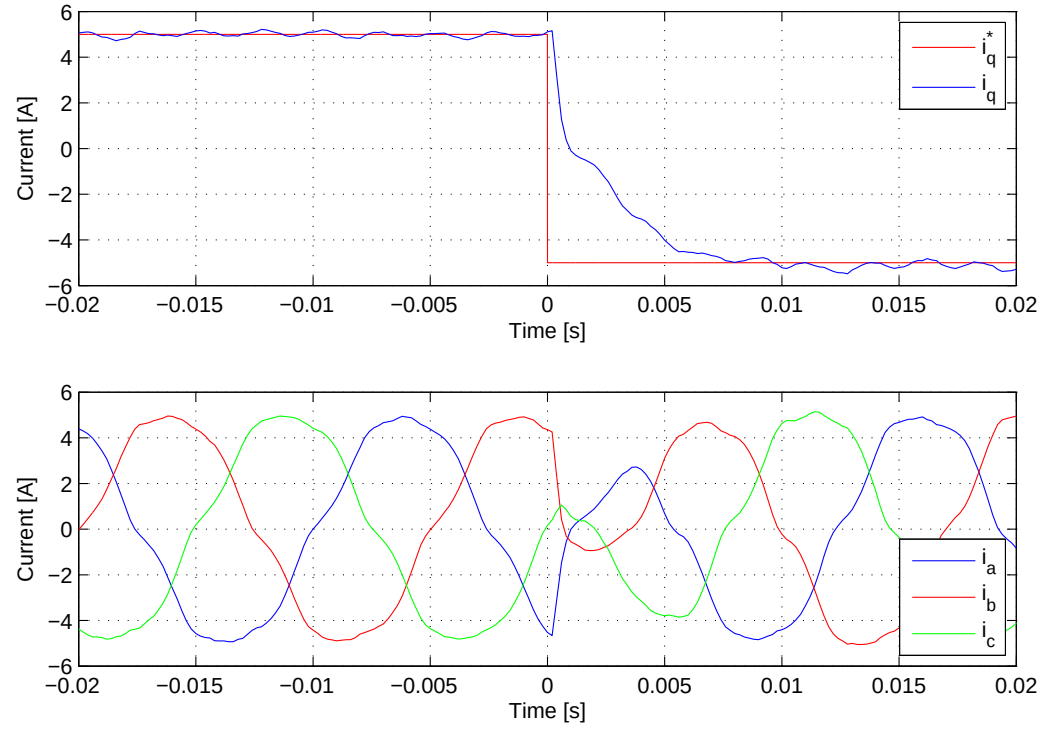


Figure 3.21. Measured current step response from 5 A to -5 A at 1000 rpm. The red line are the input reference for the controller

Deadbeat Control 4

In this chapter, deadbeat predictive control principle and implementation are described. As it will emerge from simulations and experiment, time delay in digital implementation and dead-time compensation were considered important for the performance of the controller. A separated section is used for analysing dead-time compensation. Simulations and experimental results are presented in the last two sections so that a fair comparison with a standard FOC can be performed.

Figure 4.1 shows the structure of the deadbeat controller. The deadbeat controller has replaced the PI used in the classical FOC scheme. As in FOC, a space vector modulation block (SV-PWM) converts the controller output voltage $u_{dq}(k)$ into duty cycles imposed on the inverter, these are indicated as S_a, S_b and S_c in Figure 4.1. This modulation ensures a fixed switching frequency, which is of importance in the case that a passive filter should be implemented.

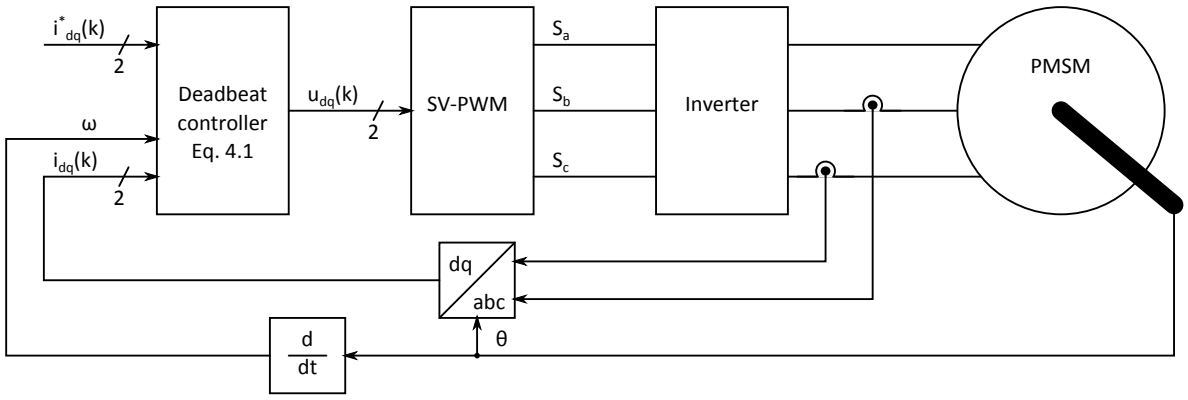


Figure 4.1. Principle of the deadbeat controller

Similar to other predictive controllers, deadbeat algorithm is based on the motor model in order to obtain the voltage reference, that should be applied in order to obtain the desired current at the end of the switching period, $\bar{i}_{dq}(k+1)$. Therefore, it is a straightforward application of the discretised motor Equation 2.21 from Section 2.2.1. The desired current is indicated by the reference, however $\bar{i}^*_{dq}(k+1)$ is non-available information at the current moment. If the current reference value $\bar{i}^*_{dq}(k)$ is set to be the predictive value to be reached

for next period $\bar{i}_{dq}(k+1)$. Then, Equation 2.21 can be rewritten like the following:

$$\begin{aligned}
&\bar{i}^*_{dq}(k) = \mathbf{A}(k)\bar{i}_{dq}(k) + \mathbf{B}\bar{u}_{dq}(k) + \mathbf{C}(k) \\
&\quad \Downarrow \\
&\bar{u}_{dq}(k) = \mathbf{B}^{-1}\left(\bar{i}^*_{dq}(k) - \mathbf{A}(k)\bar{i}_{dq}(k) - \mathbf{C}(k)\right) \qquad (4.1) \\
&\quad \Downarrow \\
&u_d(k) = \frac{L_d}{Ts}\left(i^*_d(k) - i_d(k)\right) + Ri_d(k) - \omega L_q i_q(k) \qquad (4.2) \\
&u_q(k) = \frac{L_q}{Ts}\left(i^*_q(k) - i_q(k)\right) + Ri_q(k) + \omega\left(L_d i_d(k) + \lambda_{PM}\right) \qquad (4.3)
\end{aligned}$$

The matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ have previously been defined in Section 2.2.1.

Speed, ω, and dq current measurements, $\bar{i}_{dq}(k)$ are derived from the rotor angle acquired by the encoder and the phases current measurement respectively, at every sampling period.

Due to the computation time, sensors and actuation propagation time, some delay is introduced in the system. Delay compensation can also be included in deadbeat scheme. In the conventional scheme just explained, the computation time is considered as zero, so that the voltage is applied just in the moment when the current is sampled, in instant (k) as shown in Figure 4.2

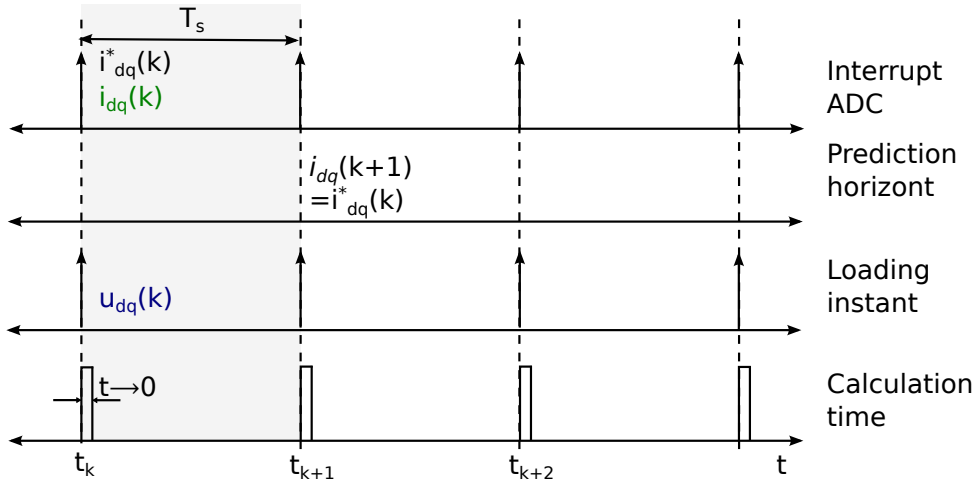


Figure 4.2. Ideal timing sequence of deadbeat operation, (green) measurements, (blue) calculated values

However, in real implementation, the voltage reference calculation requires some time for being computed. In practical implementation, the controller updates its output once every sampling period, so that the computed value will be loaded at the end of the sampling period, or multiples of it.

If $u_{dq}(k)$ is ideally applied at instant (k) the desired current $i_{dq}(k+1)$ should be obtained directly after one sampling time. Taking into account the real operation when the computation time is non zero, and introducing one sampling delay in the voltage

application, the current will be updated as in Figure 4.3, where the blue arrows are the ideal actuation. However, the values are applied with one sampling delay, so that the actual current is obtained as the black line in the figure.

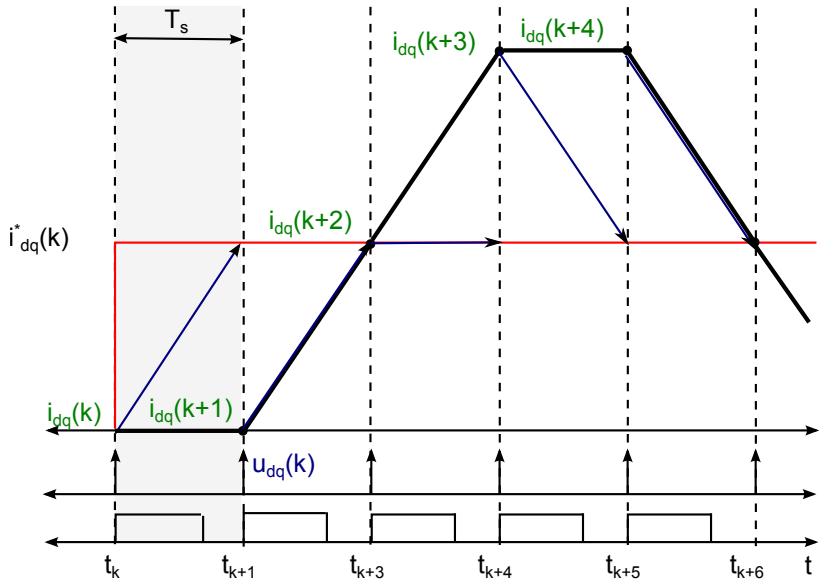


Figure 4.3. dq- current in a delayed system, (green) measurements, (blue) predicted actuation

Therefore, in order to avoid the shown oscillations and overshoot due to this delay in the actuation, the prediction horizon can be extended one more sampling time to instant $(k+2)$ as suggested in [Moon et al., 2003].

As shown in Figure 4.4 the voltage $u_{dq}(k+1)$ to be applied next period is calculated in the current period so that the desired current in instant $(k+2)$ can be obtained satisfactory:

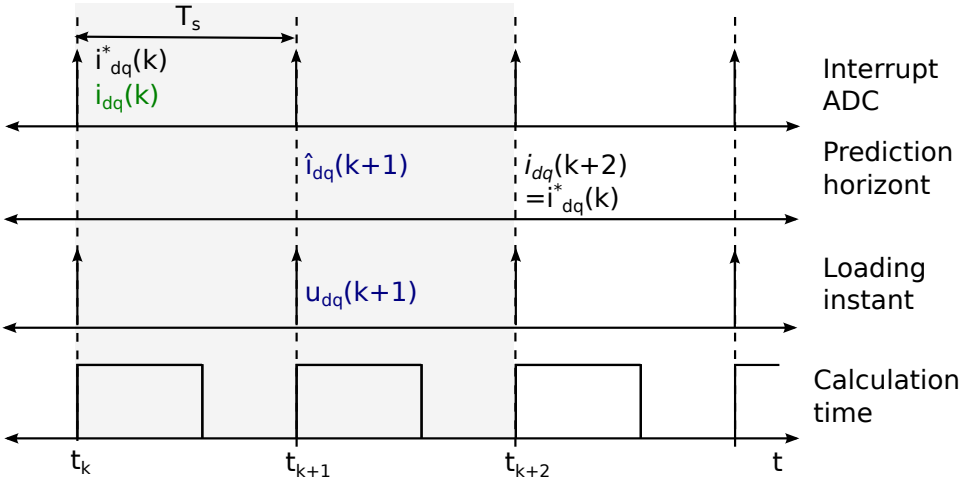


Figure 4.4. Timing sequence of the deadbeat operation with delay compensation, (green) measurements, (blue) calculated values

The current $\bar{i}_{dq}(k)$ are measured, while the dq voltage components at instant k are known, this is because $\bar{u}_{dq}(k)$ is the voltage applied at the beginning of the period as result of the

prediction computation made in the previous sampling period. With this information, the current $\bar{\hat{i}}_{dq}(k+1)$ at the beginning of next period can be estimated as in 4.4 .

$$\bar{\hat{i}}_{dq}(k+1) = \mathbf{A}(k)\bar{i}_{dq}(k) + \mathbf{B}\bar{u}_{dq}(k) + \mathbf{C}(k) \tag{4.4}$$

Based in the estimated current, $u_{dq}(k+1)$ is calculated in Equation4.5 , so that in t_{k+2} the real current $\bar{i}_{dq}(k+2)$ should be the expected current, making $\bar{i}_{dq}(k+2) = \bar{i}^*_{dq}$.

$$\bar{u}_{dq}(k+1) = \mathbf{B}^{-1}\left(\bar{i}^*_{dq}(k) - \mathbf{A}(k)\bar{\hat{i}}_{dq}(k+1) - \mathbf{C}(k)\right) \tag{4.5}$$

$\bar{u}_{dq}(k+1)$ is the voltage that should be applied at the beginning of the next sampling period, at instant $(k+1)$, so that the desired current in $(k+2)$ is obtained.

The same principle that has just been explained will be used in Chapter 5.2 for implementing delay compensation in the particular case of MPC, where the delay has large influence in the current ripple.

As shown, with delay compensation, a deadbeat controller can give a good dynamic performance and high bandwidth of the current controller compared with the FOC. A drawback of this control method is its sensibility to parameter inaccuracy and voltage error due to switching non-linearities and dead-time from the inverter, which give steady state errors. The model parameter sensibility is studied and compared with the other control schemes in Chapter 6. The next section describes the dead-time consequences together with a compensation method.

4.1 Inverter dead-time compensation

The inverter non-linearities as the voltage drop in switches and diodes, the turn on and turn off time or the added dead time result in a distorted voltage output. That distortion would depend on the hardware. The use of dead-time is needed in order to avoid the shoot through phenomena, that is short circuiting the inverter legs. In this section, voltage error compensation is developed.

Dead-time can seriously affect the voltage output, therefore several dead-time compensation methods has been treated in the literature[Sukegawa et al., 1991], [Munoz and Lipo, 1999] and [Hwang and Kim, 2010] for vector controlled motor drives.

Other references which implements dead-time compensation for PMSM are, adaptive compensation in [Urasaki et al., 2005] or the more recent compensation technique presented in [Kim et al., 2010], where a disturbance estimator determine the voltage error cause by the non-linearities from the inverter, without any current polarity detection.

The compensation may provide correction in the fundamental and harmonic components of the voltage. It consists of feed-forwarding a vector correction of the voltage error. The volts per second lost over an entire cycle is calculated and added to the voltage reference.

The effect of the inverter non-linearities is analysed in a phase leg of the PWM inverter in Figure 4.5

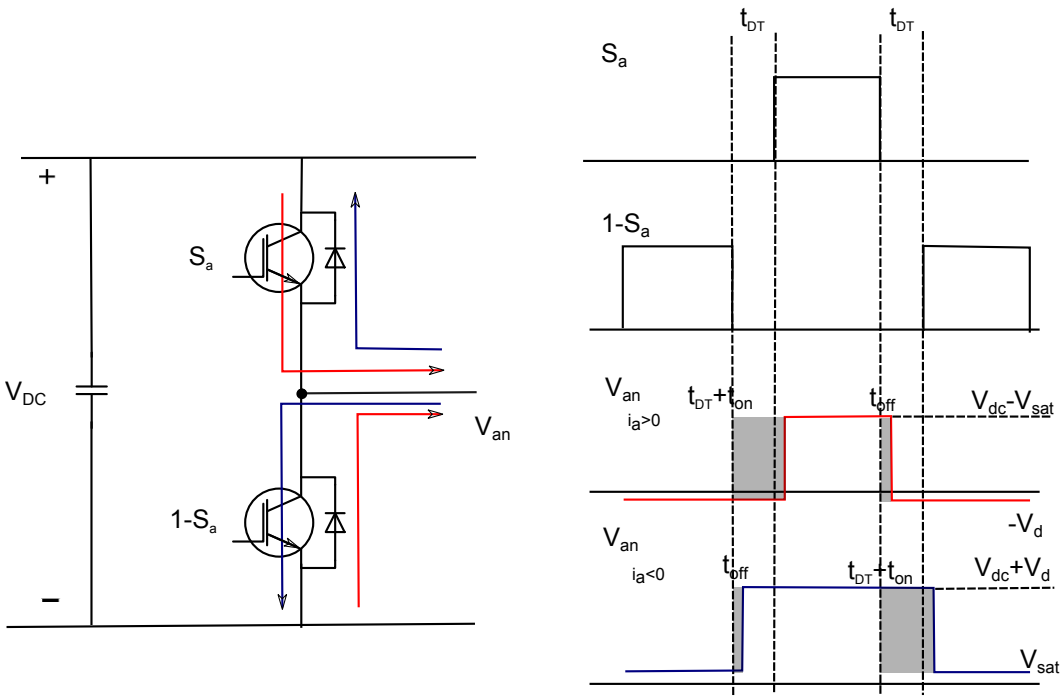


Figure 4.5. Relationship phase current sign vs. rotor angle in the static reference frame

The voltage variation depends on the dead time to switching period ratio, adding the voltage drop in transistors, V_{sat}, and diodes, V_d. The average voltage distortion added by dead-time can be calculated as:

$$\Delta V = \frac{t_{DT} + t_{on} - t_{off}}{T_s} (V_{dc} - V_{sat} + V_d) + \frac{V_{sat} + V_d}{2} \tag{4.6}$$

Since t_{on}, t_{off} and R are very much dependent on the operation point and the hardware used. In the literature [Hwang and Kim, 2010] different methods for making an on-line estimation of ΔV for a FOC control are presented. The dead-time t_{DT} is however a predetermined value, decided by the software or hardware. In the current case, the dead-time is implemented in the hardware and chosen as $2.5\mu s$. The biggest part of the voltage error is due to dead-time, so in this case, the voltage drop in the transistors and diodes are neglected and therefore simplifying Equation 4.6 to:

$$\Delta V = \frac{t_d \cdot V_{dc}}{T_s} \tag{4.7}$$

According to [Hwang and Kim, 2010], the averaged distorted voltage of the three phases can be expressed as ΔV and dependent on the direction of each phase current. Using Fourier series, the distorted voltage can be expressed in $\alpha - \beta$ stationary reference frame. Since the deadbeat controller is implemented in the d-q rotating reference frame, the distorted voltage is transformed as well, leading to Equations 4.8 [Hwang and Kim, 2010].

$$\begin{aligned} u_d^{DT} &= \frac{4}{\pi}\Delta V\left(\frac{12}{35}sin6\omega t + \frac{24}{143}sin12\omega t + ...\right) \\ u_q^{DT} &= \frac{4}{\pi}\Delta V\left(-1 + \frac{2}{35}sin6\omega t + \frac{2}{143}sin12\omega t + ...\right) \end{aligned} \tag{4.8}$$

As observed in 4.9 These voltage components u_d^{DT} and u_q^{DT} are subtracted from the uncompensated voltage when a positive reference i_q^* is commanded. Otherwise with a negative i_q^* command, u_{dq}^{DT} is added or subtracted depending the speed direction.

$$\begin{aligned} \bar{u}_{dq} &= \bar{u}_{dq} - \bar{u}_{dq}^{DT} \qquad \text{if } i_q^* > 0 \\ \bar{u}_{dq} &= \bar{u}_{dq} + \bar{u}_{dq}^{DT} \qquad \text{if } i_q^* < 0 \text{ and } \omega < 0 \\ \bar{u}_{dq} &= \bar{u}_{dq} - \bar{u}_{dq}^{DT} \qquad \text{if } i_q^* < 0 \text{ and } \omega > 0 \end{aligned} \tag{4.9}$$

The corrected voltage is the controller output for the SVM block.

4.2 Simulations Results

Simulations of the deadbeat controller are performed using the same Simulink model with dead-time implementation in PLECS as the one used for FOC in Section 3.3, with the exception of replacing the PI controller with the deadbeat controller. Again, a sampling frequency of 5 kHz was used. The controller is implemented using a Matlab function block, which produces a dq output voltage $\bar{u}_{dq}$ for the SVM.

Throughout this section, similar tests and analysis to the one performed for FOC are carry out. To clarify the design process so it is properly understood, in the simulations follow some steps. First the system is implemented with dead-time included in the inverter implementation but not calculation delay is taken into account. The response to this system with and without dead-time compensation is compared. Then, with a dead-time compensated controller, calculation delay is added. The results are then compared adding delay compensation . All the simulations for comparison of the controllers with and without compensation are tested for a step from 0-5 A at 0 rpm.

