

# POLITECNICO DI TORINO

Master course in Mechatronic Engineering



**Politecnico  
di Torino**

Master Degree Thesis

## Real Time Implementation of a First Order Thermal Model for Induction Motor Drives

Supervisors

Prof. Aldo Boglietti

Dr. Sandro Rubino

Dr. Fabio Mandrile

Candidate

Gabriele Micciché

21 April 2023



# Table of Contents

<b>List of Figures</b>	IV
<b>Abstract</b>	VI
<b>1 Introduction</b>	1
1.1 Introduction to Induction Motor . . . . .	1
1.2 Purpose of the thesis . . . . .	2
<b>2 Simulink Model of an Induction Motor</b>	3
2.1 Dynamic model . . . . .	3
2.1.1 Space Vectors for a three-phase machine . . . . .	3
2.1.2 Voltage equations . . . . .	5
2.1.3 Magnetic model . . . . .	7
2.1.4 Speed and Torque estimation . . . . .	9
2.2 Losses of an Induction Motor . . . . .	10
2.2.1 Iron losses . . . . .	10
2.2.2 Joule losses . . . . .	12
2.3 Thermal model . . . . .	13
2.3.1 Second order model . . . . .	13
2.4 Inverter . . . . .	15
2.4.1 Three-phase Inverter Model . . . . .	16
2.5 Simulink representation . . . . .	17
<b>3 Motor control</b>	25
3.1 Vector control . . . . .	26
3.2 Speed estimator through PLL observer . . . . .	27
3.2.1 Flux computation . . . . .	29
3.2.2 Flux computation for low and high frequency . . . . .	30
3.2.3 PI regulator . . . . .	32
3.2.4 Speed control and torque estimation . . . . .	33
3.2.5 Current control and reference voltage computation . . . . .	33

3.2.6	Duty Cycle computation . . . . .	35
3.3	Thermal model and temperature estimation . . . . .	36
3.4	Motor Control implementation in C code . . . . .	37
<b>4</b>	<b>Results</b>	<b>47</b>
4.1	Results validation . . . . .	49
<b>5</b>	<b>Conclusion</b>	<b>54</b>
	<b>Bibliography</b>	<b>57</b>

# List of Figures

1.1	Stator and Rotor of an Induction motor [1]. . . . .	2
2.1	Coordinates transformation [2]. . . . .	5
2.2	Equivalent Circuit of induction motor in $(d,q)$ reference frame [3]. .	6
2.3	Equivalent Circuit of induction motor with iron resistance. . . . .	11
2.4	Second order thermal model circuit of the motor [7]. . . . .	13
2.5	Circuit diagram of a voltage-source three-phase inverter [10]. . . . .	16
2.6	Stator resistance as a function of temperature. . . . .	17
2.7	Rotor resistance as a function of temperature. . . . .	18
2.8	Stator flux computation. . . . .	18
2.9	Rotor flux computation. . . . .	19
2.10	Stator current computation. . . . .	19
2.11	Rotor current computation. . . . .	20
2.12	Magnetizing flux computation. . . . .	20
2.13	Adding of the iron resistance. . . . .	21
2.14	Iron current computation . . . . .	21
2.15	Electromagnetic torque computation. . . . .	22
2.16	Stator Joule losses computation. . . . .	22
2.17	Rotor Joule losses computation. . . . .	23
2.18	Iron losses computation. . . . .	23
2.19	Second order thermal model system. . . . .	24
3.1	Block diagram of a motor control unit (MCU) functioning [12]. . . .	26
3.2	Motor control block diagram [12]. . . . .	27
3.3	PLL structure [18]. . . . .	28
3.4	Current control block diagram [12]. . . . .	34
3.5	MinMax modulation block diagram [10]. . . . .	36
3.6	First order thermal model equivalent circuit [7]. . . . .	36
4.1	Induction motor. . . . .	47
4.2	Block diagram of the motor bench. . . . .	48

4.3	dSPACE Control Desk interface. . . . .	48
4.4	Speed control test. . . . .	49
4.5	Stator current (left) and stator flux (right) in $\alpha\beta$ reference frame. .	50
4.6	Comparison of torques behavior. . . . .	50
4.7	Thermal model validation with a load torque equal to 8 Nm. . . . .	52
4.8	Thermal model validation with a load torque equal to 26 Nm. . . . .	53

# Abstract

Induction motors are often subject to stress caused by repeated fast and short torque transients. These working conditions could lead to overheating some parts of the motor, among which the most sensitive are the stator windings. This thesis deals with the implementation of a temperature estimator of the stator windings obtained through a first order thermal model.