Further testing at a constant speed of 1000 rpm were made with the complete implementation including both inverter dead-time and calculation delay compensation.

The Matlab function code for the final design, with all the implemented compensations, can be found in Appendix C.

In Figure 4.6, an uncompensated controller with dead-time in the inverter but no implemented system delay is tested at 0 rpm, with a step command from 0 to 5 A. It is seen, that the deadbeat controller provides low current ripple of 48 mA.

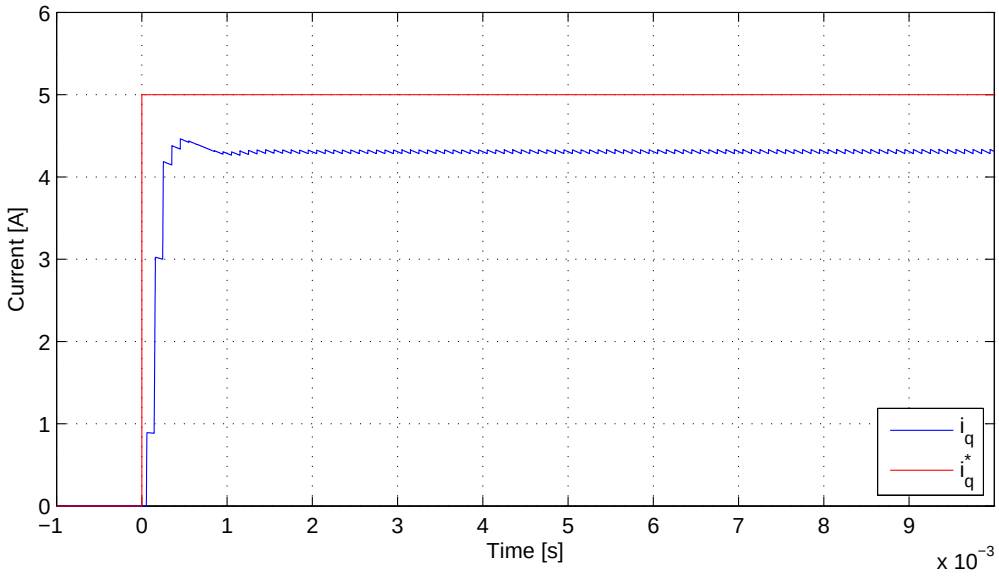


Figure 4.6. Simulated current response with dead-time from a non-linear inverter

The rising time is 0.194 ms, approximately one sampling time, T_s, as it can be seen in Figure 4.6, this time is constant for every test. Compared with the FOC from Figure 3.16 with a rising time of 2.3 ms, the deadbeat controller has five times faster dynamic response. However, deadbeat suffers from steady state errors due to the inverter non-linearities introduced by the switching and dead-time. The steady state error is measured as 24.4 % and no overshoot appears. According this results the steady-state error is a big drawback compared with FOC, therefore, the inverter dead-time compensation scheme described in Section 4.1 is implemented and the simulation results are shown in Figure 4.7. The reason why no dead-time compensation was necessary in FOC is because, if a voltage error occur, it also produces a current error, which again will be corrected by the PI controller. This is not the case for the deadbeat controller since it is only based on the discrete model equations and contains no integrator.

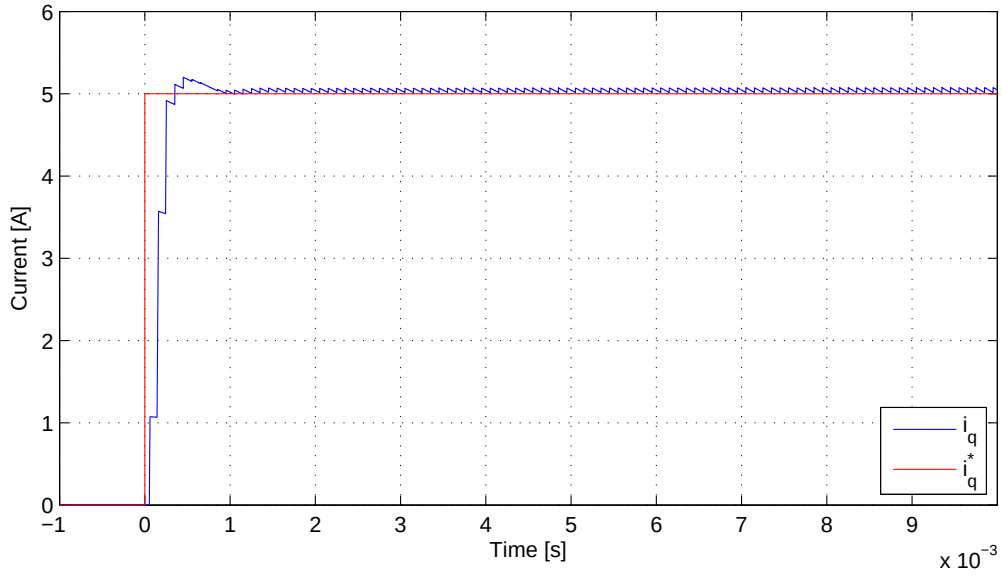


Figure 4.7. Simulated current response with dead-time compensation for a nonlinear inverter

With the dead-time compensation the steady state error is greatly reduced to a value of 0.87%. The current ripple is now 55 mA. An Overshoot of 3.8% can be measured now.

Now that the dead-time has been compensated satisfactory. It is known that the controller loads the actuation after each switching period. This calculation delay is introduced in the simulation model. In the simulations result, in Figure 4.8, an initial overshoot of 98.9% appears. The steady state error is very similar to the previous with only 0.96%, but the ripple is increased to 321 mA.

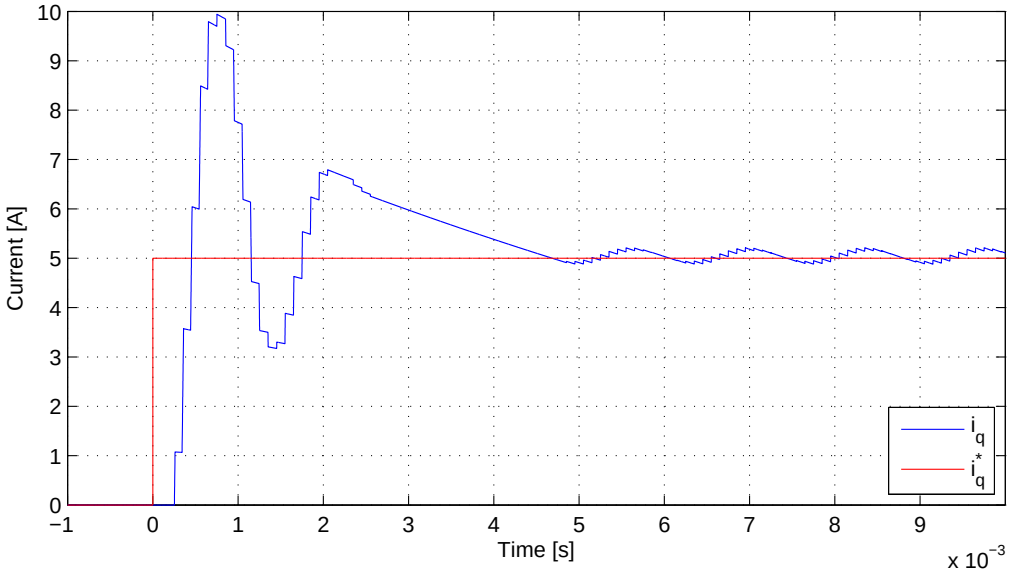


Figure 4.8. Simulated current response with dead-time compensation and actuation delay uncompensated

In order to remove such overshoot, the delay compensation explained previously in this chapter is added together with the dead-time compensation. The result is shown in Figure 4.9 where the overshoot is reduced to a value of 3.1%. As mentioned before, the rising time is 0.19 ms.. The steady state error is slightly bigger, 1.18%.

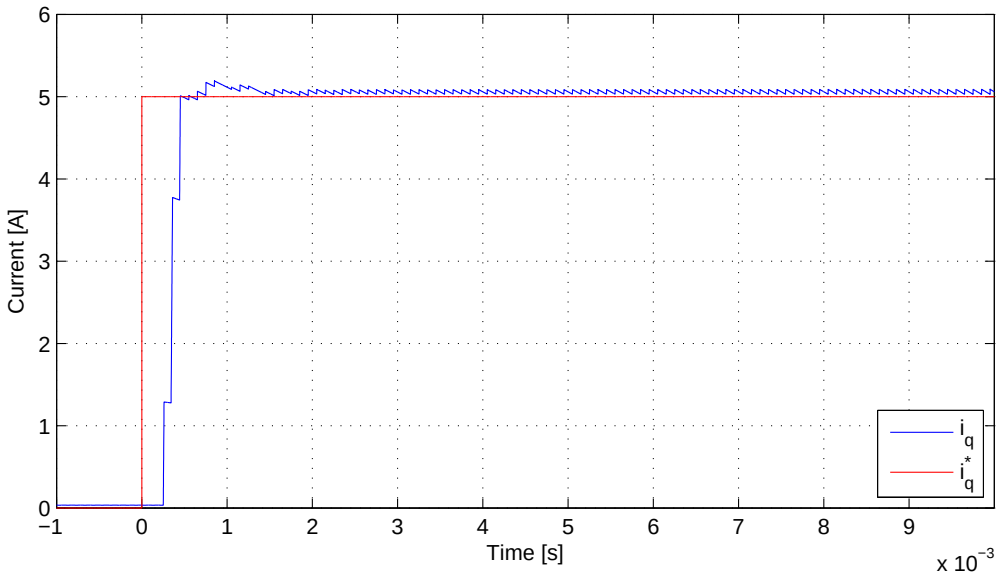


Figure 4.9. Simulated current response with dead-time and delay compensation

The response is now acceptable for steady state and with a improved transient response with respect the FOC at zero speed, however, the steady state error is still better for FOC. The complete system is now tested when the motor is running at a constant speed of 1000 rpm.

Figure 4.10 shows a step in the i_q current from 5 A to 10 A. The rising time is now 0.19 ms. The overshoot has increased to 8.9 %. The steady state error is 9.08 % and the i_q current ripple is 2.14 A. The Ripple for deadbeat is 10 mA higher than the ripple for FOC even though both use the same SVM technique.

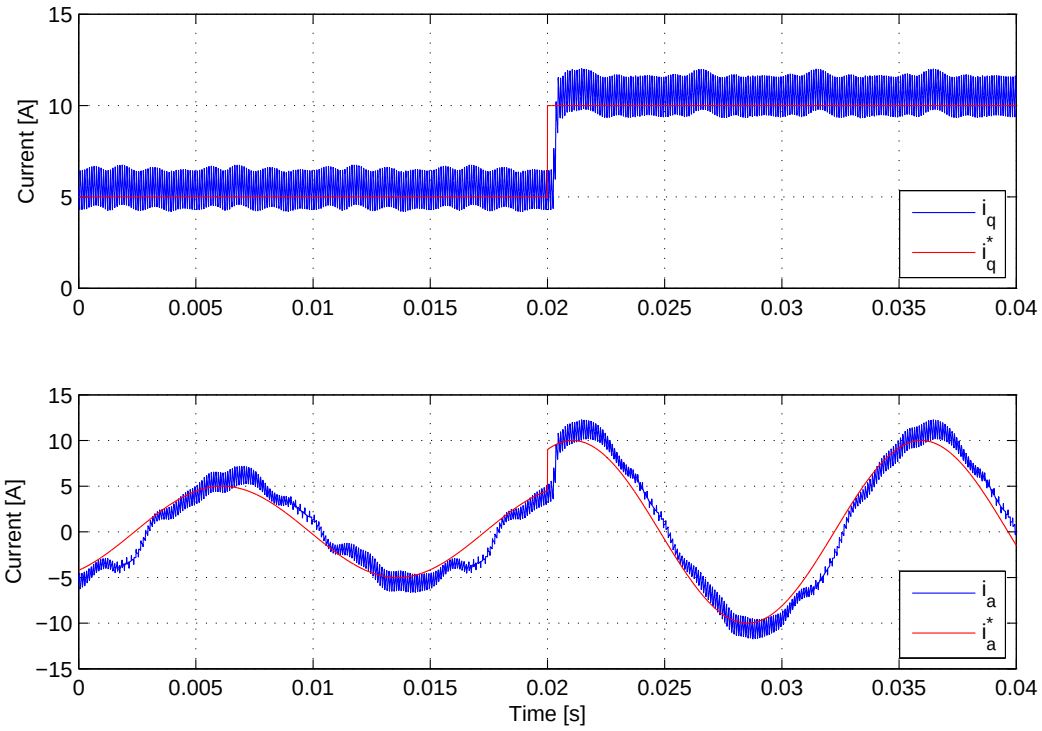


Figure 4.10. Simulated current step response from 5 A to 10 A. The red lines are the input reference for the controller

In Figure 4.11 a step from 5 A to -5 A was commanded. The steady state behaviour is exactly as in previous case. The overshoot has been removed and the rising time is 0.27 ms. Even with a negative current reference the dead-time compensation is working as expected.

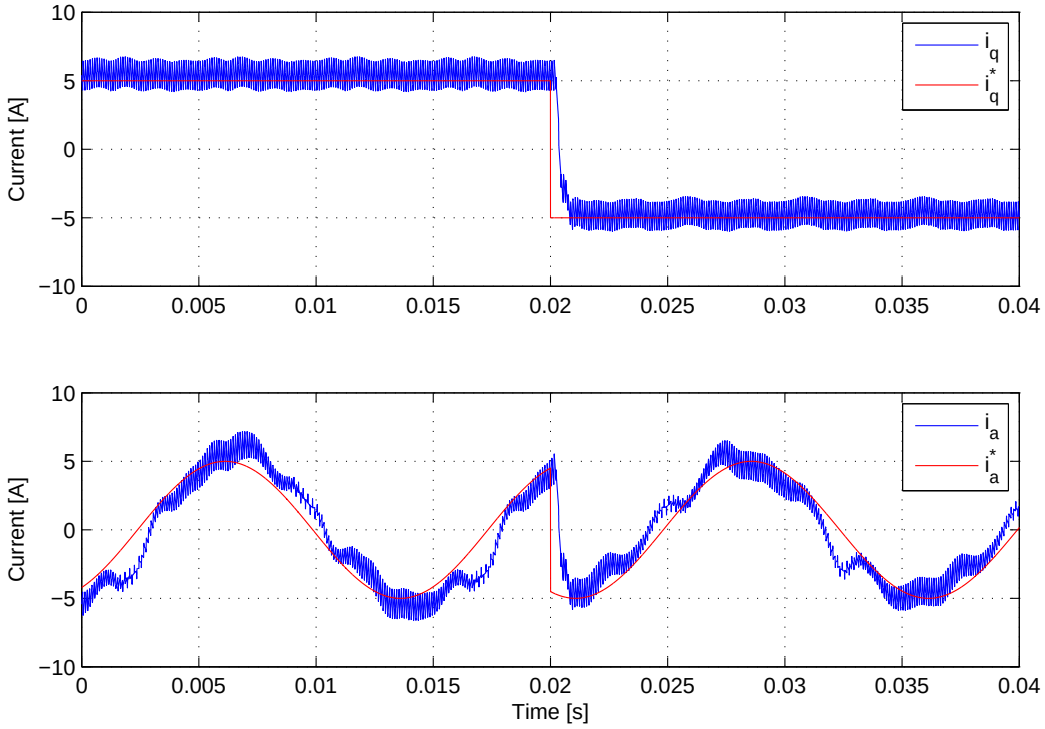


Figure 4.11. Simulated current step response from 5 A to -5 A. The red lines are the input reference for the controller

4.3 Experimental Results

For testing the deadbeat controller in the laboratory, the same system set-up and configuration for the FOC is employed. A description of the hardware and system set-up is found in Section 2.1. The controller code used includes both dead-time and delay compensation and it can be found at Appendix C . For testing the current response, the PMSM is operated in current control mode and the IM in speed control mode.

The experimental result in Figure 4.12 is the result to an 0-5 A step, at 0 rpm. The rising time is obtained as 0.2 ms precisely a sampling period and no overshoot. These two characteristics will be common for every test of deadbeat. The steady state error is measured as 8.33%, a higher value than the 1.18% in the simulations. The difference may be due to the voltage error in transistors and diodes, which has been dismissed. Therefore, the deadbeat controller still shows a better i_q current transient compared with the FOC in Figure 3.16

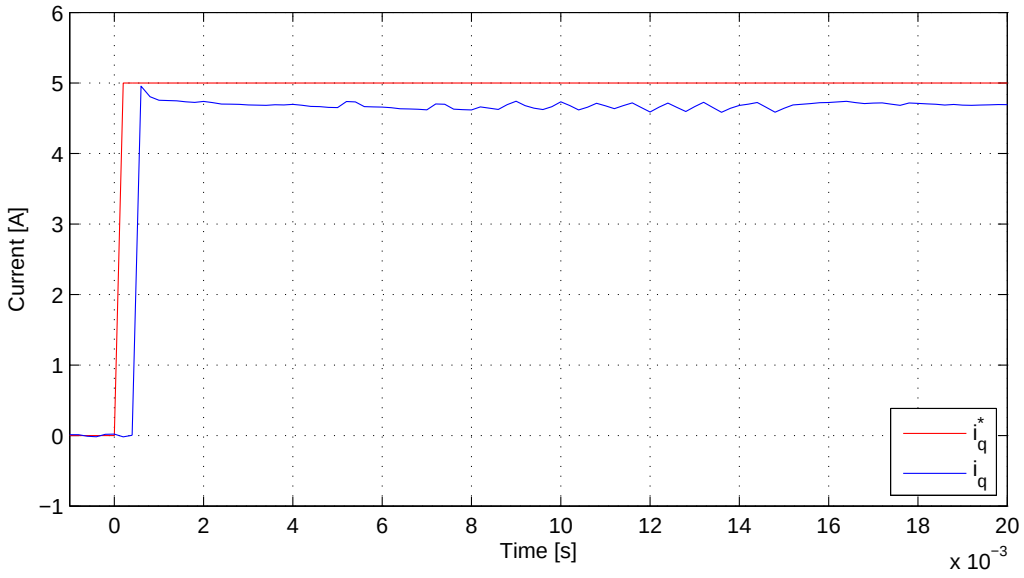


Figure 4.12. Measured current step response from 0 A to 5 A at 0 rpm

In Figure 4.13 it is commaded an step 5-10 A. The ripple measured with the oscilloscope is 2 A. The steady state error is 1.87% and 1.91% before and after the step, which is big compared with FOC in Figure 3.17

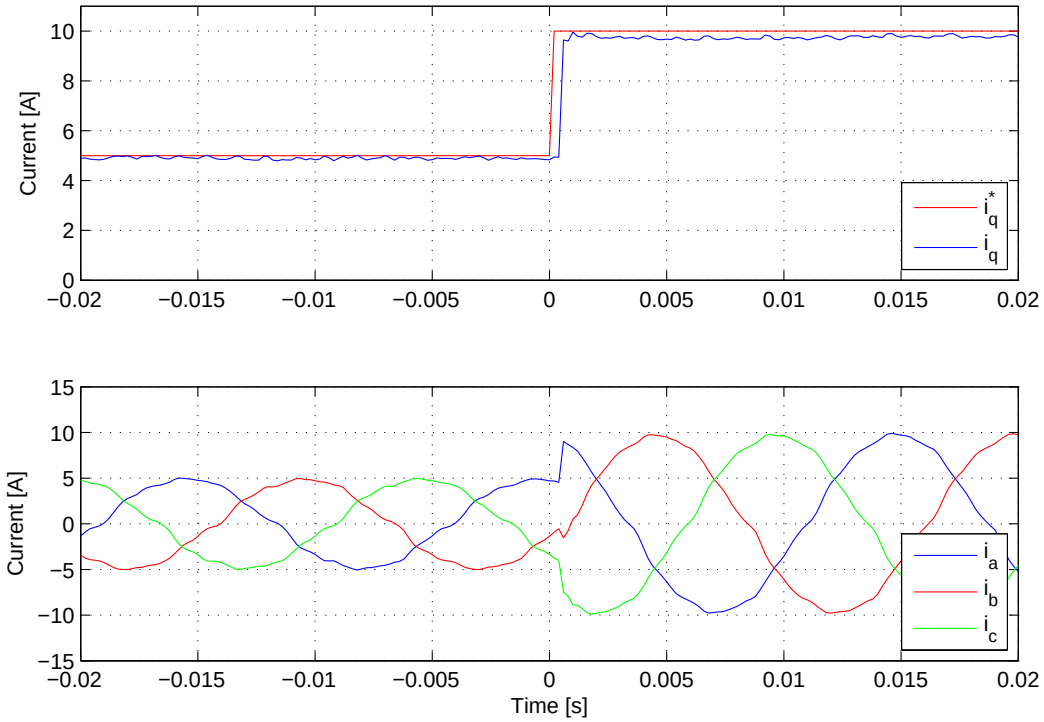


Figure 4.13. Measured current step response from 5 A to 10 A at 1000 rpm

Finally, in Figure 4.14 is presented the results to a 5 to -5 A step. The steady state error

is 1.90% and 4.93% before and after the step again bigger than in Figure 3.18 for FOC.

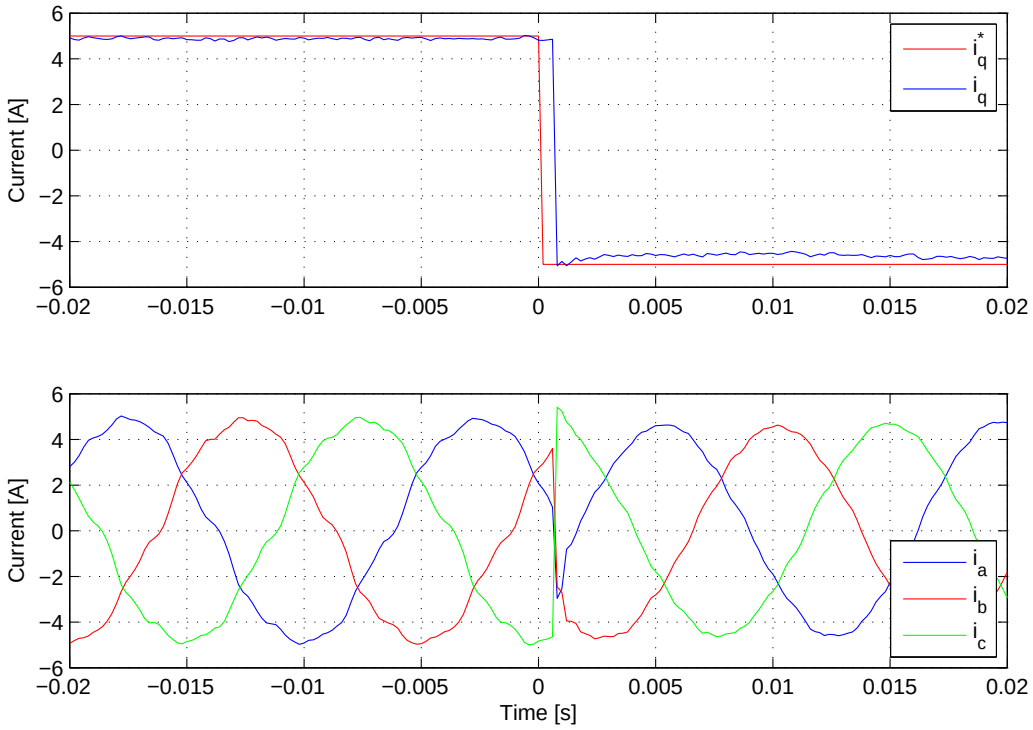


Figure 4.14. Measured current step response from 5 A to -5 A at 1000 rpm

Finite Set Model Predictive Control 5

Direct predictive control or Finite set model predictive control is one of the most researched and used predictive methods [Rodriguez et al., 2013]. This chapter will present a model predictive controller where different cost functions are compared in simulations to find the best solution for the current case. Also delay compensation will be considered to minimise the current ripple.

Model Predictive Control uses the system model in order to predict the future behaviour of the controlled variables. In order to obtain the desired actuation the controller uses the model information in a cost function. The choosing of this cost function is the main effort once an accurate model is obtained.

The control structure for a finite set model predictive control is shown in Figure 5.1.

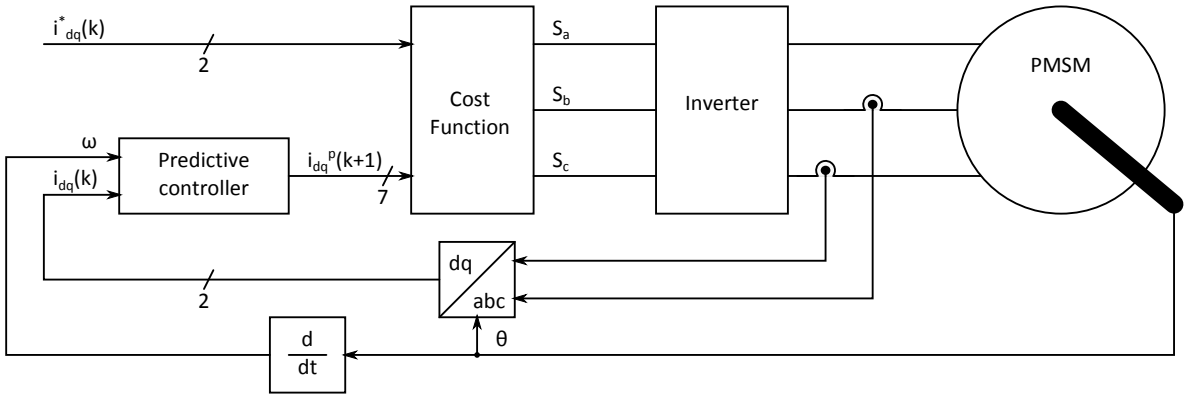


Figure 5.1. Principle of the direct model predictive controller

As shown for the SVM, the three phase inverter has 8 possible switching configuration, six of them define the six active voltage vectors and the other two corresponds with the zero voltage vectors, where either all the upper or lower switches are all on at the same time. For each of these switching configuration the inverter voltage output can be calculated applying Equation 2.24 from the inverter model. The current is measured at sampling time k and the predictive current controller calculates the seven possible current vectors

at sampling instant k+1, $i^p_{dq}(k+1)$, corresponding to the six active and the zero vector voltage using Equation 2.21 from the motor discrete model. These current vectors are evaluated by a cost function, that will find the optimum voltage vector that minimises the current error. The simplest cost function for this purpose could be the one presented in Equation 5.1, however, the cost function design will me more extensively studied in Section 5.1.

$$g = |i^*_d - i^p_d(k+1)| + \left|i^*_q - i^p_q(k+1)\right| \tag{5.1}$$

After the optimum switching state is found it will be applied directly to the inverter, without the use of any modulator like SVM.

A flowchart of the model predictive controller is shown in Figure 5.2. Also the Matlab code for the controller used in the simulations can be found in AppendixD.

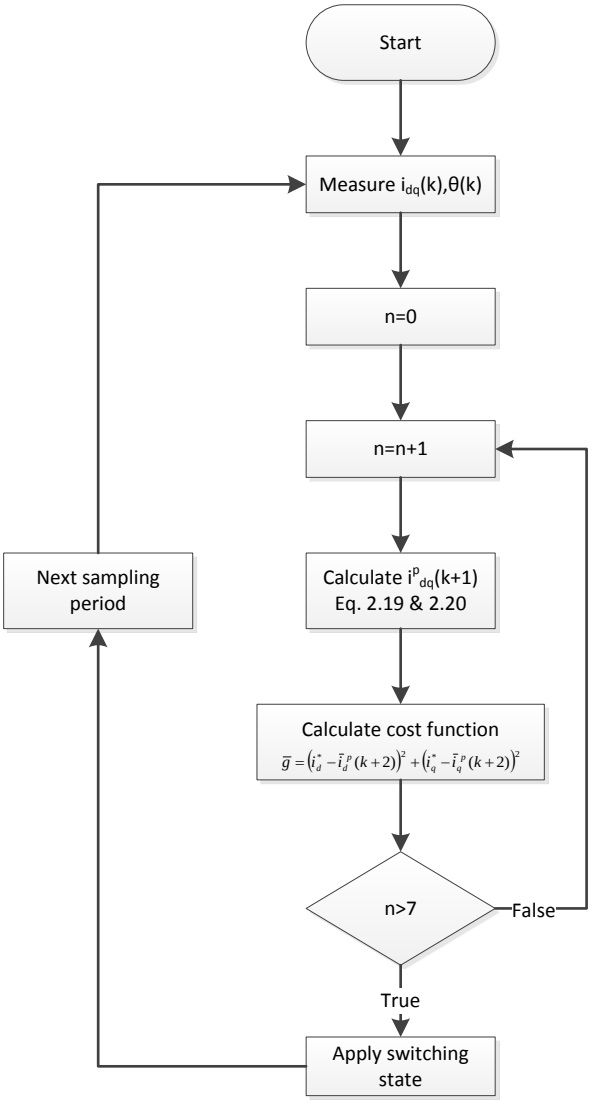


Figure 5.2. Flowchart of model predictive current controller

5.1 Cost function and weighting factor selection

This section will present some basic cost functions that provide current reference tracking. Also extra terms to reduce switching commutations will be investigated.

The cost functions given in 5.2 and 5.3 can be used in order to evaluate the current error as it is presented in Reference [Rodriguez and Cortes, 2012]. In this case, the reference is i^*_{dq} and the predicted values are the i_{dq} currents. Equation 5.2 uses the absolute error between the reference and the predicted current and Equation 5.3 uses the error squared.

$$g = |i^*_d - i^p_d(k+1)| + \left|i^*_q - i^p_q(k+1)\right| \tag{5.2}$$

$$g = \left(i^*_d - i^p_d(k+1)\right)^2 + \left(i^*_q - i^p_q(k+1)\right)^2 \tag{5.3}$$

Both absolute and squared method were simulated at a sampling frequency of 20 kHz. They seem to follow the reference as shown in Figure 5.3. The average values of the i_{dq} currents when the cost function from Equation 5.2 is used, are i_d = -0.25 A and i_q = 19.51 A. If Equation 5.3 is applied instead the mean values are i_d = -0.27 A and i_q = 19.88 A. This means that the squared cost function method gives better i_q reference tracking and a slightly better overall performance than the absolute method. The squared method is therefore chosen for this project.

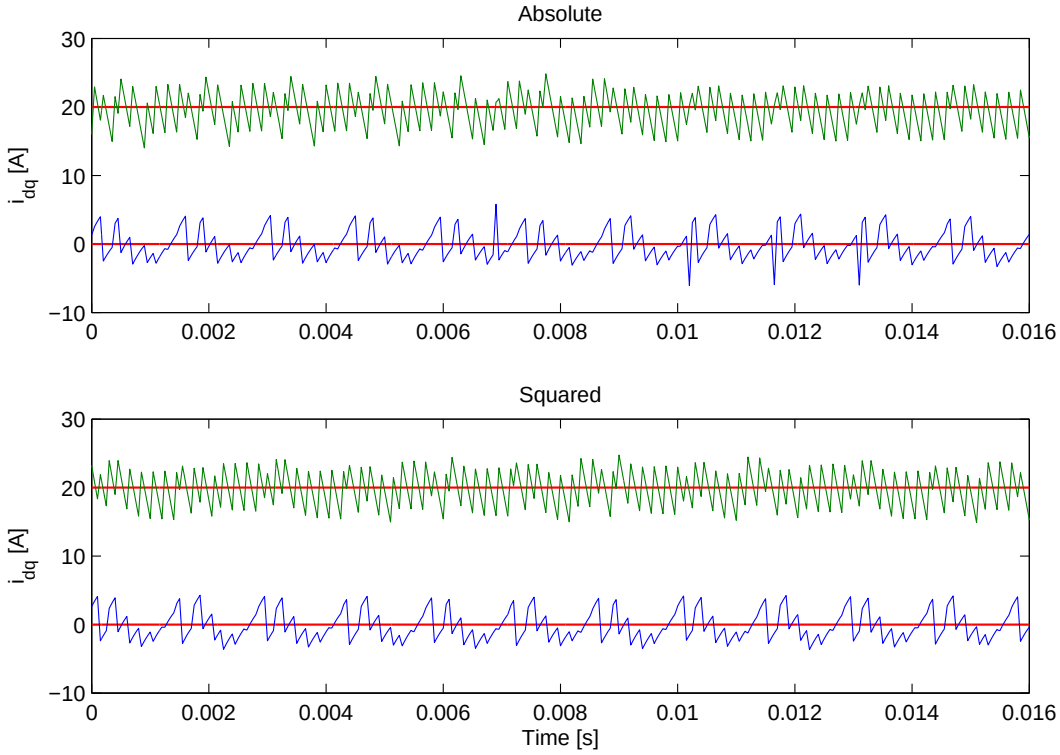


Figure 5.3. Predictive current control where the absolute error and the squared error method for the cost function is compared. The red lines are the i_{dq} references where i_d = 0 A and i_q = 20 A

In Equation 5.4 an additional term has been added which is called the weighting factor. It is one of the most important and difficult parts of the MPC design. It allows to give priority to one or another variable. It can be more complex than tuning a PI.

By adjusting the weighting factor λ_{SW} the importance of the switching frequency can be set. In situations where the switching loses are important λ_{SW} can be increased to fulfil these requirements. The n term from Equation 5.5 sums up the absolute error between the applied switching function and the one from the previous sampling. For example if all the switches at sampling instant k should change, the total error value for n will be equal to 3. The term $\lambda_{SW} \cdot n$ is added to the cost function which will make this specific switching pattern less favourable for choosing. If there is no change in switching state from last sampling instant, n will be equal to zero, hence no additional term is added to the cost function and is therefore more favourable for choosing.

$$g = (i_d^* - i_d^p(k+1))^2 + \left(i_q^* - i_q^p(k+1)\right)^2 + \lambda_{SW} \cdot n \tag{5.4}$$

Where

$$n = |S_a(k) - S_a(k-1)| + |S_b(k) - S_b(k-1)| + |S_c(k) - S_c(k-1)| \tag{5.5}$$

In Figure 5.4, 5.6 and 5.8 the effects of weighting factor are shown. Figure 5.4 and 5.5 shows the MPC controller with no weighting factor and those figures serves as a reference for comparison. Figure 5.6 and 5.7 have a weighting factor $\lambda_{SW} = 35$. In those two figures there can be seen a small reduction in the switching frequency but this reduction comes with a price of higher current ripple and distortion. In Figure 5.8 and 5.9 the weighting is set to 70. The switching frequency have clearly been reduced according to Figure 5.9 but the current ripple is increased to almost 20 A from peak-to-peak and current distortion is increased as well. This is seen in Figure 5.8.

Unfortunately the value of the weighting factor has to be found empirically by trial and error [Cortes et al., 2009]. With a weighting above 70 it has be noticed that with low current references the controller only produces zero vectors as an output. A weighting factor from 20-40 is found to be recommendable for this specific motor case. At too low values there are no big change in switching frequency. At too high values the system gets unstable. There is noticed a linear relationship between the weighting factor and the reduction of the average frequency from Figure 5.5, 5.7 and 5.9.

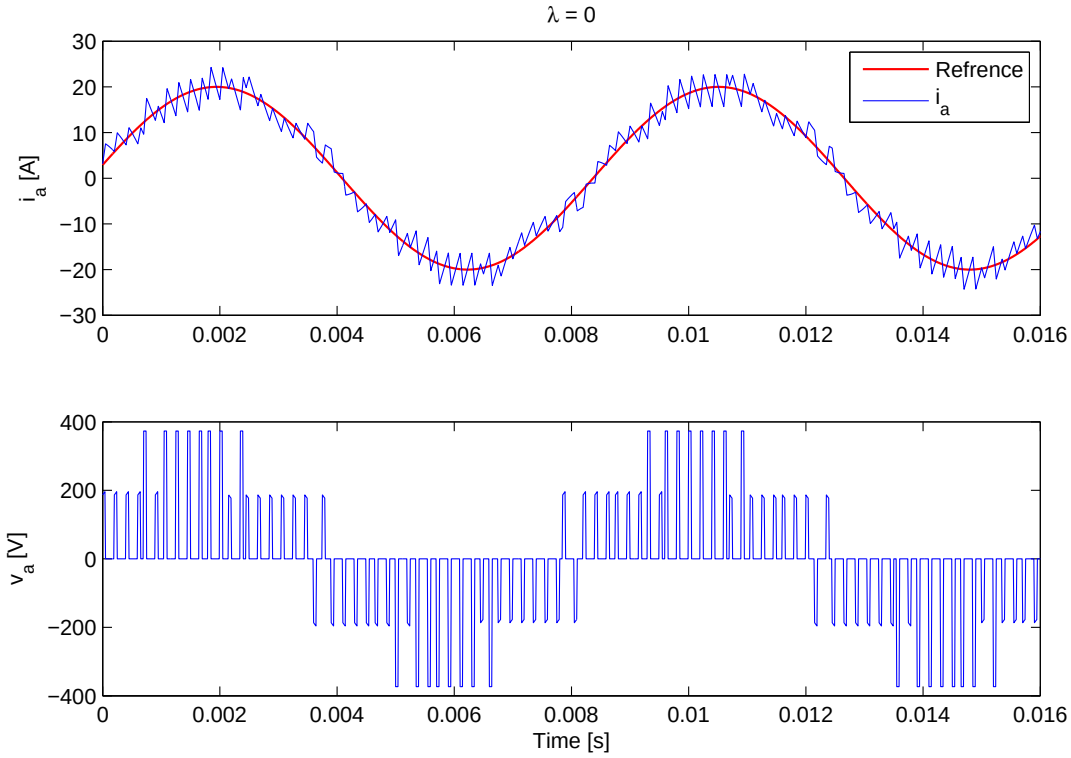


Figure 5.4. Predictive current control with a switching weighting factor $\lambda_{SW} = 0$

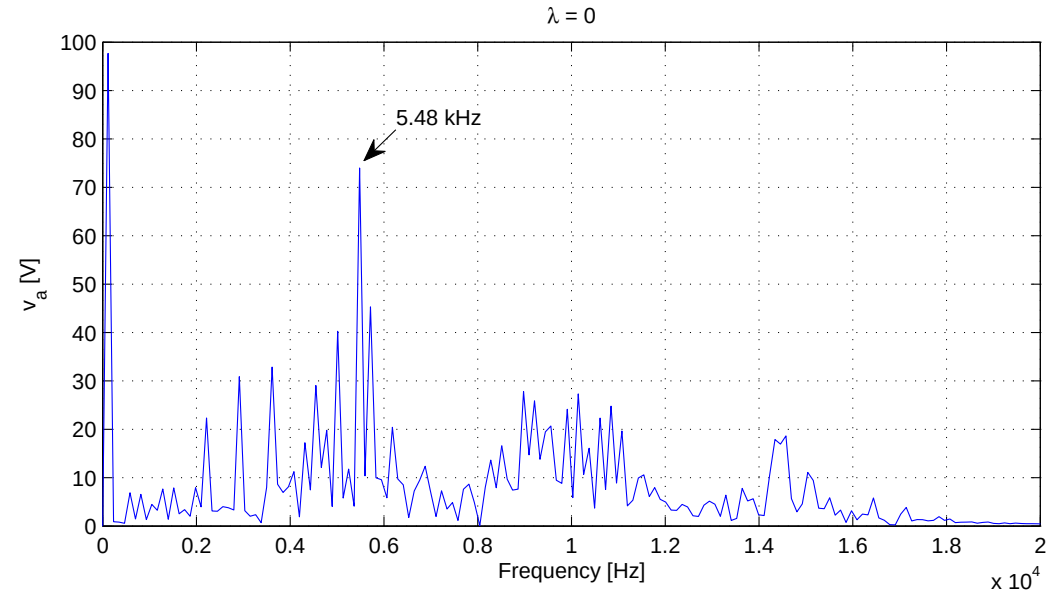


Figure 5.5. Spectrum analysis of the phase voltage for the predictive current control with a switching weighting factor $\lambda_{SW} = 0$

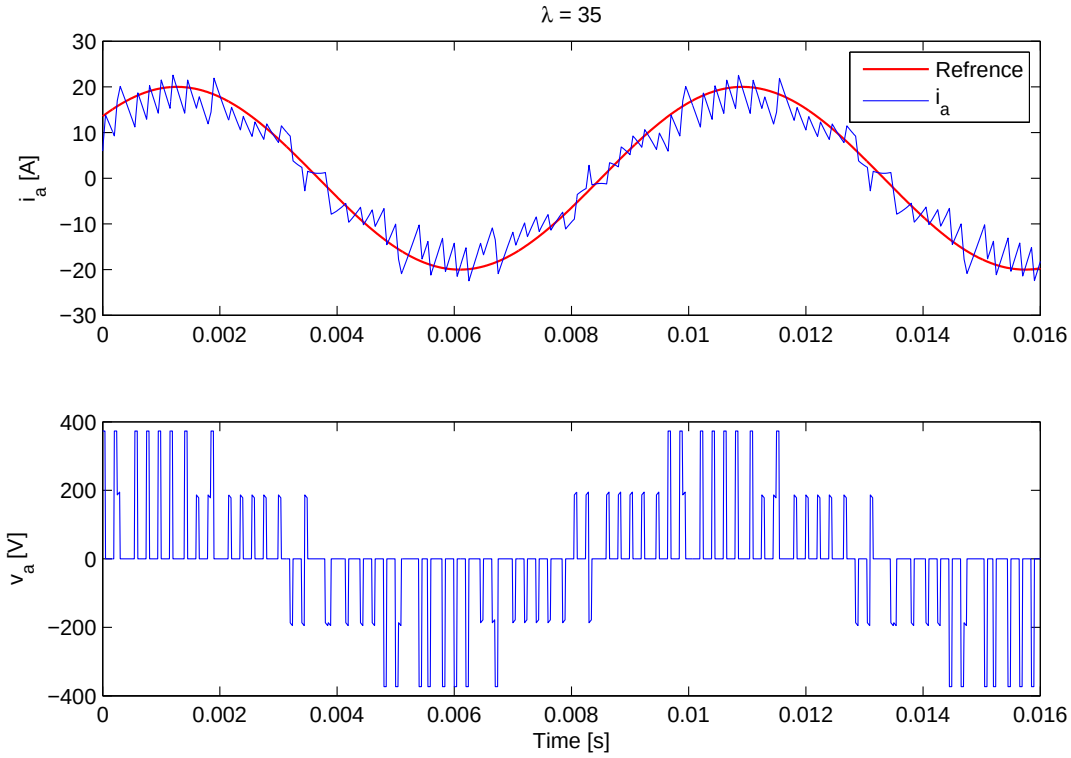


Figure 5.6. Predictive current control with a switching weighting factor $\lambda_{SW} = 35$