The first step of the project concerns the development of a model which is capable to simulate and evaluate the dynamic and magnetic parameters of the induction motor in different working condition. In particular joule losses, iron losses and magnetic saturation are taken in account and the reference frames used to model the motor are: three-phase time-domain  $(a,b,c)$ , stationary  $(\alpha,\beta)$  and rotating frame  $(d,q)$ . Furthermore analysis and validation of the model were performed in MATLAB/Simulink environment.

Then we moved on to design the torque and speed control of the motor using a "field oriented control" (FOC) scheme. Induction motor speed control is a process of manipulating currents in order to regulate speed. Field-oriented control regulates  $I_d$  and  $I_q$  such that the flux is proportional to  $I_d$  and the torque is proportional to  $I_q$ . The performance of the control strategy is tested and compared on the model already developed on MATLAB/Simulink and then on a real induction motor, in the laboratories of the Politecnico di Torino, comparing the obtained responses with those provided by dSPACE. The thermal model, based on the first order system, was implemented inside the drive control and compared with the temperature values of the windings measured in the laboratories.





# Chapter 1

## Introduction

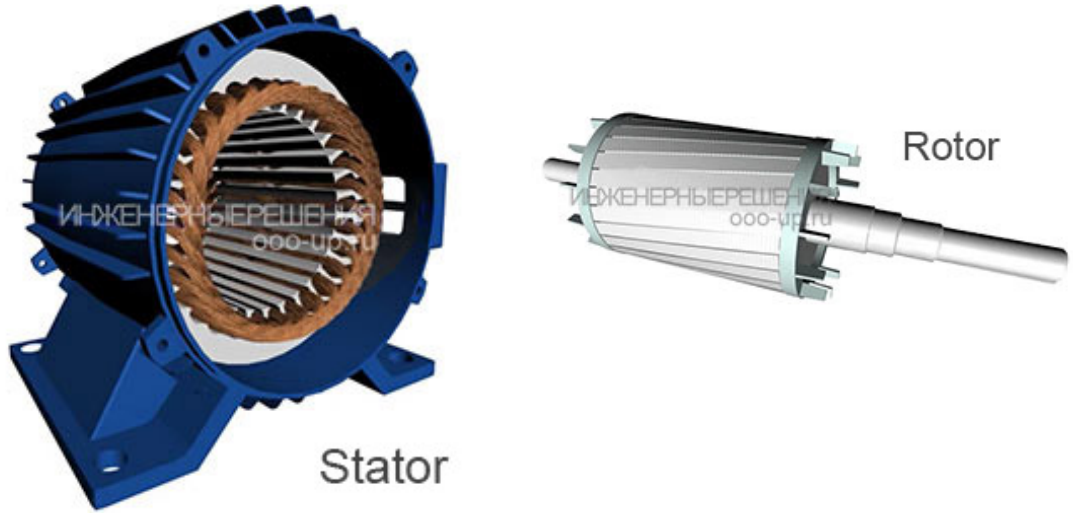
### 1.1 Introduction to Induction Motor

The main function of a motor is to change energy from one form to another like electrical to mechanical. The classification of motors can be done based on the type of supply like AC (Alternating current) motors and DC (Direct current) motors. Induction motor is an AC electrical machine and it is the most common type of motor used in industrial, commercial and residential settings.

An induction motor consists of two main parts: Stator and Rotor. The stator is the stationary portion of the motor and delivers a rotating magnetic field to interact with the rotor. One or more copper windings make up a "pole" within the stator, and there is always an even number of poles within a motor. The electric current alternates through the poles, resulting in a rotating magnetic field. The rotor is the rotating part of the electromagnetic circuit and the central component of the motor, and is fixed to the shaft. The rotor is generally constructed of copper or aluminum strips attached at each end to a circular fixture. This configuration is called a "squirrel-cage rotor" because of its appearance. The magnetic field generated by the stator induces a current in the rotor, which then creates its own magnetic field. The interaction of the magnetic fields in the stator and rotor results in a mechanical torque of the rotor.

Induction motors are classified into two types namely single-phase induction motor and three-phase induction motor. As their name suggests, a single-phase induction motor is connected to a single-phase AC power supply whereas the three-phase induction motor can be connected to a three-phase AC power supply. In a three-phase motor, there are three single-phase lines through a phase difference with  $120^\circ$ . So the rotary magnetic field includes a similar phase difference which

will rotate the rotor. For instance, considering  $a$ ,  $b$  and  $c$  as three phases once phase  $a$  gets magnetized, then the rotor will shift towards the phase  $a$ . In the next second phase,  $b$  will get magnetized, so it will magnetize the rotor and so on for the phase  $c$ . In this way, the rotor will rotate continuously.