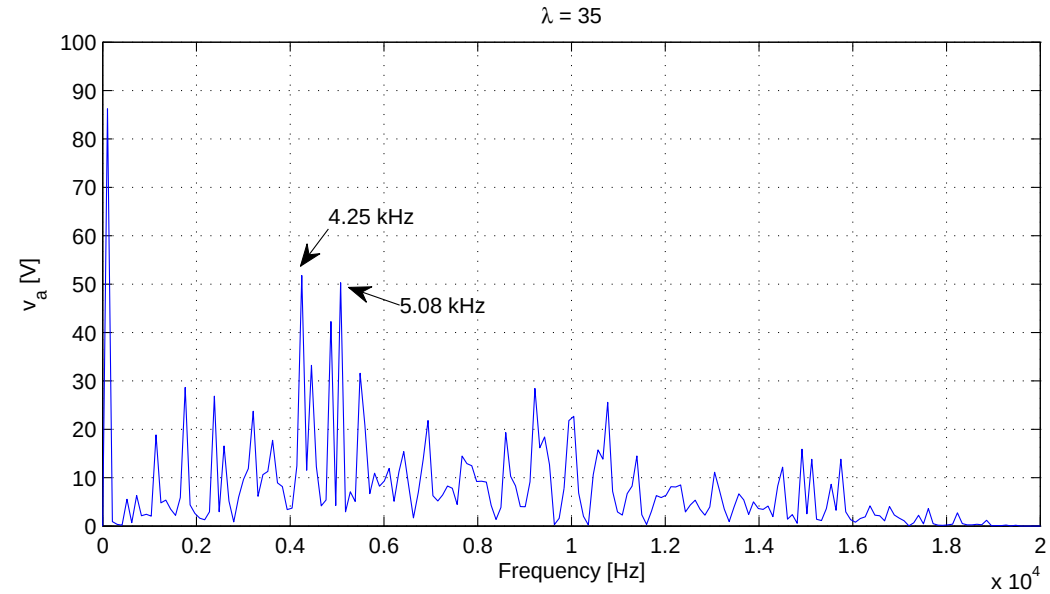


Figure 5.7. Spectrum analysis of the phase voltage for the predictive current control with a switching weighting factor $\lambda_{SW} = 35$

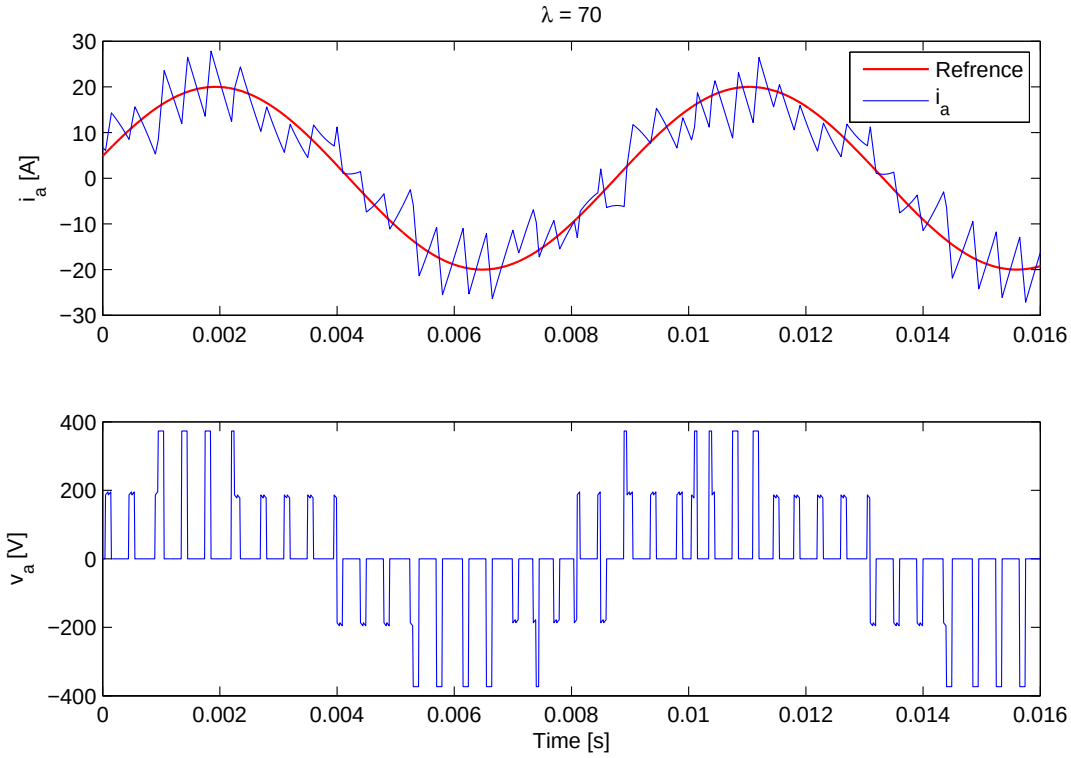


Figure 5.8. Predictive current control with a switching weighting factor $\lambda_{SW} = 70$

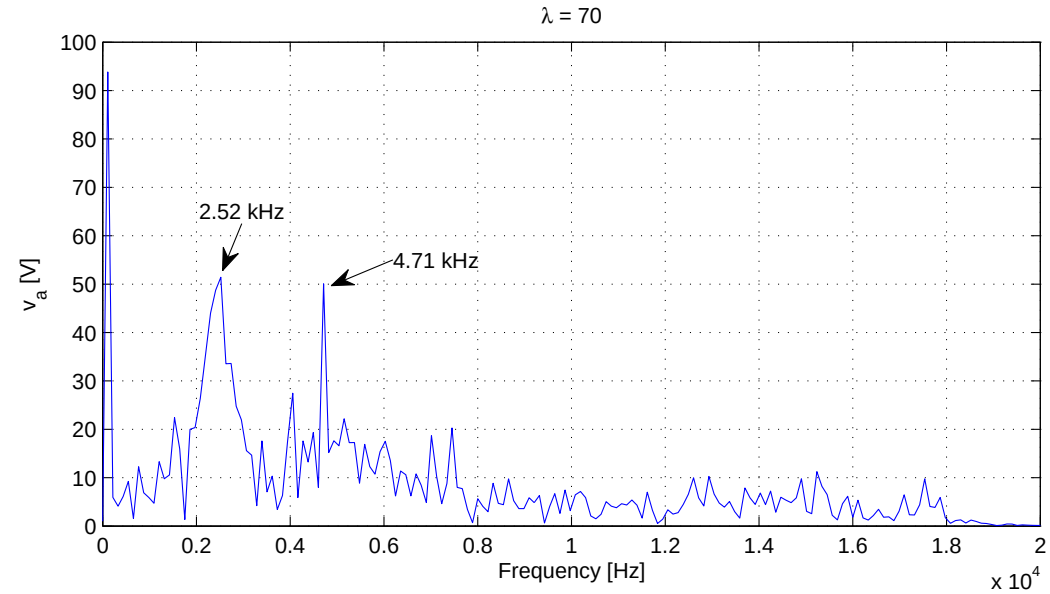


Figure 5.9. Spectrum analysis of the phase voltage for the predictive current control with a switching weighting factor $\lambda_{SW} = 70$

Hard constraints can also be added to the cost functions like in Equation 5.6. The hard constraints are a safety feature, which limits the current output magnitude. If for example the predicted current exceeds the maximum limit for the motor a penalty is added to the cost function. In Equation 5.7 infinity is added to the cost function if i_{max} is exceeded the

cost function takes the value $g = \infty$ which means that this voltage vector is not chosen. If the predicted current is less than i_{max}, zero is added to the cost function and only the voltage vector that minimize the current error is chosen.

$$g = (i_d^* - i_d^p(k+1))^2 + \left(i_q^* - i_q^p(k+1)\right)^2 + f_{lim} \tag{5.6}$$

where

$$f_{lim} = \begin{cases} \infty & \text{if } |i_d^p(k+1)| \text{ or } |i_q^p(k+1)| > i_{max} \\ 0 & \text{if } |i_d^p(k+1)| \text{ and } |i_q^p(k+1)| \leq i_{max} \end{cases} \tag{5.7}$$

Reduction of the inverter commutations when choosing a zero vector can help to reduce and distribute the switching losses in the inverter. This is achieved by either choosing the 0 (000) or 7 (111) vector based on previous state. For example if the previous state is 4 (011) and the next state have to be a zero vector, then 7 (111) would be the best choice, because only one inverter leg have to change. If 0 (000) was chosen, two inverter legs would have to change state. An example is shown in Figure 5.10. The top figure shows a case where only the zero vector 0 is chosen. The second figure shows the choosing of the zero vectors based on the previous state. Between the two dashed lines switching loss reduction is achieved.

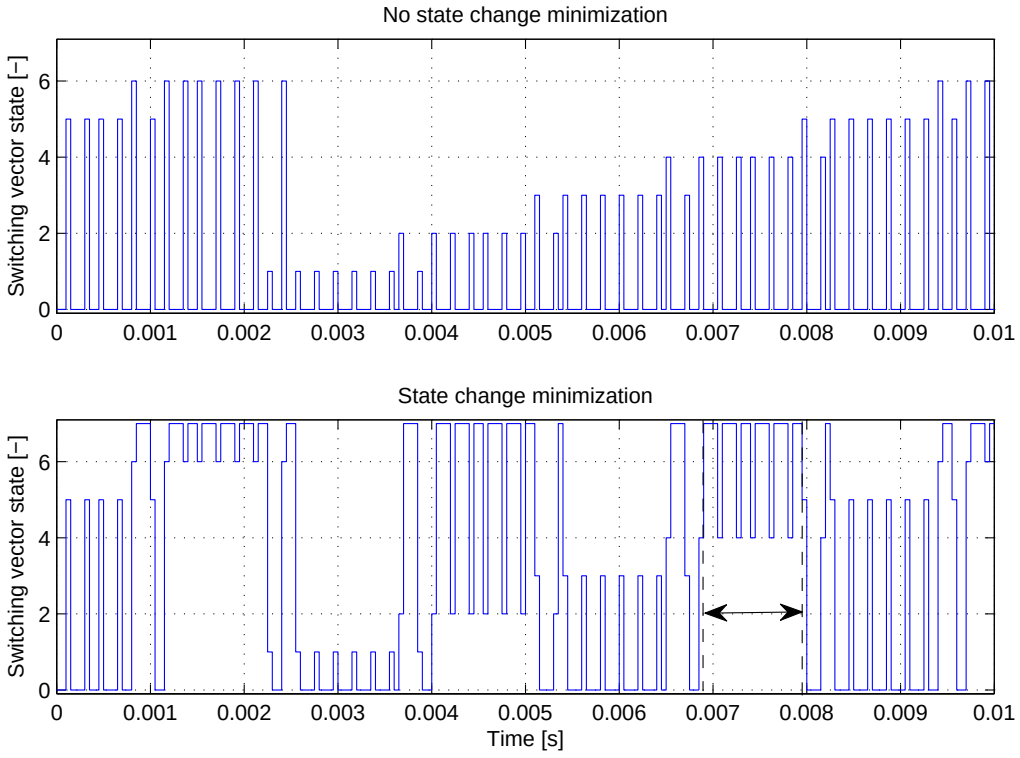


Figure 5.10. First plot show if only one zero vector is selected, for this case 0 is chosen. The second plot shows a more effective choosing of the two zero vectors minimizing inverter commutations

For this project a cost function with squared current errors is chosen because it provides a better i_q reference tracking. When it comes to switching frequency reduction the weighting factor λ_{SW} is set to zero because a small current ripple is more important for the project. The hard constraints is implemented in the used cost function to protect the PMSM motor which have a current limit $i_{max} = 86$ A.

5.2 Calculation Delay compensation

In this section, the delay compensation which was suggested for deatbeat in Chapter 4 is further developed for the particular case of MPC. In this case, the interest on implementing delay compensation is more complex and of bigger relevance. This is because the delay increases the current ripple, as it will be shown in the following.

Ideally, the optimum voltage calculated is applied at the beginning of the sampling time, so that the computation time is considered zero and the system operation looks like Figure 5.11. In real implementation, a delay is introduced in the actuation due to the large amount of calculations in between the current acquisition and the application of the new calculated switching states and inverter voltage as shown in Figure 5.12. Note that, in benefit of the understanding, the figure represents a general case, where the voltage output update happens at the end of the calculation time. However, In this particular case the actuation loading from the controller only takes place at the end of the sampling time as shown in Figure 5.13

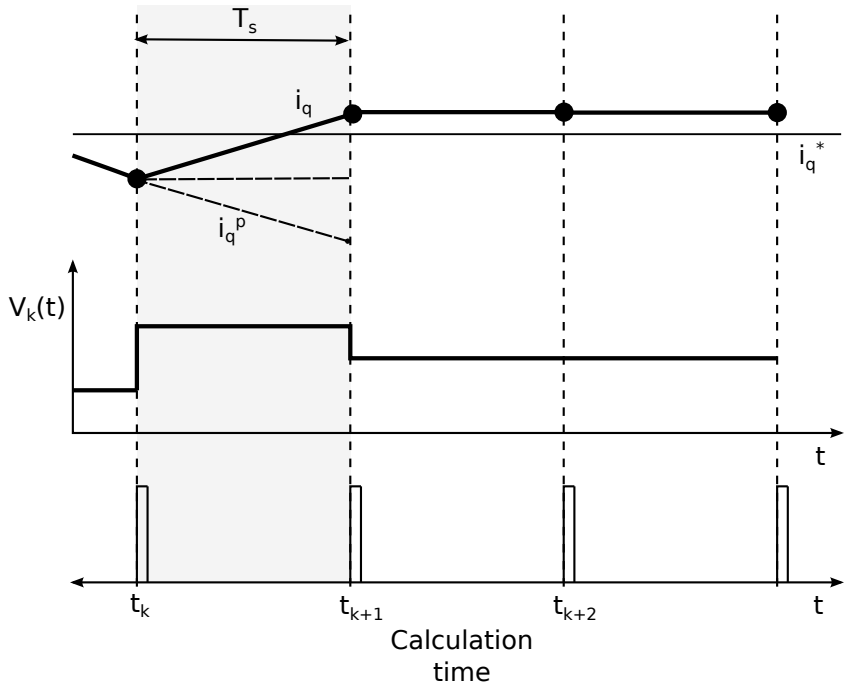


Figure 5.11. Ideal predictive control operation

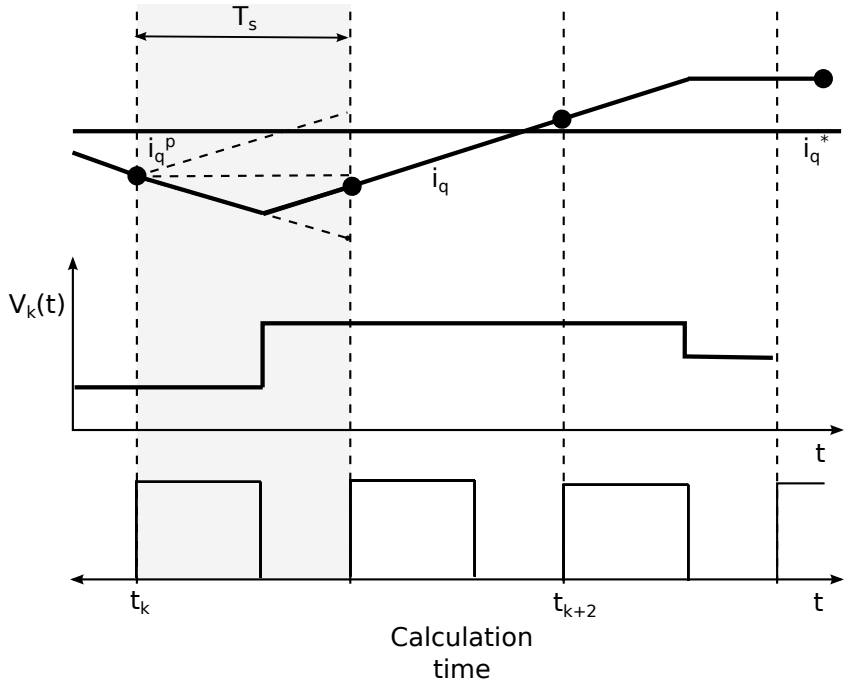


Figure 5.12. Real predictive control operation

According to [Cortes et al., 2012], a way to avoid this delay is to evaluate the current at instant *k+2*. Using again Equations 2.19, 2.20 from the discrete motor model, $i_{dq}^{p}(k+2)$ is now calculated as shown in Equation 5.8 and 5.9.

$$i_d^p(k+2) = \frac{Ts}{L_d}u_d(k+1) + \left(1 - \frac{RTs}{L_d}\right)\hat{i}_d(k+1) + \frac{\omega(k)L_qTs}{L_d}\hat{i}_q(k+1) \tag{5.8}$$

$$i_q^p(k+2) = \frac{Ts}{L_q}u_q(k+1) + \left(1 - \frac{RTs}{L_q}\right)\hat{i}_q(k+1) - \frac{\omega(k)L_dTs}{L_q}\hat{i}_d(k+1) - \frac{\omega(k)\lambda_{PM}Ts}{L_q} \tag{5.9}$$

In this way, the voltage to apply in instant *k+1* is calculated in advance and applied at the beginning of the switching period t_{k+1}. The future current value *i(k+1)* is not known so it should be estimated as $\hat{i}(k+1)$. That is again calculated from the discrete model Equations 2.19 and 2.20 using the measured *i(k)* and the just applied voltage *u(k)*. This is shown in 5.10 and 5.11.

$$\hat{i}_d(k+1) = \frac{Ts}{L_d}u_d(k+1) + \left(1 - \frac{RTs}{L_d}\right)\hat{i}_d(k+1) + \frac{\omega(k)L_qTs}{L_d}\hat{i}_q(k+1) \tag{5.10}$$

$$\hat{i}_q(k+1) = \frac{Ts}{L_q}u_q(k+1) + \left(1 - \frac{RTs}{L_q}\right)\hat{i}_q(k+1) - \frac{\omega(k)L_dTs}{L_q}\hat{i}_d(k+1) - \frac{\omega(k)\lambda_{PM}Ts}{L_q} \tag{5.11}$$

The compensated controller would operate as shown in Figure 5.13.

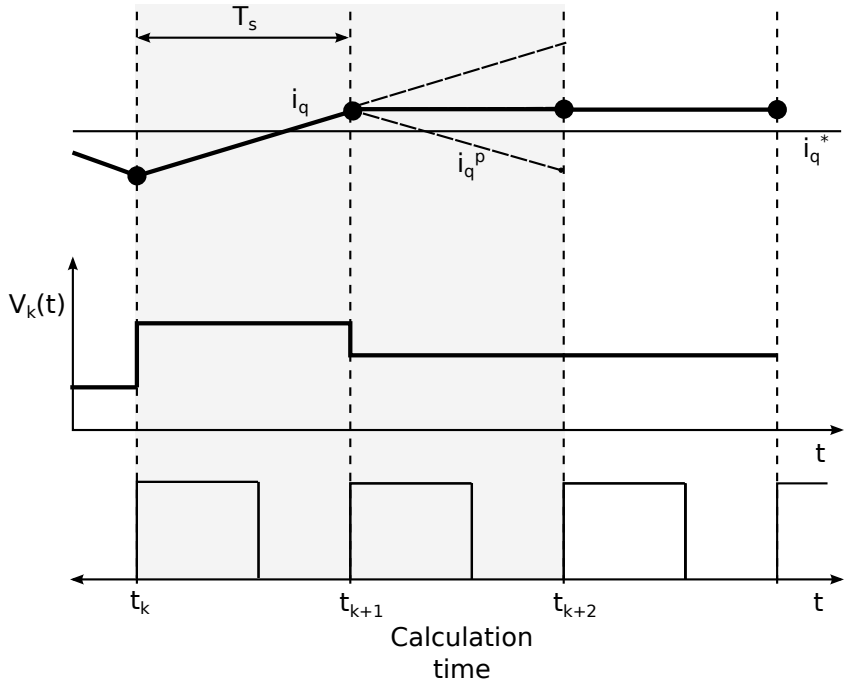


Figure 5.13. Predictive control operation with delay compensation

The cost function 5.12 would evaluate the error two sampling time ahead.

$$g = (i_d^* - i_d^p(k+2))^2 + \left(i_q^* - i_q^p(k+2)\right)^2 \tag{5.12}$$

When the future reference values are used in the cost function 5.12, this is considered to be the same as the actual reference. This is valid when the reference is a constant value or the switching frequency is much higher than the frequency of the reference variable. In such cases, a method for calculating future reference should be introduced [Rodriguez and Cortes, 2012].

In Figure 5.14 a new flowchart including the delay compensation, described in this section, is shown.

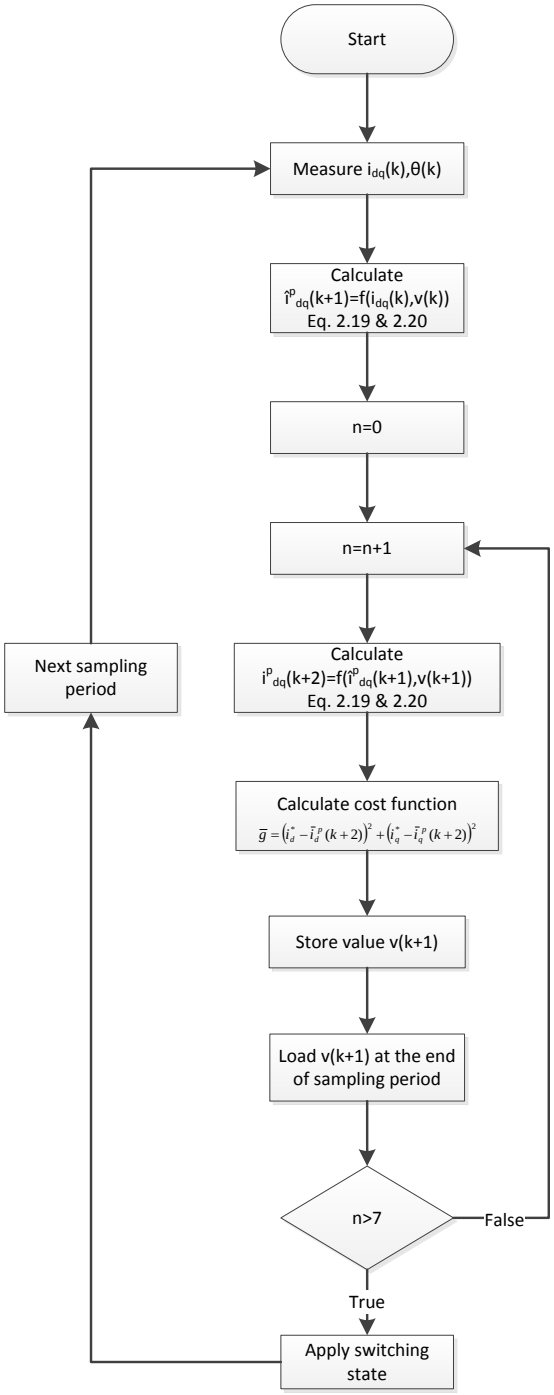


Figure 5.14. Flowchart of model predictive current controller with delay compensation

With this compensation, it is intended to obtain a reduction on the current error in the real implementation and therefore avoiding an increase in the current ripple. The results are shown in simulations.

5.3 Inverter dead-time compensation

In Section 4.1, the influence of the inverter dead-time and its effects on the steady state error was pointed out. The principles explained for deadbeat are valid now, however, the compensation approach is particularise for MPC. This has be treated in [Imura et al., 2012]. According the simulations, it will be shown in following section that MPC scheme is not as affected by this voltage error as deadbeat. However, due to the easy implementation and according to the main goal of obtaining the best performance possible, dead-time implementation is explained in this chapter. The results obtained in simulations will be presented in Section 5.4.

In MPC the most appropriate inverter switching state is chosen for each switching period and give out directly without the need of a modulator. Due to the dead-time within the switching periods, the calculated $u_{dq}(k)$ is not applied along the full period as ideally, instead, a voltage vector u_{dq}^{DT} is applied during the dead-time. Therefore, the real voltage applied during a switching period would be as in Equation 5.13.

$$\bar{u}_{dq}(k) = \frac{t_{DT}}{T_s}\bar{u}_{dq}^{DT} + \frac{T_s - t_{DT}}{T_s}\bar{u}_{dq}(k) \tag{5.13}$$

The voltage vector $\bar{u}_{dq}^{DT}$ can be calculated using Equation 2.24, it is only needed to obtain what is the value of S_a, S_b, S_c during dead-time. The inverter performance in this time gap is dependent on the initial and final switching state and the phase current sign. The influence on the sign in the dead-time effect was already shown in Figure 4.5. A closer look is given in Figure 5.15 where the inverter state before with voltage U_1, during, with voltage U^{DT}, and after, with voltage U_2, the dead-time is detailed and the currents indicated. In this case, only leg b is switched.

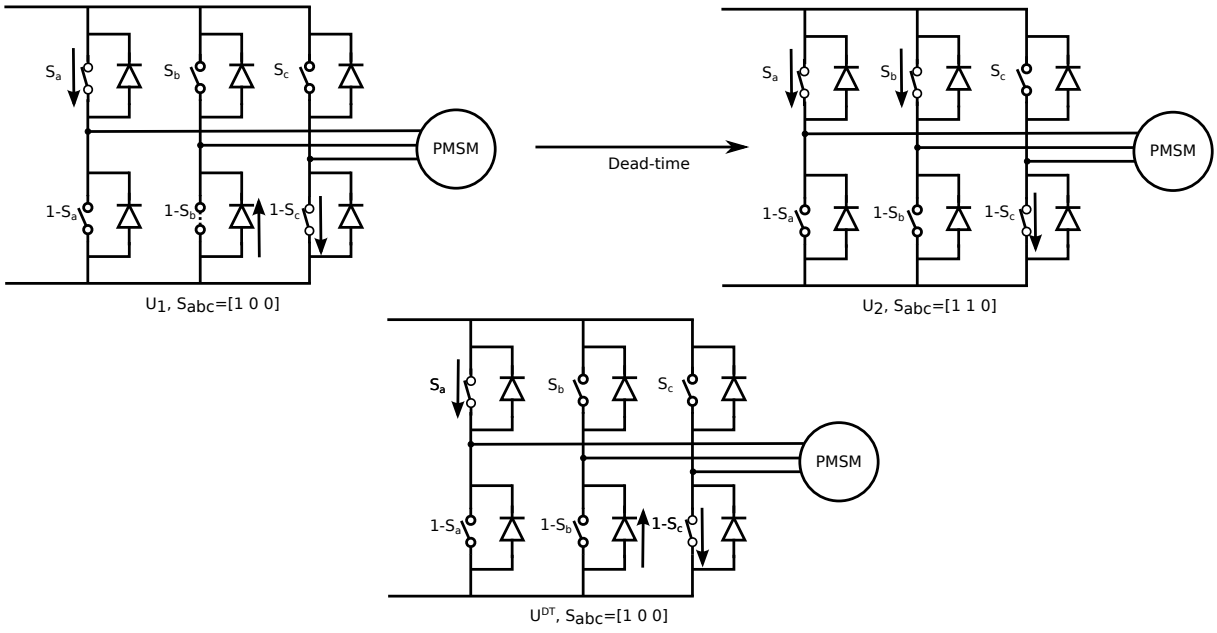


Figure 5.15. Inverter behaviour during dead-time

After the analysis of that phenomenon, it can be stated that, if the current phase is positive the current flow will be maintain along the dead-time. With a negative current during dead time it will be deviated to the current direction for next sampling time. In order to know, each moment, the sign of each of the currents it is enough to know the rotor angle according to Figure 5.16 where the phase sign of a, b and c respectively are indicated for each of the sectors. An encoder attached to the motor shaft provides the position information.

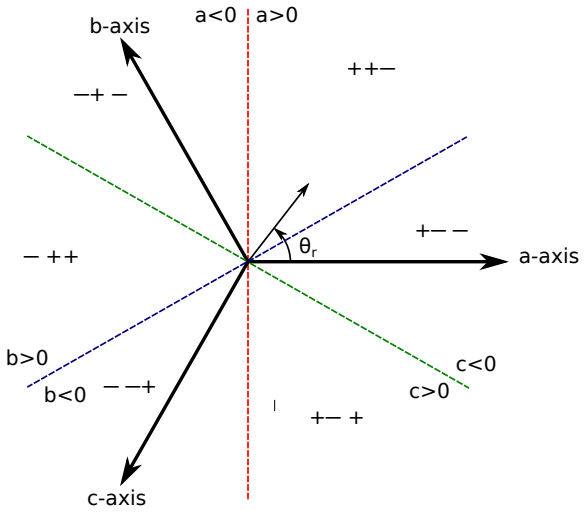


Figure 5.16. Relationship phase current sign vs. rotor angle in the static reference frame

Once the vector $u_{dq}^{\bar{D}T}$ is determined, it can be added to $\bar{u}_{dq}(k)$ according Equation 5.13. The new $\bar{u}_{dq}(k)$ is used in order to predict real current that will be obtained after one sampling time for each of the inverter configurations. The cost function will chose the corrected voltage vector which minimises the current error as evaluated in the cost function.

5.4 Simulation Results

The Simulink diagram for the MPC is illustrated in Figure 5.17. The PI and the SVM block from the FOC simulation in Section 3.3 have been replaced with a Matlab function block. The output from the MPC block are the switching states, which are applied directly to the inverter. A more detailed description of the Matlab function code is found in Appendix D.

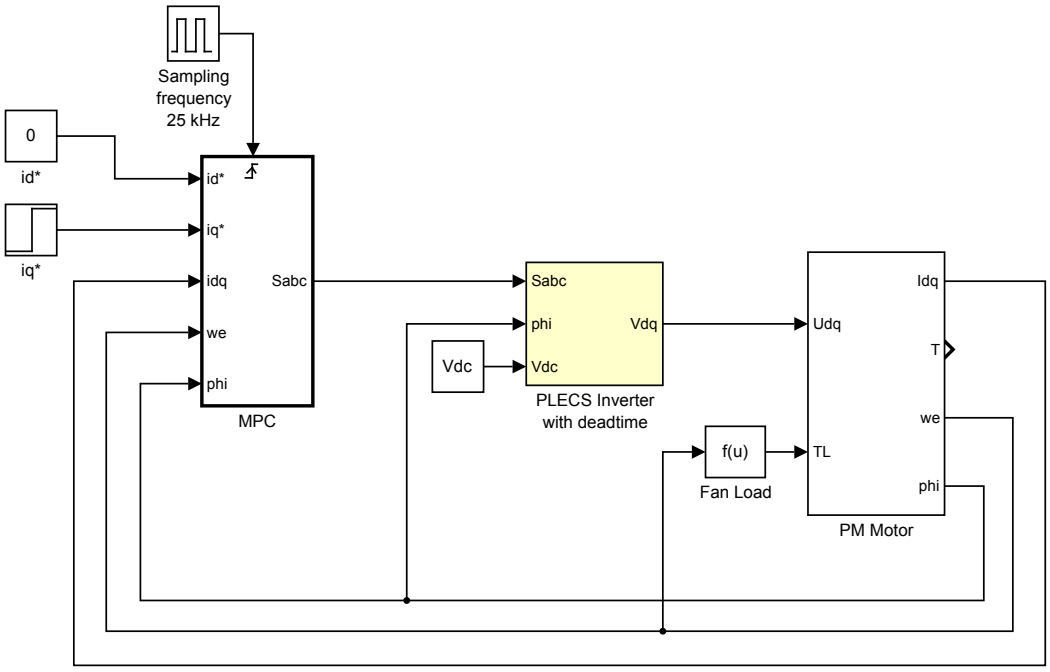


Figure 5.17. Simulink model for the MPC

According to the working principle of MPC, as it has been seen Section 5.1, the switching frequency is variable and unpredictable. It has been noticed that the mean switching frequency depends on the operating point, thus depending on the speed.

The MPC algorithm computation time was measured as 36 μs for the dSpace DS1103 with a microprocessor running at 1 GHz. That means that the maximum sampling frequency that can be chosen is 27.77 kHz.

In order to make a fair decision based on the sampling frequency, the switching spectrum for a 25 kHz sampling frequency and a testing speed of 1000 rpm, was analysed. In Figure 5.18 it can be seen that, as expected, the switching frequency is not concentrated to a fixed number but spread out along the spectrum. The main switching frequency content is between 0 and 2.5 kHz. It could be estimated that the mean switching frequency is located in that gap.

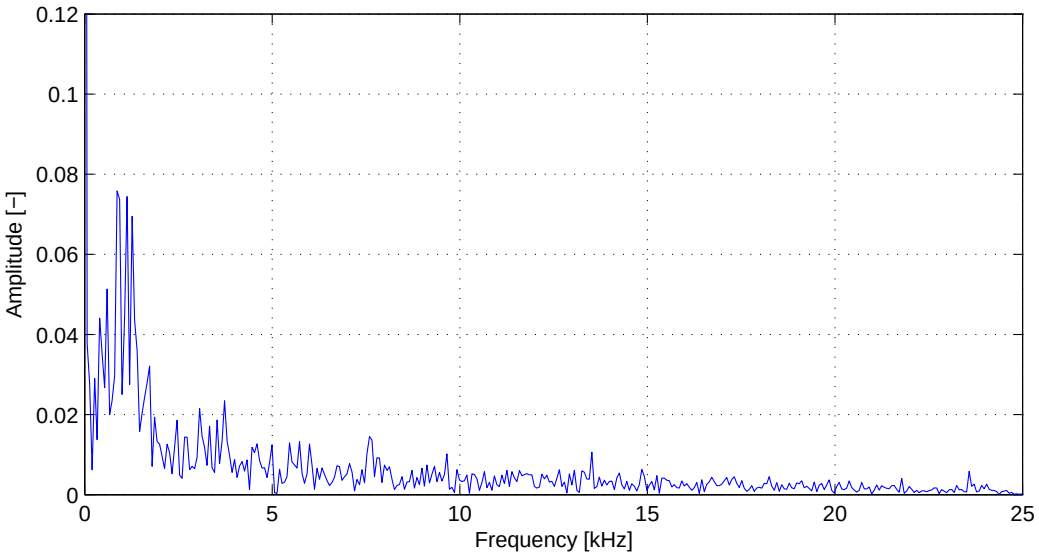


Figure 5.18. Switching frequency spectrum with 25 kHz sampling frequency for 1000 rpm

It would be interesting for the comparison that the mean switching frequency would reach higher values to be nearer the 5 kHz used for both deadbeat and FOC. However, the sampling frequency should be greatly increased to 80 kHz. In Figure 5.19 the spectrum for 80 kHz is shown, which may be considered to have 5 kHz mean switching frequency. That high sampling frequency is not easy to implement in practice due to the very limited time left for running the controller algorithm in DSP. 25 kHz is chosen for being the maximum possible sampling frequency with the available hardware.

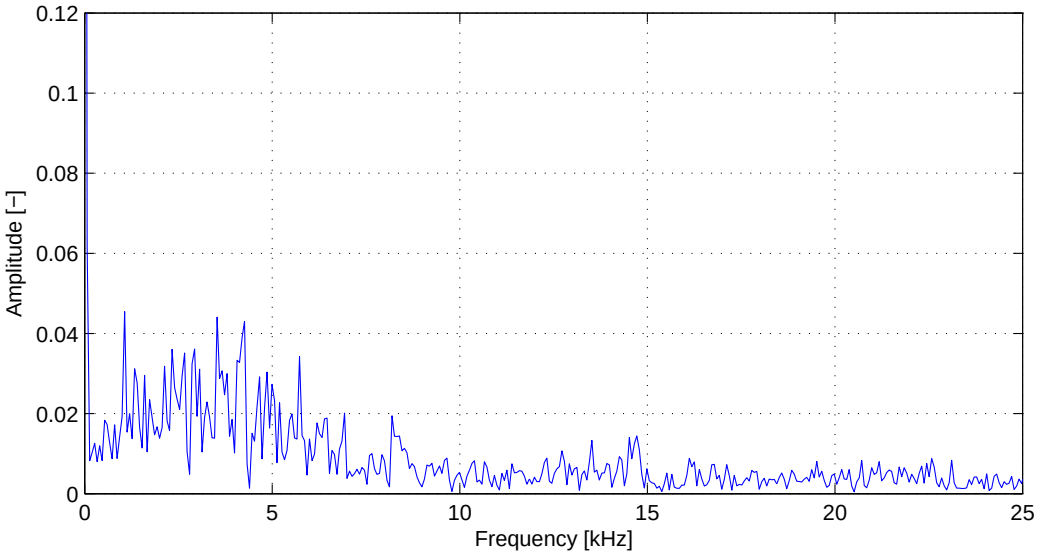


Figure 5.19. Switching frequency spectrum with 80 kHz sampling frequency for 1000 rpm

The characteristics of MPC and the design achievements are supported with the simulations which are presented in the following. Firstly, the basic implementation of

MPC is simulated at 0 rpm without taking into account the system delay. The effect of an actuation delay is shown and compensation delay is found convenient. Due to the small effect of dead-time in the steady state error the simulations for 1000 rpm, were performed only with delay compensation. Due to the big ripple occurring in the simulations at 25 kHz, the steps references are 5 A bigger than the ones shown in previous control implementation, so that the Figures appear more clear.

A step from 0 A to 10 A at zero speed is shown in Figure 5.20. It could be intuitively said, that the rising time should be a little less than a sampling period, which is 40μs with a sampling frequency of 25 kHz. However, the rising time is measured to be 56μ. In MPC for any sampling frequency over 17.8 kHz, this rising time will be invariantly 56μs at zero speed. The reason for this minimum rising time is given by the inductance di/dt. That can be seen in 5.14, where $u_q = \frac{2}{3} u_{dc} \cdot \sin(60)$=300 V, i_q=10 A, L=2.1 mH, R=0.203 Ω and ω=0 rad/s. Therefore the maximum di/dt is 142 kA/s. This corresponds with the measured 56μ.

$$u_q = R i_q + L_q \frac{di_q}{dt} + \omega \left(L_d i_d + \lambda_{PM} \right) \tag{5.14}$$

Besides this rising time the overshoot is also in any case 0%, so that the transient response is invariable for every zero speed test. In Figure5.20, the current ripple reaches a value as high as 6.36 A and the steady state error is 5.11%.

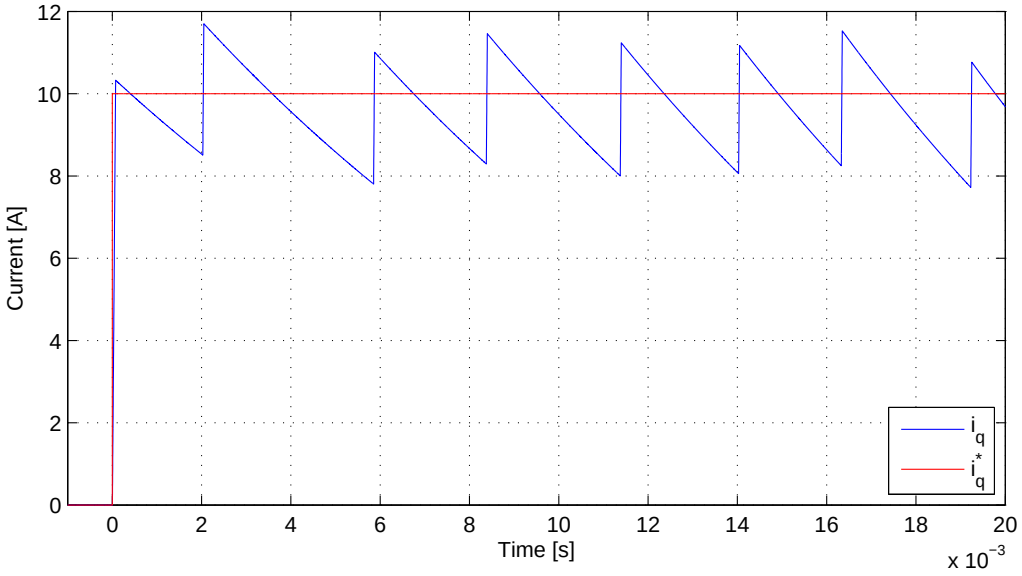


Figure 5.20. Motor start up current response

The MPC has high dynamic performance, but with high amplitude and low frequency current ripple. The ripple frequency will increase as the rotor gains speed. The ripple occurs because the MPC can only choose within the seven possible switching stages of the inverter, which will be applied for the full switching period. In contrast, a SVM,

for example, is able to reproduce the reference voltage vector by the combination of two active vector and the zero vectors every switching period, therefore the ripple is reduced. The ripple amplitude can be reduced by a decrease of the sampling time, in that way, voltage vector can change more often and track the reference more accurately.

For simulating the real system, a delay time should be implemented in the system in order to simulate one sampling time delay in the actuation of the controller, as it would appear in the real system. With the delay implemented, it is observed a great increase in the current ripple in Figure 5.21, with a value of 16.97 A. The steady state error is decreased in this case to 2.88%.

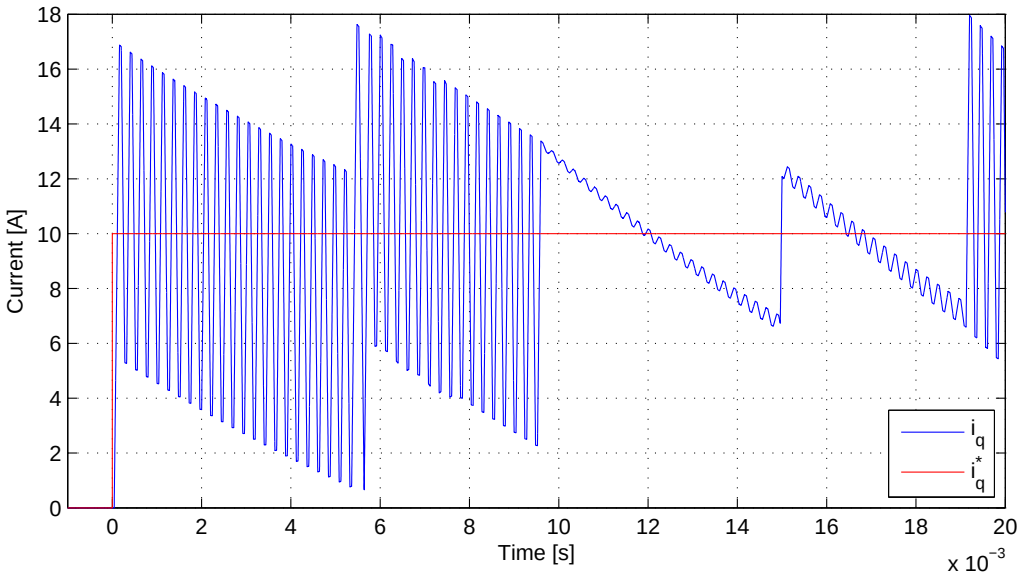


Figure 5.21. Simulated current step response with actuation delay

This increase of the current ripple was uphold in Section 5.2 where also a compensation method was proposed. The result with this delay compensation is shown in Figure 5.22. Note that the delay compensation adds one sampling delay between the reference and the current response. It can be seen a reduction on the current ripple to 6.34 A with steady state error measured as 5.11%. These values are similar to the ideal case in Figure 5.20.