**Figure 1.1:** Stator and Rotor of an Induction motor [1].

## 1.2 Purpose of the thesis

The analysis of the induction motor failure reasons show that many of them are caused by prolonged heating of the different parts involved in motor operation. The most sensitive parts of an induction motor to thermal overloads are the stator windings. Exceeding the temperature limit results in acceleration of the oxidation process in insulation materials what eventually leads to motor damage.

Purpose of this thesis is therefore to obtain a thermal model of the stator windings which is able to predict the thermal trend due to heating in induction motor working conditions. This model is based on the relationship between motor power loss and temperature, which can be obtained from experimental data or computer simulations. Obtaining an accurate estimate of the thermal behavior also allows us to understand how much it is possible to exploit the motor in certain conditions without reaching temperatures that could damage the windings and therefore the motor itself.

## Chapter 2

# Simulink Model of an Induction Motor

This chapter focuses on the step-by-step development of the MATLAB/Simulink model of the induction motor: the principles on which the motor is based were studied and explained and subsequently shown how each of these was implemented on Simulink. In particular, the dynamic model of the motor, the losses generated, the thermal model and the inverter that powers the motor were examined.

### 2.1 Dynamic model

The dynamic model of an induction motor is a mathematical representation of the motor's behavior that takes into account the time-varying electrical and mechanical dynamics of the machine. The dynamic model is useful for studying the motor's transient and steady-state behavior, and for designing and optimizing motor control systems.

#### 2.1.1 Space Vectors for a three-phase machine

The space vector representation is a mathematical tool that can be used to analyze the behavior of electrical quantities in three-phase systems. In a three-phase system, the electrical quantities (such as voltages, currents, or fluxes) can be represented by three sinusoidal waves that are 120 degrees out of phase with each other. All phase space vectors related to a phase winding (current, flux) have a well defined position depending on the phase magnetic axis.

To represent an electrical quantity using the space vector representation, the three-phase waveforms are first combined into a complex number. Using for example

the magnetic axes of phase  $a$  as real axis, is defined the complex operator  $\bar{a}$ :

$$\bar{a} = e^{j\frac{2\pi}{3}} = -\frac{1}{2} + j\frac{\sqrt{3}}{2} \quad (2.1)$$

$\bar{a}$  is a versor with magnitude equal to 1 and phase equal to  $\frac{2\pi}{3}$ . If a complex number is multiplied by  $\bar{a}$  its module is unchanged while its phase is increased by  $\frac{2\pi}{3}$ . In other words  $\bar{a}$  represents a rotation of  $\frac{2\pi}{3}$ .

The general expression of a space vector representing three-phase quantities (where  $a, b$  and  $c$  are the three phases) is:

$$\mathbf{x} = \frac{2}{3} \cdot (x_a + \bar{a} \cdot x_b + \bar{a}^2 \cdot x_c) \quad (2.2)$$

In three-phase motor control applications is commonly used the Clarke transformation to convert the three-phase electrical quantities (such as voltage or current) into two-phase ( $\alpha\beta$ ) quantities that can be more easily processed. The ( $\alpha, \beta$ ) axes are the real-axis and the imaginary-axis of a complex plane. A three-phase set  $(x_a, x_b, x_c)$  will correspond to a vector  $\mathbf{x}$  whose projections on the axes are  $(x_\alpha, x_\beta)$ , respectively. The Clarke transformation can be written as a 3x2 matrix  $[T]$ :

$$\begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} = [T] \cdot \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} \quad (2.3)$$

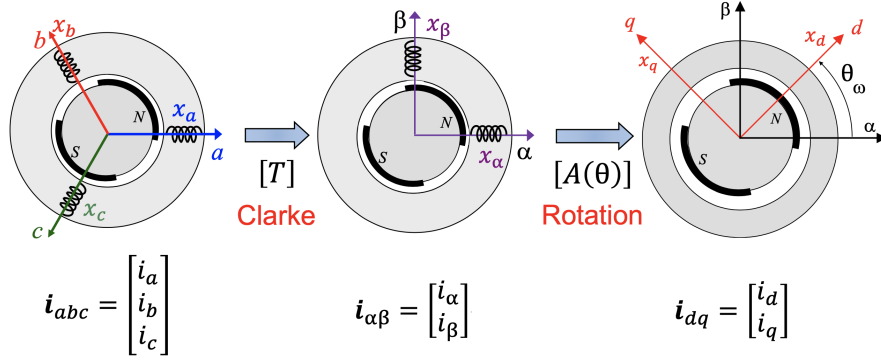
$$[T] = \frac{2}{3} \cdot \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \quad (2.4)$$

The Clarke transformation is also a key step in the Park transformation, which is used to convert the two-phase signals into a rotating reference frame that simplifies the control of AC motors. Such reference frame is called  $dq$  frame. Considering

1.  $\alpha\beta \rightarrow$  fixed coordinates;
2.  $dq \rightarrow$  rotating coordinates;
3.  $\theta \rightarrow$  position of  $d$  axis with respect to the  $\alpha$  axis.

the rotation transformation can be written in matrix form ( $[A(\theta)]$  = direct rotational matrix):

$$\begin{bmatrix} x_d \\ x_q \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \cdot \begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} = [A(\theta)] \cdot \begin{bmatrix} x_d \\ x_q \end{bmatrix} \quad (2.5)$$



**Figure 2.1:** Coordinates transformation [2].

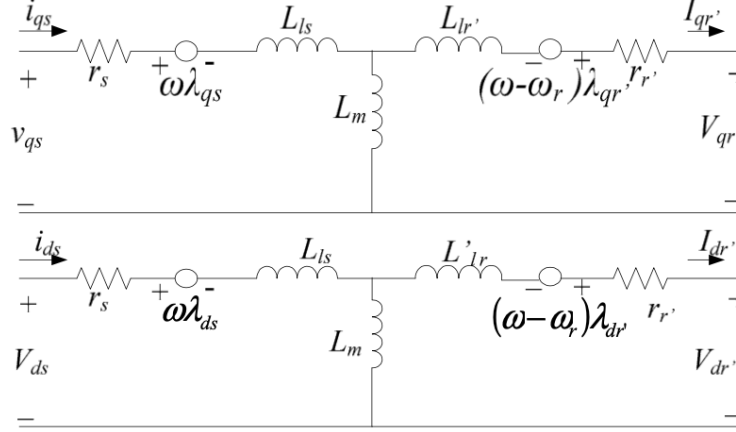
### 2.1.2 Voltage equations

The voltage equation of an induction motor can be derived from its equivalent circuit by applying Kirchhoff's laws to the circuit. An equivalent circuit of an induction motor is a simplified representation of the motor that allows for easy analysis and calculations of the motor's electrical and mechanical characteristics. The equivalent circuit consists of electrical and mechanical components that represent the various losses and parameters of the motor.

The equivalent circuit of a three-phase induction motor includes the following components:

1. Stator resistance and leakage reactance: these components represent the electrical losses in the stator winding and the inductive voltage drop in the stator windings due to the leakage flux;
2. Rotor resistance and leakage reactance: these components represent the electrical losses in the rotor winding and the inductive voltage drop in the rotor windings due to the leakage flux;
3. Magnetizing reactance: this component represents the magnetic field produced by the stator winding;
4. Core loss resistance: this component represents the losses due to hysteresis and eddy currents in the motor's magnetic core;
5. Mechanical load: this component represents the external mechanical load on the motor.

The equivalent circuit is usually drawn in the form of a circuit diagram with the stator and rotor windings shown as inductors, and the magnetizing reactance and core loss resistance shown as separate components.



**Figure 2.2:** Equivalent Circuit of induction motor in  $(d,q)$  reference frame [3].

The resulting equations describe the voltages induced in the stator and rotor windings as a result of the time-varying magnetic flux in the motor. The stator voltage equation referred to the stationary  $(\alpha,\beta)$  frame is given by:

$$\begin{bmatrix} v_{s,\alpha} \\ v_{s,\beta} \end{bmatrix} = R_s \cdot \begin{bmatrix} i_{s,\alpha} \\ i_{s,\beta} \end{bmatrix} + \frac{d}{dt} \cdot \begin{bmatrix} \lambda_{s,\alpha} \\ \lambda_{s,\beta} \end{bmatrix} \quad (2.6)$$

$$\mathbf{v}_{s,\alpha\beta} = R_s \cdot \mathbf{i}_{s,\alpha\beta} + \frac{d}{dt} \cdot \lambda_{s,\alpha\beta} \quad (2.7)$$

The rotor voltage equation referred to the stator  $(\alpha,\beta)$  frame is instead given by:

$$\begin{bmatrix} v_{r,\alpha} \\ v_{r,\beta} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} = R_r \cdot \begin{bmatrix} i_{r,\alpha} \\ i_{r,\beta} \end{bmatrix} + \frac{d}{dt} \cdot \begin{bmatrix} \lambda_{r,\alpha} \\ \lambda_{r,\beta} \end{bmatrix} - j\omega_r \cdot \begin{bmatrix} \lambda_{r,\alpha} \\ \lambda_{r,\beta} \end{bmatrix} \quad (2.8)$$

$$\mathbf{v}_{r,\alpha\beta} = \mathbf{0}_{r,\alpha\beta} = R_r \cdot \mathbf{i}_{r,\alpha\beta} + \frac{d}{dt} \cdot \lambda_{r,\alpha\beta} - j\omega_r \cdot \lambda_{r,\alpha\beta} \quad (2.9)$$