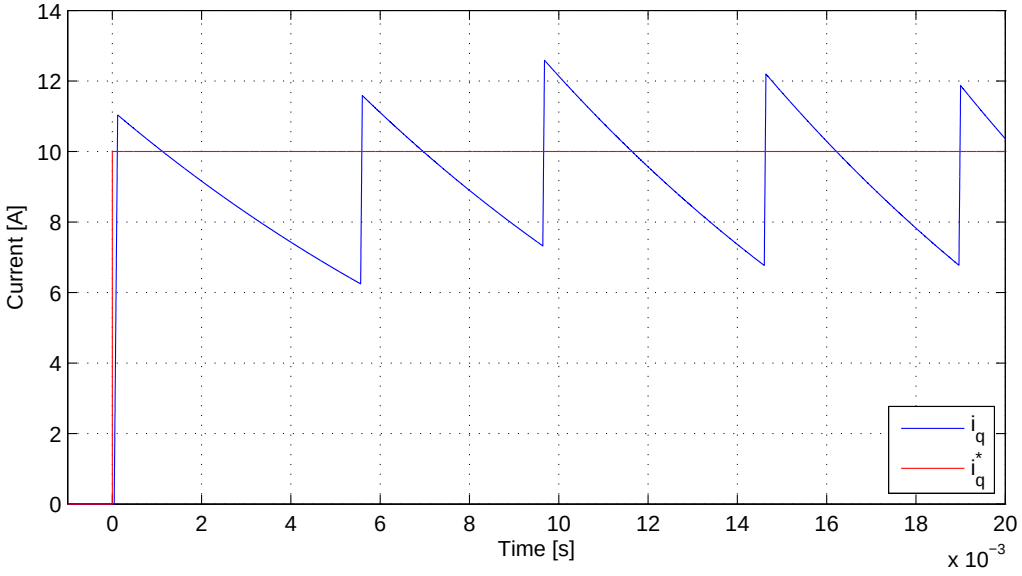


Figure 5.22. Simulated current step response with delay compensation

To make a more fair comparison of the ripple difference between the delayed uncompensated and the compensated MPC controller, both will now be tested for different steps at 1000 rpm. An step command from 10A to 20A for the uncompensated system in Figure 5.23 and the compensated systems in Figure 5.24.

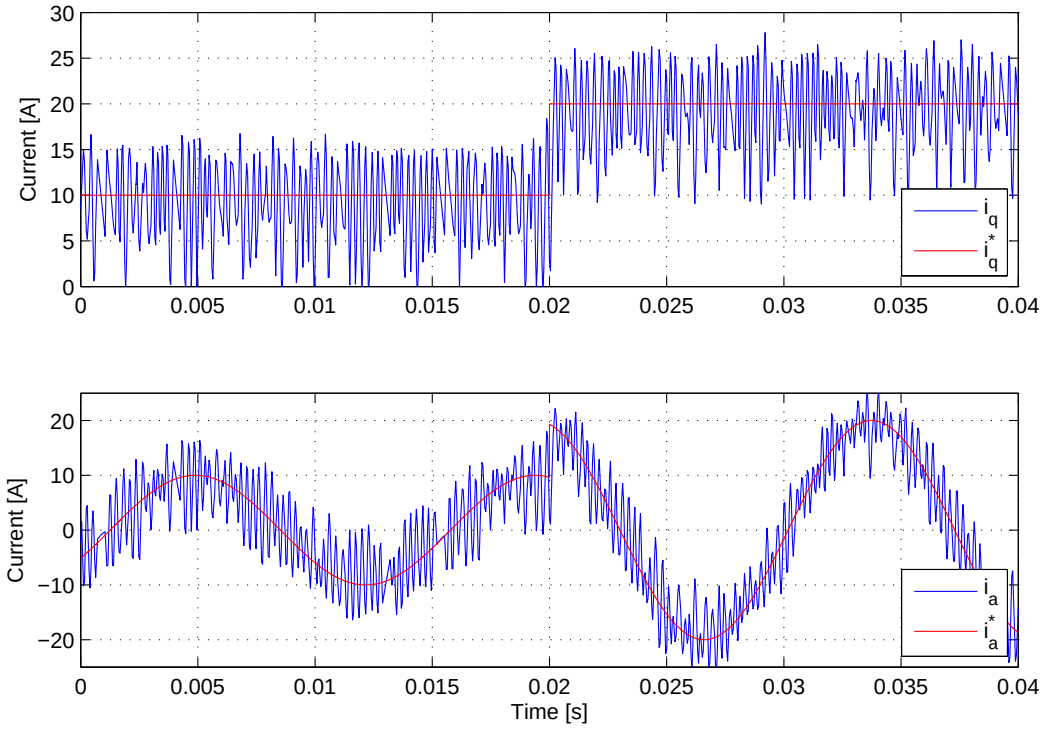


Figure 5.23. Simulated current step response from 10 A to 20 A for the delayed system, running at 1000 rpm

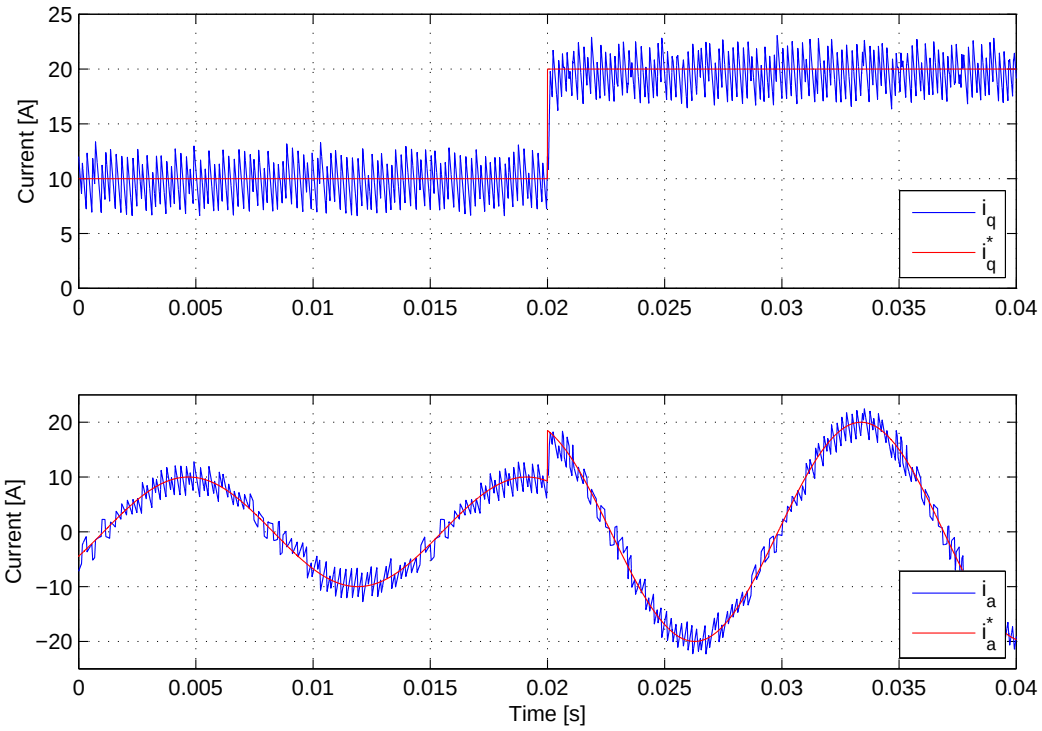


Figure 5.24. Simulated current step response from 10 A to 20 A with delay compensation, running at 1000 rpm

The rising time for both the delay compensated and uncompensated implementations are 70 μs, again limited by the inductance but with a higher value due to the speed increase, now adding the speed dependent term in Equation 5.14. The overshoot is, again, 0% for both. Besides, with the delay compensation, the ripple has been reduced from 17.44 A to 6.34 A. It should be noticed that the current ripple is independent of the speed since the ripple value have not change from Figure 5.22 to 5.24. The steady state error is measured as 8.94% for the uncompensated controller and 2.50 % for the one with delay compensation.

For further verification of the final delay compensated model, Figure 5.25 shows a step from 10 A to -10 A. The motor current rising time is 90 μs, but still with a steady state error of 2.50% and current ripple of 5.986 A

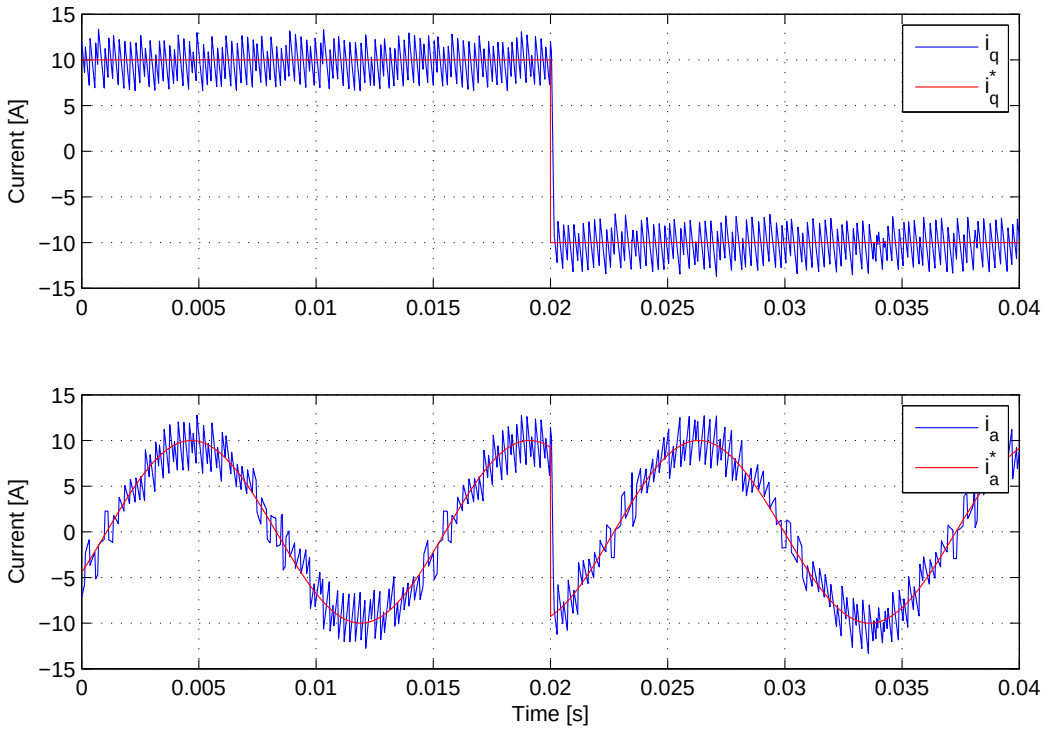


Figure 5.25. Simulated current step response from 10 A to -10 A with delay compensation, running at 1000 rpm

In the previous, only delay compensation was implemented. It should be possible to decrease the steady state error if the dead-time compensation proposed in Section 5.3 is implemented. The observed steady error for 1000 rpm without dead-time compensation is already small in the ranges of the measured for FOC and smaller than for deadbeat with inverter dead-time included. It was considered that the increase of complexity in the code and computation time was not worthy for this case. However, dead-time compensation was implemented and its effect was validated comparing a totally uncompensated MPC controller and one which included dead-time compensation. Both systems perform similarly and only the steady state is affected.

A similar performance to the uncompensated system with an improvement of the steady state error was obtained. Firstly, a step from 10 A to 20 A was commanded. With the non-compensated control method, the steady state error at 10 A was measured to be 2.74% and 1.74% for 20 A. With dead-time compensation, the error is reduced to 0.611% and 0.5982% and the steady state is, therefore improved.

According to the simulations, the current ripple introduced by the MPC is much higher than for deadbeat. This is a big drawback. If MPC has to be comparable with deadbeat, some technique fore reducing the current ripple should be introduced, such as an increase of the sampling frequency or the addition of some kind of modulation scheme. In [Morel et al., 2009] a simple modulation approach called Two-Configuration Predictive Control

was introduced and a similar approach was also given in [Drobnic et al., 2009] and [Nemec et al., 2009] in order to minimise the current ripple.

It was observed that at low current reference or too low sampling frequency, the MPC does not work properly, giving out just zero vectors. If one voltage vector is applied for a full sampling time and if the sampling periods are too long or if the reference is too low, then the resulting current will greatly deviated from the reference by the end of the sampling period. Therefore any of active vector will never produce a minimum for the cost function which will apply only zero vectors. Only the transient response is still faster for MPC, however, this is due to the much higher sampling frequency which is needed. It can be conclude that for this specific system deadbeat is the best option and the only predictive control that it was possible to implement on the available hardware.

Predictive control parameter sensitibity 6

As mentioned, one of the main characteristics of predictive control is that it is based on a good knowledge of the model. Once a satisfactory Matlab/Simulink model has been made for each of the presented control schemes, it is found of interest to add a final chapter where the predictive control schemes can be validated regarding its robustness and sensitivity against uncertainties in the model parameters.

For the classical control theory, which is used in the FOC, an analytical approach can be used to determine the effects of parameter uncertainties and variations. A change in the plant can simply be analysed by looking for instance at pole zero location or the step response. For the deadbeat and the MPC described in this report, the classical control theory, based on Laplace transformation to the frequency domain, can not be applied since they use a discrete time model. Therefore, this chapter will be based on the simulation results.

The simulations data for this comparison uses a non-ideal inverter with a 2.5 μs dead-time. There could be an incongruence between the motor parameters (resistance and inductance) used in the controller implementation and the actual motor value. This can be due to a not precise knowledge of the parameters or possible variations during operation because of, for example, temperature changes. The influence of the parameters error is analysed for changes in the resistance and inductance. In the simulations, variations are applied to one of the parameters while the other is kept at its rated value. This makes it possible to see the effect of the parameter variations separately. The d and q reference currents are 0 A and 10 A respectively, with the motor running at 1000 rpm. The switching frequency for deadbeat and FOC was set as 5 kHz. For the MPC, the sampling frequency is 25 kHz , so that as it was shown in Figure 5.18 the mean switching frequency may be in the around 1.25 kHz.

Current ripple and steady state error are compared using bar charts. The steady state error is calculated with Equation 6.1 where the the difference between the reference and the average of the i_{dq} current for n data points. The ripple is defined by 6.2. i_{dq} is the

simulated current data for a period of 30 ms.

$$\tilde{i}_{dq,ss} = i^*_{dq} - \frac{1}{n}\sum_{j=1}^{n} i_{dq,j} \tag{6.1}$$

$$\Delta i_{dq} = i_{dq,max} - i_{dq,min} \tag{6.2}$$

In Figure 6.2 a simulation with no parameter changes is made to serve as a reference for the comparison. As it has been already seen in previous simulations, MPC has a current ripple two times bigger that FOC and deadbeat. It should be noticed that the average switching frequency for MPC is much smaller than the 5 kHz used for FOC and deadbeat. If MPC were simulated with a 80 kHz sampling frequency, so that the average switching frequency can be considered 5 kHz, then, the current ripple is reduced to 2.14 A which is the same current ripple measured for deadbeat.

When comparing FOC and deadbeat current ripple the values are quite even, this is because they are using the same SVM.

In the right hand graph, steady state error is compared. From this, it is obviously that the main drawback from deadbeat is the steady state error. The steady state error in i_q for deadbeat, is the largest even though, no dead-time compensation was included in either MPC or FOC. Due to the integration part of the PI controller in FOC, the steady state error is minimised. A steady state error of 1.73 A in the i_d current is caused by an uncompensated delay in the rotor position feedback for the deadbeat controller. In Figure 6.1 two different simulations result for deadbeat are presented.

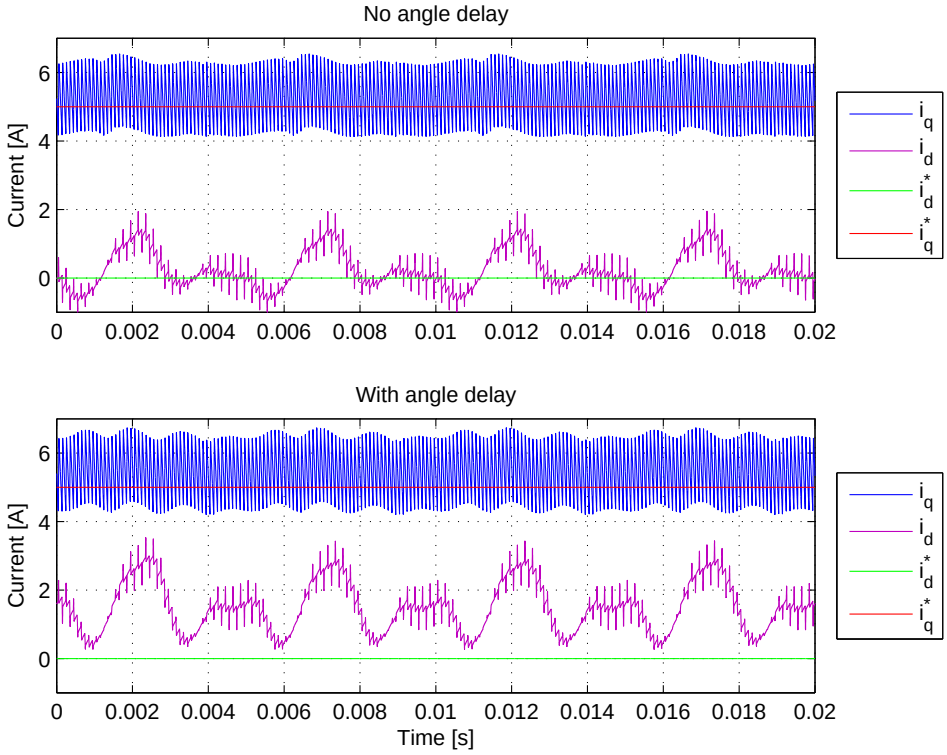


Figure 6.1. Deadbeat simulations with an ideal system without delay and a non-ideal system with a delay added to the angle feedback

In the first plot the simulation is performed with no delay in the control system and the motor running at 1000 rpm. The steady state error for i_d in the non-delayed system is only 0.28 A. The second plot in Figure 6.1, when a delay is added to the rotor position feedback the i_d steady state error increases to 1.73 A. Therefore, at especially high speed, a rotor position estimation is needed to improve steady state performance for deadbeat controller. This has not be investigated in this report.

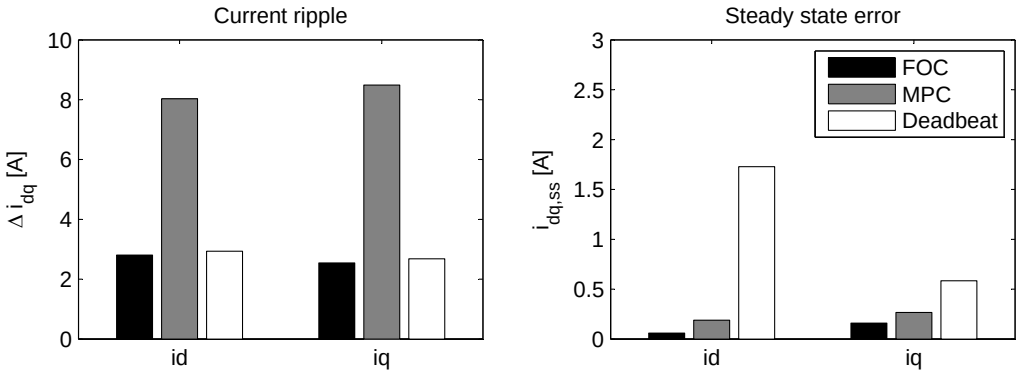


Figure 6.2. Steady state error and and peak-to-peak current ripple with no parameter variations

The resistance is temperature dependent and therefore, the motor model resistance have

been decreased and increased with 50% in Figure 6.3 and6.4 respectively. The current ripple is decreased for MPC and maintained for the others. When it comes to steady state error the deadbeat is the most sensible to resistance error.

Deadbeat is based on Equation 4.1, direct application of the motor drive model. If there is an error in the resistance, the calculated reference voltage $\bar{u}_{dq}(k)$, will give a different current response $\bar{i}_{dq}(k+1)$, than the desired. The actual steady state current will be either higher or lower than the used in the controller motor model, adding up or decreasing the steady state error as it may be observed in the Figures 6.3 and 6.4, respectively.

A different case for the MPC, where a finite set of inverter voltage vectors are being taking into account and the current error is evaluated using a cost function. A variation in the parameters will deviate the obtained $\bar{i}_{dq}$ position from the predicted value, this is why the current ripple is affected. However, each of the possibilities from the set of inverter voltage vectors are deviated in the same amount so that the average current value, and therefore, the steady state error, is not affected so much.

The steady state error for FOC is not affected because it is not using a motor model in order to obtain the voltage reference.