Passing from the stationary  $(\alpha,\beta)$  to the rotating  $(d,q)$  reference system, the voltage equations can be written in the form:

$$\mathbf{v}_{s,dq} = R_s \cdot \mathbf{i}_{s,dq} + \frac{d}{dt} \cdot \lambda_{s,dq} + j\omega \cdot \lambda_{s,dq} \quad (2.10)$$

$$\mathbf{v}_{r,dq} = \mathbf{0}_{r,dq} = R_r \cdot \mathbf{i}_{r,dq} + \frac{d}{dt} \cdot \lambda_{r,dq} + j(\omega - \omega_r) \cdot \lambda_{r,dq} \quad (2.11)$$

1.  $\mathbf{v}_s$  is the stator voltage;
2.  $\mathbf{v}_r$  is the rotor voltage;
3.  $\mathbf{i}_s$  is the stator current;
4.  $\mathbf{i}_r$  is the rotor current;
5.  $\lambda_s$  is the stator flux linkage with the stator windings;
6.  $\lambda_r$  is the rotor flux linkage with the stator windings;
7.  $\omega_r$  is the electrical rotor speed;
8.  $\omega_{\text{slip}} = (\omega - \omega_r)$  is the electrical slip speed.

### 2.1.3 Magnetic model

In an induction motor, the magnetic flux and current are closely related. The flow of current through the stator windings produces a magnetic field which rotates, producing a rotating magnetic flux. This rotating magnetic flux induces an alternating current in the rotor windings, which in turn produces a magnetic field that interacts with the stator magnetic field to produce torque and rotation.

The magnetic model of an induction motor describes the relationship between the magnetic field, the magnetic flux, and the magnetic forces that are generated as a result of the interaction between the stator and rotor magnetic fields [4]. This model can be represented by the following equations:

$$\lambda_{s,\alpha\beta} = L_s \cdot \mathbf{i}_{s,\alpha\beta} + L_m \cdot \mathbf{i}_{r,\alpha\beta} \quad (2.12)$$

$$\lambda_{r,\alpha\beta} = L_m \cdot \mathbf{i}_{s,\alpha\beta} + L_r \cdot \mathbf{i}_{r,\alpha\beta} \quad (2.13)$$

The model expression does not depend on the reference frame:

$$\lambda_{s,dq} = L_s \cdot \mathbf{i}_{s,dq} + L_m \cdot \mathbf{i}_{r,dq} \quad (2.14)$$

$$\lambda_{r,dq} = L_m \cdot \mathbf{i}_{s,dq} + L_r \cdot \mathbf{i}_{r,dq} \quad (2.15)$$

Where:

1.  $L_s$  is the stator inductance of an induction motor which is a parameter that characterizes the electrical behavior of the stator winding. It is related to the motor's ability to produce a magnetic field and to the amount of voltage that is induced in the stator winding due to the rotation of the magnetic field. The stator inductance can be calculated from the physical dimensions of the motor, such as the number of turns in the stator winding, the length and diameter of the winding, and the permeability of the core.
2.  $L_r$  is the rotor inductance and it is the inductance of the rotor winding. It is a measure of the ability of the rotor to store energy in the form of a magnetic field. The rotor inductance is a function of the motor's physical dimensions just like the stator inductance.
3.  $L_m$  is the magnetizing inductance of an induction motor which is a parameter that affects the motor's performance. It is a measure of the motor's ability to store and release energy in the form of magnetic flux, and it is related to the amount of current required to establish a magnetic field in the motor's core.

These inductances are related to each other through the following relationship:

$$L_s = L_m + Lls \quad (2.16)$$

$$L_r = L_m + Llr \quad (2.17)$$

Where:

1.  $Lls$  is the stator leakage inductance. It represents the portion of the stator inductance that is due to the magnetic flux that does not link with the rotor, but instead leaks out of the stator winding and returns to the stator through the air gap.
2.  $Llr$  is the rotor leakage inductance. It refers to the inductance associated with the flux leakage between the rotor winding and the stator winding. This leakage inductance results in a delay of the current flowing through the rotor, which in turn affects the motor's performance.