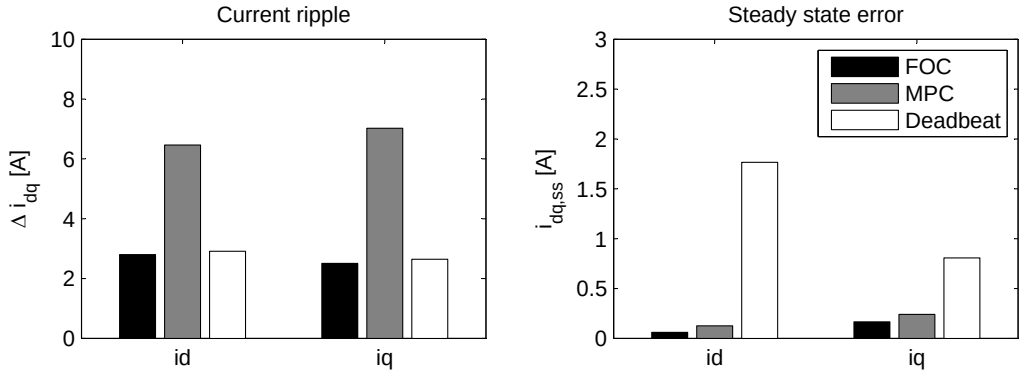


Figure 6.3. Steady state error and and peak-to-peak current ripple with half of the rated resistance

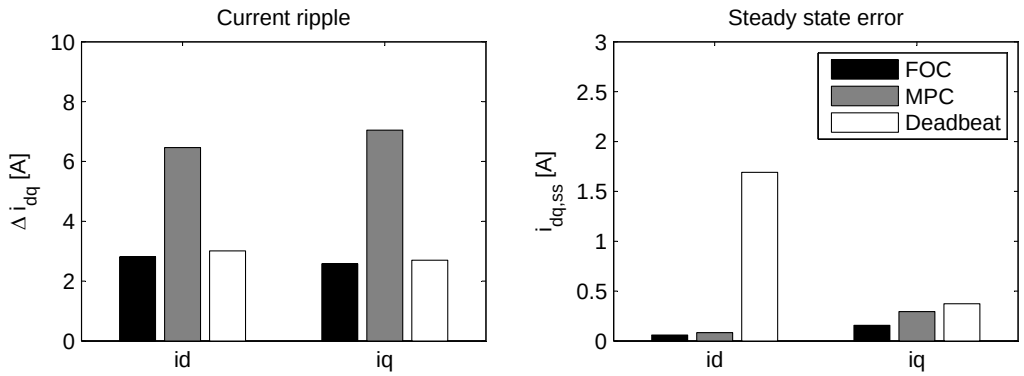


Figure 6.4. Steady state error and and peak-to-peak current ripple with a 50 % increase of the rated resistance

From Figure 6.5 and 6.6 the inductance is decreased and increase with 50 % respectively. It should be noticed that in Figure 6.5, the y-axis scale for the current ripple is changed. A variation of the inductance is, for all the cases, inverse proportional to the current ripple. The current ripple decreases when the inductance increases and viceversa. FOC and deadbeat variates evenly. However, the most affected is clearly MPC. As mention before, the use of a finite set for MPC make it really sensitive to parameter uncertainties, mainly affecting the current ripple. Besides, a low inductance in the motor increases the di/dt, which directly result in a big deviation from the reference, it is seen that the ripple triplicates the current reference of 10 A for a halved inductance.

Looking at the steady state error both MPC and deadbeat are affected. Again deadbeat is the most affected.

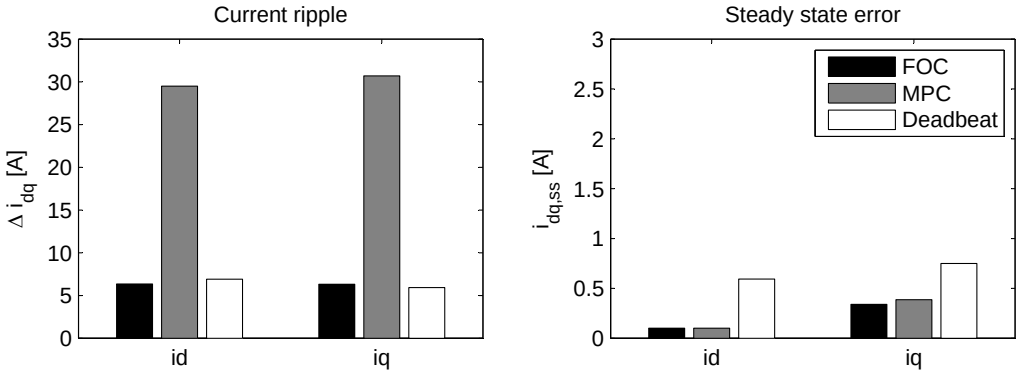


Figure 6.5. Steady state error and and peak-to-peak current ripple with half of the rated inductance

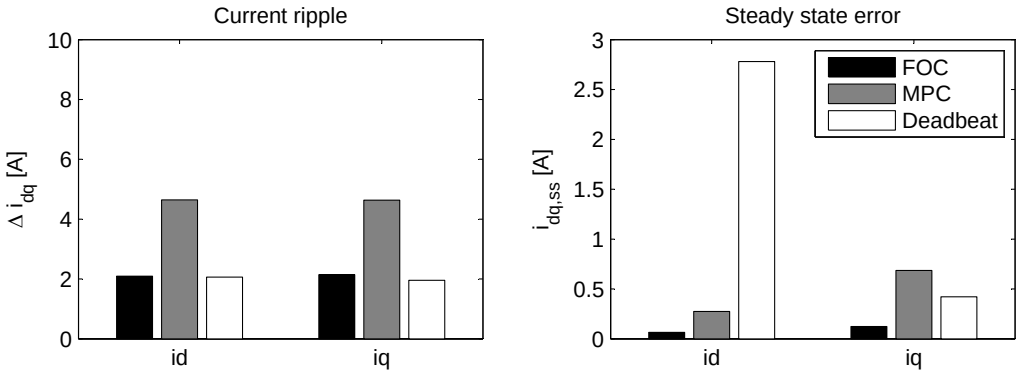


Figure 6.6. Steady state error and and peak-to-peak current ripple with double of the rated inductance

It could be concluded that if the motor parameter varies from the used in the controller design, FOC barely varies its performance. However, the predictive controllers are noticeable affected. Deadbeat always shows the biggest steady state error and it is also in the steady state error where the main variations happens. However, in any of the cases, the deadbeat steady state error never surpass 1%. MPC is the most affected for

uncertainties in the parameters, in this case, the biggest performance variation is observe with inductance variations.

The behaviour of the predictive controllers, when error in the model parameters, could be improved using parameter estimation techniques for on-line estimation of the model parameter in order to get a more accurate result.

Conclusion 7

This project is comprised of the analysis, design and implementation of Predictive Current Control for a PMSM. The goal was to obtain an alternative to the classical FOC method with improved dynamics.

Deadbeat and MPC were the proposed predictive control methods for this report. Matlab/Simulink models for the two predictive controllers and the FOC were successfully developed. Simulations were run at both zero speed and 1000 rpm.

Inverter dead-time was included in the model. The voltage error due to the inverter dead-time causes steady state error. The integral part of FOC works to decrease this error. Therefore the FOC simulation and experimental results have the lowest steady state error of the implemented methods, with a range from almost zero to 3.15%. MPC shows an small steady state error which could be reduced with dead-time compensation to values under the 1%. However, because of the complexity introduced by the dead-time compensation, the code is not used for MPC. Whereas, for deadbeat, inverter dead-time compensation is critical, the error is reduced in more than 10%. Overall, the steady state error for deadbeat is the biggest of the tested control methods but its result is still considered satisfactory.

The calculation delay can degrade the predictive control performance. If the delay compensation is implemented in simulations, the current ripple for MPC is reduced in a 65% of its value and for deadbeat the overshoot is reduced in more than a 90% of the initial overshoot value. However, even with the calculation delay compensation, the current ripple for MPC is still significantly bigger than for deadbeat with a sampling frequency of 25 kHz.

The MPC is operated without a modulator, so the switching frequency is not constant. It has been proved that MPC requires a sampling frequency in the ranges of 80 kHz in order to obtain mean switching frequencies of 5 kHz, which was the switching frequency chosen for deadbeat and FOC. It should be noticed that with a sampling frequency of 80 kHz for MPC the current ripple is the same as deadbeat and FOC. Deadbeat has the advantage of using SVM which implies a fix switching frequency. Another advantage of Deadbeat is

that it can be directly substituted with the PI controller from the FOC scheme.

The given laboratory set-up imposes a maximum sampling frequency of 27.77 kHz from the micro-controller. With a sampling frequency of 25 kHz, MPC has shown an unacceptable high current ripple. Therefore, deadbeat is the predictive control chosen for the implementation in the laboratory with dSpace. The simulation models for FOC and deadbeat were verified experimentally.

It can be said that the predictive controllers implementation is intuitive and relatively easy compared with the classic schemes. However, the predictive controller is based on a discrete model of the motor drive in order to be able to predict the future value. Therefore, the main difficulty comes in the precision required in model parameters. In the last chapter of this project, the sensitivity to motor parameter errors in the controller implementation is analysed. High sensitivity is a new drawback for MPC. Deadbeat sensitivity is reflected in an increase of the steady state error, however, for the tested conditions it never surpasses 1% and the transient response is never affected.

According to the experimental results, although in steady state FOC is still better, deadbeat has an acceptable steady state performance with a very low current ripple and an improved transient response, with no overshoot and a rising time 5 times faster than the FOC. Deadbeat is proved as a good alternative to FOC with a improved torque response and dynamic performance.

7.1 Future Work

Deadbeat has shown to be a substantial improvement in the transient response compared to the classical FOC scheme. The deadbeat predictive controller provides current ripple that can compete the FOC. However, it is hard for deadbeat to improve upon the low steady state error found in FOC. The designed deadbeat controller included dead-time compensation, however it neglected the voltage error due to the voltage drop in transistors and diodes and the on-off switching time in the transistors, this is because their influence is smaller and their value is variable and difficult to obtain. For a maximised accuracy it would be of interest the development of an on-line voltage error estimation which would include that inverter non-linearities.

Deadbeat is simple and it fulfils the goals regarding dynamic response but MPC could also turn out to be an interesting option. The MPC can provide at least the same transient response as deadbeat and in order to improve the steady state response, the ripple can be reduced by using some modulation technique as proposed in [Morel et al., 2009], [Nemec et al., 2009] and [Drobnic et al., 2009]. The modulator would offer an advantageous fix frequency as well.

Besides, it should be noticed that a smaller DC-link voltage, a higher sampling frequency or a higher motor inductance are all factors which would improve the MPC current ripple.

Any of the factor helps to make the current to variate less and therefore, they facilitate a more precise tracking of the current reference. Therefore, in another test set-up with those characteristics MPC and deadbeat could be more fairly compared. Unlike in deadbeat, in case of implementing MPC, thanks to the cost function, new constraints could be added, such as switching losses minimisation.

It has been mentioned the importance of the parameter accuracy in order to design a good predictive controller. An on-line estimation of the model parameters would be used so that they can be updated in real time in the controller motor model.

Parameter determination A

This appendix will describe how to measure the inductance and resistance for the PMSM. This serves as verification of the values provided from the datasheet.

The L_d inductance can be determined with the rotor in zero position ($\theta = 0$). At zero position the d-axis is aligned with phase a. If the motor is connected as illustrated in Figure A.1 with a DC power supply or the inverter, the voltages and currents becomes:

$$v_a = -2v_b = -2v_c \tag{A.1}$$

$$i_a = -2i_b = -2i_c \tag{A.2}$$

$$v_{ab} = \frac{3}{2}v_d \tag{A.3}$$

$$i_a = i_d \tag{A.4}$$

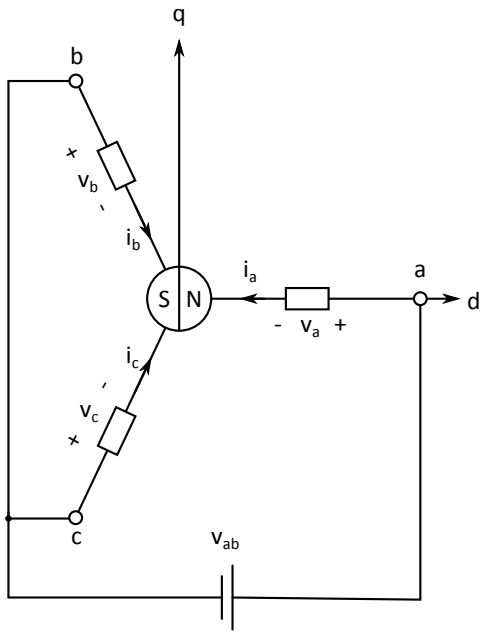


Figure A.1.

If only a d-axis current is applied, the motor will not produce any torque, keeping the rotor locked in zero position. With a zero speed and q-axis voltage and current set to

zero the d-axis voltage equation is simplified as shown in Equation A.5. Since it is not possible to measure v_d and i_d directly with an oscilloscope, Equation A.6 replaces these d-axis voltage and currents with a measurable line-to-line voltage and phase currents with respect to Equation A.3 and A.4.

$$v_d = Ri_d + L_d \frac{di_d}{dt} \tag{A.5}$$

$$\Updownarrow$$

$$v_{ab} = \frac{3}{2} Ri_a + \frac{3}{2} L_d \frac{di_a}{dt} \tag{A.6}$$

When applying a step voltage with a Delta-Elektronika DC power supply, the stator resistance is determine by measuring the line-to-line voltage v_{ab} and phase a current in steady state as shown in Equation A.7. v_{ab} and i_a was measured in the lab to be 1.901 V with a Fluke 179 multimeter and 6.24 A with a Tektronix oscilloscope. This makes the resistance of a single phase 0.203 Ω, including the cable from the inverter to the motor. The resistance value from the datasheet is 0.18 Ω without a cable.

$$R = \frac{\frac{2}{3} v_{ab,ss}}{i_{a,ss}} \tag{A.7}$$

The inductance can be determine by applying a step voltage to the motor, which behaves as a 1st order RL circuit. The measured system response shown in Figure A.2 is used to determine the electrical time constant τ, which represents the time it takes the current response to reach approximately 63.2 % of it's steady state value. τ is 10.2 ms as shown in Figure A.2, this gives a inductance value of 2.1 mH using Equation A.8. According to the datacheet the field inductance is 2.0 mH which almost fits with the measured inductance.

For verification of the simulink model parameter, the d-axis current response was compared with the a measure response from the lab, under the same conditions. The result of the simulation is presented in Figure A.2 with the measured response. There is a good agreement between the simulation and the measurement. Therefor the the found electrical parameters for the PMSM is found valid.