Denoted  $\lambda_{\mathbf{m}} = \mathbf{i}_{\mathbf{m}} \cdot L_m$  as magnetizing flux and  $\mathbf{i}_{\mathbf{m}} = \mathbf{i}_{\mathbf{s}} + \mathbf{i}_{\mathbf{r}}$  as magnetizing current it is possible to represent the magnetic model with the following equations:

$$\begin{aligned}
 \lambda_{\mathbf{s}} &= L_s \cdot \mathbf{i}_{\mathbf{s}} + L_m \cdot \mathbf{i}_{\mathbf{r}} \\
 &= (L_m + Lls) \cdot \mathbf{i}_{\mathbf{s}} + L_m \cdot \mathbf{i}_{\mathbf{r}} \\
 &= Lls \cdot \mathbf{i}_{\mathbf{s}} + L_m \cdot (\mathbf{i}_{\mathbf{s}} + \mathbf{i}_{\mathbf{r}}) \\
 &= Lls \cdot \mathbf{i}_{\mathbf{s}} + L_m \cdot \mathbf{i}_{\mathbf{m}} \\
 &= Lls \cdot \mathbf{i}_{\mathbf{s}} + \lambda_{\mathbf{m}}
 \end{aligned} \tag{2.18}$$

$$\begin{aligned}
 \lambda_{\mathbf{r}} &= L_r \cdot \mathbf{i}_{\mathbf{r}} + L_m \cdot \mathbf{i}_{\mathbf{s}} \\
 &= (L_m + Llr) \cdot \mathbf{i}_{\mathbf{r}} + L_m \cdot \mathbf{i}_{\mathbf{s}} \\
 &= Llr \cdot \mathbf{i}_{\mathbf{r}} + L_m \cdot (\mathbf{i}_{\mathbf{s}} + \mathbf{i}_{\mathbf{r}}) \\
 &= Llr \cdot \mathbf{i}_{\mathbf{r}} + L_m \cdot \mathbf{i}_{\mathbf{m}} \\
 &= Llr \cdot \mathbf{i}_{\mathbf{r}} + \lambda_{\mathbf{m}}
 \end{aligned} \tag{2.19}$$

For the model developed on Simulink these last two equations were used.

However, as the current in the stator winding increases, the magnetic saturation of the motor's core increases, which limits the amount of flux that can be produced. At a certain point, increasing the current further will not result in a proportional increase in torque. This is why induction motors are typically designed to operate at a specific voltage and frequency, which limits the amount of current that can flow through the stator winding to avoid saturating the motor's core.

#### 2.1.4 Speed and Torque estimation

The electromagnetic torque in an induction motor is generated due to the interaction between the magnetic field produced by the stator and the induced currents. This torque is produced due to the principle of electromagnetic induction, which states that when a conductor is moved in a magnetic field, a voltage is induced across the conductor, and if the conductor is part of a closed circuit, a current will flow through it:

$$T_e = \frac{3}{2} \cdot p \cdot (\lambda_{\alpha} \cdot i_{\beta} - \lambda_{\beta} \cdot i_{\alpha}) \tag{2.20}$$

$$T_e = \frac{3}{2} \cdot p \cdot (\lambda_d \cdot i_q - \lambda_q \cdot i_d) \tag{2.21}$$

$$T_e = \frac{3}{2} \cdot p \cdot (\lambda_s \wedge \mathbf{i}_s) \quad (2.22)$$

The speed of an induction motor can be computed from the electromagnetic torque and inertia moment using the following formula:

$$\omega = \frac{1}{J} \cdot \int (T_e - T_L) \quad (2.23)$$

Where  $p$  is the pole-pairs number,  $J$  is the moment of inertia and  $T_L$  is the load torque. In this case  $p = 2$  and  $J = 0.05$ . The moment of inertia of the motor can be determined by measuring the mass distribution and geometry of the rotating components, such as the rotor and shaft. The formula is based on the conservation of energy principle, which states that the energy supplied to the motor in the form of electromagnetic torque is converted into kinetic energy, which is proportional to the moment of inertia and the square of the motor speed.

## 2.2 Losses of an Induction Motor

Some of the main losses in an Induction Motor are:

1. Stator and Rotor joule Losses: these losses occur in the stator and rotor winding due to the flow of current in the winding.
2. Iron losses: these losses occur in the iron core of the stator and rotor due to the alternating magnetic field. The losses are caused by hysteresis and eddy current effects and are dependent on the frequency and the magnetic properties of the core material.
3. Mechanical Losses: these losses occur due to friction and windage in the motor. Friction losses occur in the bearings and other moving parts, while windage losses occur due to the movement of air around the rotor and stator.

All of these losses contribute to the overall efficiency of the motor. In the model developed for the purpose of this thesis only the stator and rotor Joule losses and the iron losses have been considered and implemented.

### 2.2.1 Iron losses

Iron losses in an induction motor refer to the energy losses that occur in the iron core of the motor due to the flow of magnetic flux. Iron losses in an induction motor are generally caused by two phenomena: hysteresis loss and eddy current loss.

Hysteresis losses occur as a result of the magnetic reversal of the iron core in the motor. This results in the energy loss due to the reorientation of the magnetic domains in the iron core.

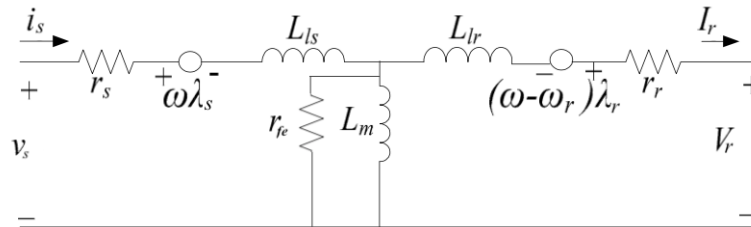
Eddy current losses occur due to the presence of circulating currents in the iron core. These circulating currents are induced as a result of the changing magnetic field in the motor.

It is possible to provides an estimation of iron losses using the iron resistance and iron current of the motor. The iron resistance is the resistance of the iron core to the flow of magnetic flux, and the iron current is the current flowing through the iron core of the motor. The iron resistance of the motor can be determined experimentally by measuring the resistance of the stator winding when the rotor is locked, or by measuring the voltage drop across the stator winding and the current flowing through it. In the model developed on MATLAB/Simulink, the value of the iron resistance was set to  $1K\Omega$ . Once the iron resistance is known, the iron losses can be estimated using the following formula:

$$P_{fe} = R_{fe} \cdot \mathbf{i}_{fe}^2 = \frac{3}{2} \cdot R_{fe} \cdot (\mathbf{i}_{fe,\alpha}^2 + \mathbf{i}_{fe,\beta}^2) \quad (2.24)$$

The iron current is the current flowing through the iron core of the motor, and it can be measured by subtracting the stator current from the total current flowing through the motor.

It is possible to represent the contribution of the iron losses by adding the iron resistance [5] in the equivalent circuit in Fig. 2.2 as follow:



**Figure 2.3:** Equivalent Circuit of induction motor with iron resistance.

### 2.2.2 Joule losses

Joule losses are one of the main sources of power losses in an induction motor. These losses occur due to the resistance of the copper wire used in the construction of the stator and rotor. When an electric current flows through the stator and rotor windings of an induction motor, it encounters resistance in the wire. This resistance results in the generation of heat, which is dissipated into the surrounding environment. The more energy dissipated, the higher the temperature of the resistance [6].

To calculate Joule losses in induction motors, it is necessary to know the current flowing through the stator and rotor windings and the resistance of the copper wire used. The amount of joule losses in the motor depends on the square of the current and the resistance of the windings, which can vary with temperature:

$$\begin{aligned} P_{j,s} &= R_s \cdot \mathbf{i}_s^2 \\ &= R_s \cdot (\mathbf{i}_{s,a}^2 + \mathbf{i}_{s,b}^2 + \mathbf{i}_{s,c}^2) \\ &= \frac{3}{2} \cdot R_s \cdot (\mathbf{i}_{s,\alpha}^2 + \mathbf{i}_{s,\beta}^2) \end{aligned} \quad (2.25)$$

$$\begin{aligned} P_{j,r} &= R_r \cdot \mathbf{i}_r^2 \\ &= R_r \cdot (\mathbf{i}_{r,a}^2 + \mathbf{i}_{r,b}^2 + \mathbf{i}_{r,c}^2) \\ &= \frac{3}{2} \cdot R_r \cdot (\mathbf{i}_{r,\alpha}^2 + \mathbf{i}_{r,\beta}^2) \end{aligned} \quad (2.26)$$

The variation of the stator and rotor resistance due to the temperature can be expressed by the following formulas:

$$R_s(\theta_s) = R_s(\theta_0) \cdot \frac{k_s + \theta_s}{k_s + \theta_0} \quad (2.27)$$

$$R_r(\theta_r) = R_r(\theta_0) \cdot \frac{k_r + \theta_r}{k_r + \theta_0} \quad (2.28)$$

Where:

1.  $k_s$  is the temperature coefficient of resistance for copper (used in the stator) and its value is approximately 234.5;
2.  $k_r$  is the temperature coefficient of resistance for aluminum (used in the rotor) and its value is approximately 225;