$$L_d = \tau R \tag{A.8}$$

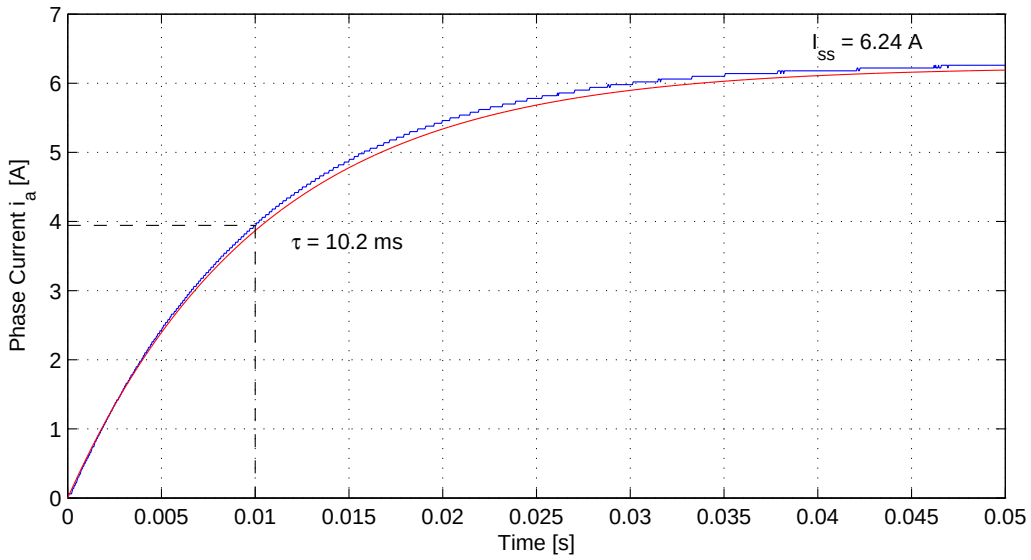


Figure A.2. Measured phase a current response with with a Tektronix oscilloscope

FOC Simulink model B

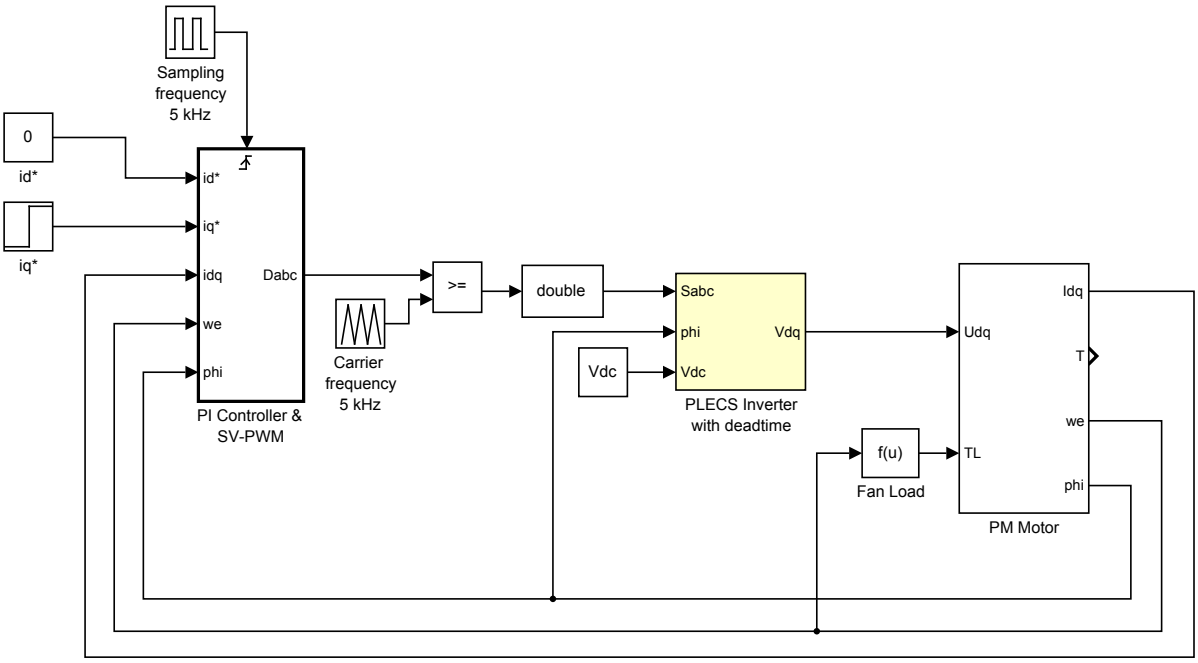


Figure B.1. FOC complete model

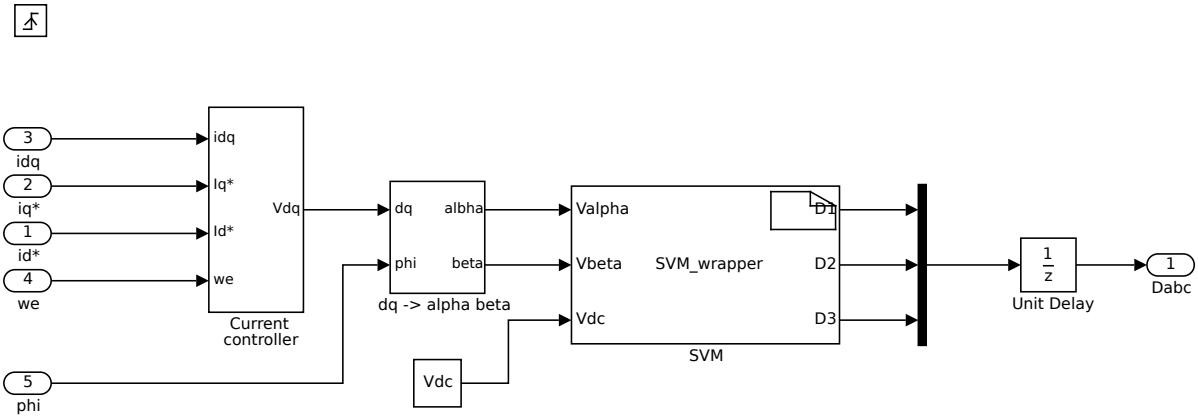


Figure B.2. FOC current controller model

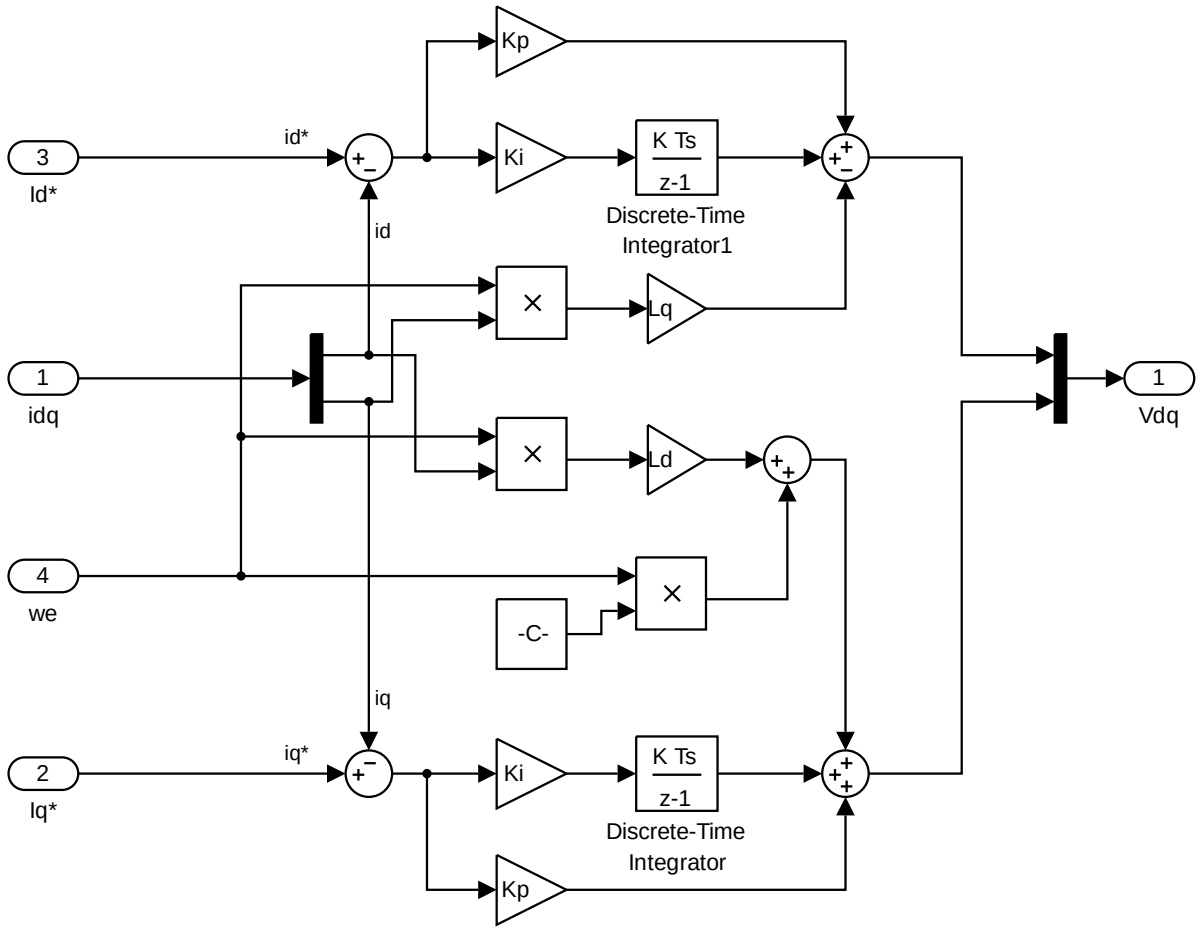


Figure B.3. PI controller model

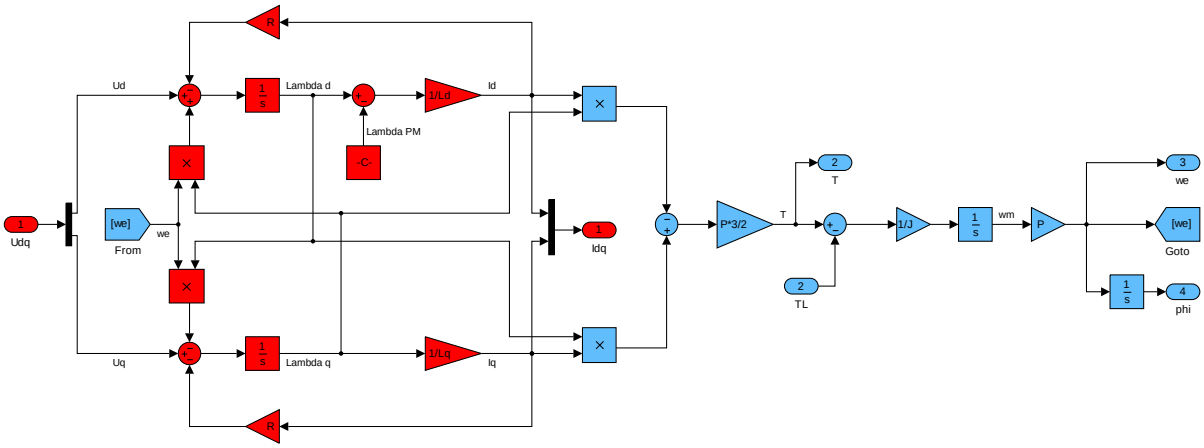


Figure B.4. PMSMs model

Matlab code for deadbeat controller

C

```
function udq = fcn(idref, iqref, id, iq, w, Vdc, R, Ld, Lq, Lambda_PM,↙
Ts1, Td, phi)
%% Initialisation
global udq01;

%% Motor Model (Estimated current vector for k+1)
id1 = (1 - (R*Ts1)/Ld)*id + (Ts1*udq01(1,1))/Ld + (w*Lq*Ts1*iq)/Ld;
iq1 = (1 - (R*Ts1)/Lq)*iq + (Ts1*udq01(2,1))/Lq - (w*Ld*Ts1*id)/Lq -↙
(w*Ts1*Lambda_PM)/Lq;

%% Deadbeat control (voltage output for k+1)
ud1p = Ld/Ts1*(idref - (1-(R*Ts1)/Ld)*id1 - ((w*Ld*Ts1)/Ld)*iq1);
uq1p = Lq/Ts1*(iqref - (1-(R*Ts1)/Lq)*iq1 + ((w*Ld*Ts1)/Lq)*id1 +↙
w*Ts1*Lambda_PM/Lq);

%% Voltage Compensation
delta_u = (Td/Ts1)*(Vdc);

delta_ud = (4/pi)*delta_u*((12/35)*sin(6*phi)+(24/143)*sin(12*phi));
delta_uq = (4/pi)*delta_u*(-1+(2/35)*cos(6*phi)+(2/143)*cos(12*phi));

% delta_u sign depending on iq sign
if iqref >= 0
    a = -1;
else
    a = 1;
end

if w > 0 && iqref < 0
    b = -1;
else
    b = 1;
end
```

```
%Compensated voltage output for k+1
ud1 = b*a*delta_ud+ud1p;
uq1 = b*a*delta_uq+uq1p;

%% Voltage restriction (linear area maximum amplitude- Vdc/sqrt(3))
if sqrt(ud1^2+uq1^2)>Vdc/sqrt(3)
    ud1 = ((Vdc/sqrt(3))/sqrt(ud1^2+uq1^2))*ud1;
    uq1 = ((Vdc/sqrt(3))/sqrt(ud1^2+uq1^2))*uq1;
    udq1 = [ud1; uq1];
else

    udq1 = [ud1; uq1];
end

 %% Save current state
 udq01 = [ud1p;uq1p] ;
 udq = udq1;
```

Matlab code for MPC D

D.1 Delay compensation

```
function Sabc_now = fcn(idref, iqref, id, iq, w, phi, Vdc, R, Ld, Lq,↙
Lambda_PM, Ts, imax)
%% Initialisation
persistent state_1;
if isempty(state_1)
    state_1 = 1;
end

%% Switching functions
S = [0 0 0; 1 0 0; 1 1 0; 0 1 0; 0 1 1; 0 0 1; 1 0 1; 1 1 1];

%% Voltage applied k+1 (I use the voltage I just applied to calculate↙
the current at the end of the period)
ud = Vdc*2/3*( S(state_1,1)*cos(phi) + S(state_1,2)*cos(phi - 2*pi/3) +↙
S(state_1,3)*cos(phi + 2*pi/3));
uq = Vdc*2/3*(-S(state_1,1)*sin(phi) - S(state_1,2)*sin(phi - 2*pi/3) -↙
S(state_1,3)*sin(phi + 2*pi/3));

%% Motor Model (Estimated current vector for k+1)
id1 = (1 - (R*Ts)/Ld)*id + (Ts*ud)/Ld + (w*Lq*Ts*iq)/Ld ;
iq1 = (1 - (R*Ts)/Lq)*iq + (Ts*uq)/Lq - (w*Ld*Ts*id)/Lq -↙
(w*Ts*Lambda_PM)/Lq;

%% Terminal Voltage (dq transformation)
ud1 = Vdc*2/3*( S(:,1)*cos(phi) + S(:,2)*cos(phi - 2*pi/3) + S(:,3)*cos↙
(phi + 2*pi/3));
uq1 = Vdc*2/3*(-S(:,1)*sin(phi) - S(:,2)*sin(phi - 2*pi/3) - S(:,3)*sin↙
(phi + 2*pi/3));

%% Motor Model (Predicted current values for k+2)
idp2 = (1 - (R*Ts)/Ld)*id1 + (Ts*ud1)/Ld + (w*Lq*Ts*iq1)/Ld ;
iqp2 = (1 - (R*Ts)/Lq)*iq1 + (Ts*uq1)/Lq - (w*Ld*Ts*id1)/Lq -↙
(w*Ts*Lambda_PM)/Lq;
```

```
%% Cost Function's
% Uncomment to use cost function with absolute error
% g = abs(idref-idp2) + abs(iqref-iqp2); % Standart current control↙
cost function with absolut values

% Uncomment to use cost function with the error squared
% g = (idref-idp2).^2 + (iqref-iqp2).^2; % Current control cost↙
function which gives better refrence
% tracking when more termes is added to the cost function compaed with↙
the using of absolut values.

% Uncomment to use cost function with switching reduction
% n = abs(S(:,1) - S(state_1,1)) + abs(S(:,2) - S(state_1,2)) + abs(S↙
(:,3) - S(state_1,3));
% g = (idref-idp).^2 + (iqref-iqp).^2 + 35*n; % Current control with↙
minimization of switching
% frequency. The weighting factor is emperical dertermined and if↙
increased the frequency decreases
%(for this motoer a typical values would be between 20-40. The range is↙
from 0-??)

% Hard constraints of stator current
f = zeros(8,1);
for j = 1:8
    if abs(idp2(j,1)) > imax || abs(iqp2(j,1)) > imax
        f(j,1) = inf;
    end
end
g = (idref-idp2).^2 + (iqref-iqp2).^2 + f;

%% Finds the vector index with the minimum value
m = find(g == min(g),1);
state = m(1,1);

%% State change minimizer
if state == 2 || state == 3 || state == 4 || state == 5 || state == 6↙
|| state == 7

elseif state_1 == 2 || state_1 == 4 || state_1 == 6 || state_1 == 1
    state = 1;
else
    state = 8;
end

%% Save current state
state_1 = state;

%For testing like dSpace we apply the state one period before and we↙
delay the signal like dSpace would do
state_now = state;
Sabc_now = S(state_now,:).';
```

D.2 Inverter dead-time compensation

```
function Sabc = fcn(idref, iqref, id, iq, w, phi, Vdc, R, Ld, Lq,↙
Lambda_PM, Ts, Td, imax)
%% Initialisation
persistent state_1;
if isempty(state_1)
    state_1 = 1;
end
g_optimal = inf;
state = 0;

%% Switching function
S = [0 0 0; 1 0 0; 1 1 0; 0 1 0; 0 1 1; 0 0 1; 1 0 1; 1 1 1];

%% Phases[a b c] sing (plus(p), minus(m)) depending on the  6 sectors
PS=[1 -1 -1;1 1 -1;-1 1 -1;-1 1 1; -1 -1 1; 1 -1 1];
% Locate phi in its section
if phi<0
    phi=2*pi+phi;
end
if phi>2*pi
    phi=phi-2*pi;
end

% Locate the current in the sector (phases sign)
if (11*pi/6)<phi||phi<=(pi/6)
    m= PS(1,:);
elseif(pi/6)<phi||phi<=(pi/2)
    m=PS(2,:);
elseif(pi/2)<phi||phi<=(5*pi/6)
    m=PS(3,:);
elseif(5*pi/6)<phi||phi <=(7*pi/6)
    m=PS(4,:);
elseif(7*pi/6)<phi||phi <=(3*pi/2)
    m=PS(4,:);
else
    m=PS(6,:);
end

% Trigonometric functions
cos_1 = cos(phi);
sin_1 = sin(phi);

%Define A, B and C
idq = [id; iq];
```

```
A = [1-(R*Ts/Ld) Ts*w*Lq/Ld; -Ts*w*Ld/Lq 1-(R*Ts/Lq)];
B = [Ts/Ld 0; 0 Ts/Lq];
C = [0; -(Ts*w*Lambda_PM)/Lq];

M = [cos_1 sin_1; -sin_1 cos_1];
D = (Vdc*(2/3))*[1 -0.5 -0.5; 0 sqrt(3)/2 -sqrt(3)/2];

for j = 1:8
    %Obtain dead-time switching states

    if S(j,1)==S(state_1,1)
        Sdta= S(j,1);
    elseif S(state_1,1)==1
        if m(1)== 1
            Sdta= 1;
        else
            Sdta= 0;
        end
    else
       if m(1)== 1
            Sdta= 0;
        else
            Sdta= 1;
        end
    end

    if S(j,2)==S(state_1,2)
        Sdtb=S(j,2);
    elseif S(state_1,2)==1
        if m(1)== 1
            Sdtb= 1;
        else
            Sdtb= 0;
        end
    else
       if m(1)== 1
            Sdtb= 0;
        else
            Sdtb= 1;
        end
    end

    if S(j,3)==S(state_1,3)
        Sdtc= S(j,3);
```

```
    elseif S(state_1,3)==1
        if m(1)== 1
            Sdtc= 1;
        else
            Sdtc= 0;
        end
    else
       if m(1)== 1
            Sdtc= 0;
        else
            Sdtc= 1;
        end
    end

    Sdt=[Sdta Sdtb Sdtc];
    udt = M*D*Sdt.';
    udq = udt*(Td/Ts)+((Ts-Td)/Ts)*M*D*S(j,:).';
    idqp = A*idq+B*udq+C;

    %% Cost Function
    % Hard constraints of stator current
    f = 0;
    if abs(idqp(1,1)) > imax || abs(idqp(2,1)) > imax
        f = inf;
    end
    g = (idref-idqp(1,1)).^2 + (iqref-idqp(2,1)).^2 + f;

    if g < g_optimal
       g_optimal = g;
       state = j;
    end;
end

%% State change minimizer
if state == 2 || state == 3 || state == 4 || state == 5 || state == 6↙
|| state == 7
    Sabc = S(state,:).'; % Normal output
elseif state_1 == 2 || state_1 == 4 || state_1 == 6 || state_1 == 1
    state = 1;
    Sabc = S(state,:).';
else
    state = 8;
    Sabc = S(state,:).';
end


%% Save current state
state_1 = state;
```

Bibliography

Cortes, Kouro, La Rocca, Vargas, Rodriguez, Leon, Vazquez, and Franquelo, 2009. P. Cortes, S. Kouro, B. La Rocca, R. Vargas, J. Rodriguez, J.I. Leon, S. Vazquez, and L.G. Franquelo. *Guidelines for weighting factors design in Model Predictive Control of power converters and drives.* Industrial Technology, 2009. ICIT 2009. IEEE International Conference on, pages 1–7, 2009. doi: 10.1109/ICIT.2009.4939742.

Cortes, Rodriguez, Silva, and Flores, 2012. Patricio Cortes, Jose Rodriguez, Cesar Silva, and Alexis Flores. *Delay compensation in model predictive current control of a three-phase inverter.* Industrial Electronics, IEEE Transactions on, 59(2), 1323–1325, 2012.

Silva, Santos, and Jacobina, 2011. Edison Roberto C da Silva, E Cipriano dos Santos, and Cursino Brandao Jacobina. *Pulsewidth modulation strategies.* Industrial Electronics Magazine, IEEE, 5(2), 37–45, 2011.

Drobnic, Nemec, Nedeljkovic, and Ambrozic, 2009. Klemen Drobnic, Mitja Nemec, David Nedeljkovic, and Vanja Ambrozic. *Predictive direct control applied to AC drives and active power filter.* Industrial Electronics, IEEE Transactions on, 56 (6), 1884–1893, 2009.

Fitzgerald, Charles Kingsley, and Umans, 2002. A. E. Fitzgerald, Jr. Charles Kingsley, and Stephen D. Umans. *Electric Machinery.* ISBN: 978-0-07-366009-7. McGraw-Hill, 6th edition edition, 2002.

Hwang and Kim, 2010. Seon-Hwan Hwang and Jang-Mok Kim. *Dead time compensation method for voltage-fed PWM inverter.* Energy Conversion, IEEE Transactions on, 25(1), 1–10, 2010.

Imura, Takahashi, Fujitsuna, Zanma, and Doki, 2012. Akihiro Imura, Tomoya Takahashi, Masami Fujitsuna, Tadanao Zanma, and Shinji Doki. *Dead-time compensation in model predictive instantaneous-current control.* pages 5037–5042, 2012.

Iqbal, Lamine, Ashra, et al., 2006. Atif Iqbal, A Lamine, I Ashra, et al. *Matlab/Simulink Model of Space Vector PWM for Three-Phase Voltage Source Inverter.* 3, 1096–1100, 2006.

Irwin, Kazmierkowski, Krishnan, and Blaabjerg, **2002**. JD Irwin, Marian P Kazmierkowski, Ramu Krishnan, and Frede Blaabjerg. *Control in power electronics: selected problems*. Academic press, 2002.

Jones, Levi, and Dujic, **2009**. M. Jones, E. Levi, and D. Dujic. *The Impact of inverter dead time on performance of n-motor drives supplied from (2n+1)-leg VSI*. Industrial Electronics, 2009. IECON '09. 35th Annual Conference of IEEE, pages 1350–1355, 2009. ISSN 1553-572X. doi: 10.1109/IECON.2009.5414707.

Kennel, Kazmierkowski, Rodriguez, and Cortes, **2008**. R.M. Kennel, M. Kazmierkowski, J. Rodriguez, and P. Cortes. *Predictive control in power electronics and drives*. Industrial Electronics, 2008. ISIE 2008. IEEE International Symposium on, pages 1–90, 2008. doi: 10.1109/ISIE.2008.4677323.

Kim, Lee, Rho, and Park, **2010**. Sam-Young Kim, Wootaik Lee, Min-Sik Rho, and Seung-Yub Park. *Effective dead-time compensation using a simple vectorial disturbance estimator in PMSM drives*. Industrial Electronics, IEEE Transactions on, 57(5), 1609–1614, 2010.

Kouro, Cortés, Vargas, Ammann, and Rodríguez, **2009**. Samir Kouro, Patricio Cortés, René Vargas, Ulrich Ammann, and José Rodríguez. *Model predictive control. A simple and powerful method to control power converters*. Industrial Electronics, IEEE Transactions on, 56(6), 1826–1838, 2009.

Li, Ming, and Dian-guo, **2012**. Niu Li, Yang Ming, and Xu Dian-guo. *Deadbeat predictive current control for PMSM*. pages LS6b.1–1–LS6b.1–6, 2012. doi: 10.1109/EPEPEMC.2012.6397489.

Lipo, **1996**. Thomas A Lipo. *Vector control and dynamics of AC drives*, volume 41. Oxford University Press, USA, 1996.

Moon, Kim, and Youn, **2003**. Hyung-Tae Moon, Hyun-Soo Kim, and Myung-Joong Youn. *A discrete-time predictive current control for PMSM*. Power Electronics, IEEE Transactions on, 18(1), 464–472, 2003.

Morel, Lin-Shi, Retif, Allard, and Buttay, **2009**. F. Morel, Xuefang Lin-Shi, J.-M. Retif, B. Allard, and C. Buttay. *A Comparative Study of Predictive Current Control Schemes for a Permanent-Magnet Synchronous Machine Drive*. Industrial Electronics, IEEE Transactions on, 56(7), 2715–2728, 2009. ISSN 0278-0046. doi: 10.1109/TIE.2009.2018429.

Munoz and Lipo, **1999**. Alfredo R Munoz and Thomas A Lipo. *On-line dead-time compensation technique for open-loop PWM-VSI drives*. Power Electronics, IEEE Transactions on, 14(4), 683–689, 1999.

Nemec, Drobnic, Nedeljkovic, and Ambrozic, **2009**. Mitja Nemec, Klemen Drobnic, David Nedeljkovic, and V Ambrozic. *Direct current control of a synchronous machine in field coordinates.* Industrial Electronics, IEEE Transactions on, 56(10), 4052–4061, 2009.

Novotny and Lipo, **1996**. D. W. Novotny and T. A. Lipo. *Vector Control and Dynamics of AC Drives.* ISBN: 978-0-19-856439-3. Oxford Science Publications, 1th edition edition, 1996.

Rodriguez and Cortes, **2012**. Jose Rodriguez and Patricio Cortes. *Predictive control of power converters and electrical drives*, volume 37. Wiley-IEEE Press, 2012.

Rodriguez, Kazmierkowski, Espinoza, Zanchetta, Abu-Rub, Young, and Rojas, **2013**. Jose Rodriguez, M Kazmierkowski, J Espinoza, Pericle Zanchetta, Haitham Abu-Rub, H Young, and C Rojas. *State of the Art of Finite Control Set Model Predictive Control in Power Electronics.* 2013.

Sukegawa, Kamiyama, Mizuno, Matsui, and Okuyama, **1991**. Takashi Sukegawa, Kenzo Kamiyama, Katsuhiro Mizuno, Takayuki Matsui, and Toshiaki Okuyama. *Fully digital, vector-controlled PWM VSI-fed AC drives with an inverter dead-time compensation strategy.* Industry Applications, IEEE Transactions on, 27(3), 552–559, 1991.

Summers and Betz, **2004**. T.J. Summers and R.E. Betz. *Dead-time issues in predictive current control.* Industry Applications, IEEE Transactions on, 40(3), 835–844, 2004. ISSN 0093-9994. doi: 10.1109/TIA.2004.827772.

Swierczynski and Kazmierkowski, **2002**. D Swierczynski and MP Kazmierkowski. *Direct torque control of permanent magnet synchronous motor (PMSM) using space vector modulation (DTC-SVM)-simulation and experimental results.* 1, 751–755, 2002.

Urasaki, Senjyu, Uezato, and Funabashi, **2005**. Naomitsu Urasaki, Tomonobu Senjyu, Katsumi Uezato, and Toshihisa Funabashi. *An adaptive dead-time compensation strategy for voltage source inverter fed motor drives.* Power Electronics, IEEE Transactions on, 20(5), 1150–1160, 2005.

Ye and Zhang, **2010**. Xiaoting Ye and Tao Zhang. *Direct torque control of permanent magnet synchronous motor using space vector modulation.* pages 1450–1453, 2010.