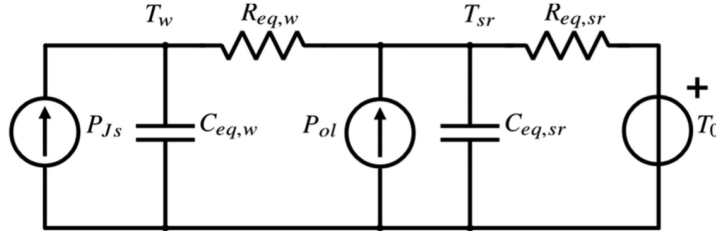
3.  $\theta_s$  and  $\theta_r$  are the stator and rotor actual resistance temperatures, respectively;
4.  $\theta_0$  is the ambient temperature (25°C);
5.  $R_s(\theta_s)$  and  $R_s(\theta_0)$  are the stator resistances at actual and ambient temperatures, respectively;
6.  $R_r(\theta_r)$  and  $R_r(\theta_0)$  are the rotor resistances at actual and ambient temperatures, respectively.

## 2.3 Thermal model

An induction motor is a type of AC motor that uses electromagnetic induction to convert electrical energy into mechanical energy. The operation of an induction motor involves complex thermal processes, as the motor generates heat due to various factors such as electrical losses, mechanical losses, and magnetic losses.

The thermal model of an induction motor is a mathematical representation of the heat transfer processes occurring inside the motor and can be used to optimize the design of the motor, predict the temperature rise during operation, and estimate the lifetime of the motor.

### 2.3.1 Second order model



**Figure 2.4:** Second order thermal model circuit of the motor [7].

A second-order thermal model has been chosen. It is a mathematical model that describes the dynamics of a thermal system using second-order differential equations. The proposed thermal model is shown in Fig. 2.4, where:

1.  $R_{eq,w}$  is the equivalent thermal resistance between the winding copper and the stator lamination.

2.  $C_{eq,w}$  is the equivalent thermal capacitance of the winding by including copper and insulation system.
3.  $R_{eq,sr}$  is the equivalent thermal resistance between the stator lamination and the ambient.
4.  $C_{eq,sr}$  is the thermal capacitance of stator and rotor without including the stator winding capacitance.
5.  $T_w$  is the temperature of the stator winding.
6.  $T_0$  is the ambient temperature.
7.  $T_{sr}$  is the temperature associated with the thermal capacitance  $C_{eq,sr}$ .
8.  $P_{js}$  are the Joule losses of the stator windings.
9.  $P_{ol}$  are the other machine losses (iron and rotor Joule).  $P_{ol} = P_{fe} + P_{jr}$ .

The parameters of the proposed thermal model can be determined by supplying the machine with a dc source. The thermal parameters of the winding  $R_{eq,w}$  and  $C_{eq,w}$  are computed using the samples belonging to the first minutes of the test like shown in [7], [8], and [9].

The values considered for the parameters  $R_{eq,sr}$  and  $C_{eq,sr}$  are those obtained from [7]. In order to consider the forced convection effects caused by the self-ventilation, the thermal resistance  $R_{eq,sr}$  considered in the MATLAB/Simulink model was calculated as the result of two resistances in parallel, i.e. the real  $R_{eq,sr}$  and the thermal resistance of forced convection  $R_{fc}$ .

The values obtained for the parameters of interest are the following:

1.  $R_{eq,w} = 0.0700 \text{ K/W}$
2.  $C_{eq,w} = 1708.2 \text{ J/K}$
3.  $R_{eq,sr} = 0.0709 \text{ K/W}$
4.  $C_{eq,sr} = 10369.0 \text{ J/K}$

$$\begin{bmatrix} T_w \\ T_{sr} \end{bmatrix} = [A] \cdot \begin{bmatrix} T_w \\ T_{sr} \end{bmatrix} + [B] \cdot \begin{bmatrix} P_{js} \\ P_{ol} \\ T_0 \end{bmatrix} \quad (2.29)$$

Eq.(2.29) is the representation of the space-state model associated with the thermal circuit of Fig.2.4 when the motor is supplied with a dc source. Where:

$$[A] = \begin{bmatrix} \frac{-1}{R_{eq,w}C_{eq,w}} & \frac{1}{R_{eq,w}C_{eq,w}} \\ \frac{1}{R_{eq,w}C_{eq,sr}} & -\frac{1}{C_{eq,sr}}\left(\frac{1}{R_{eq,w}} + \frac{1}{R_{eq,sr}}\right) \end{bmatrix} \quad (2.30)$$

$$[B] = \begin{bmatrix} \frac{1}{C_{eq,w}} & 0 & 0 \\ 0 & \frac{1}{C_{eq,sr}} & \frac{1}{C_{eq,sr}R_{eq,sr}} \end{bmatrix} \quad (2.31)$$

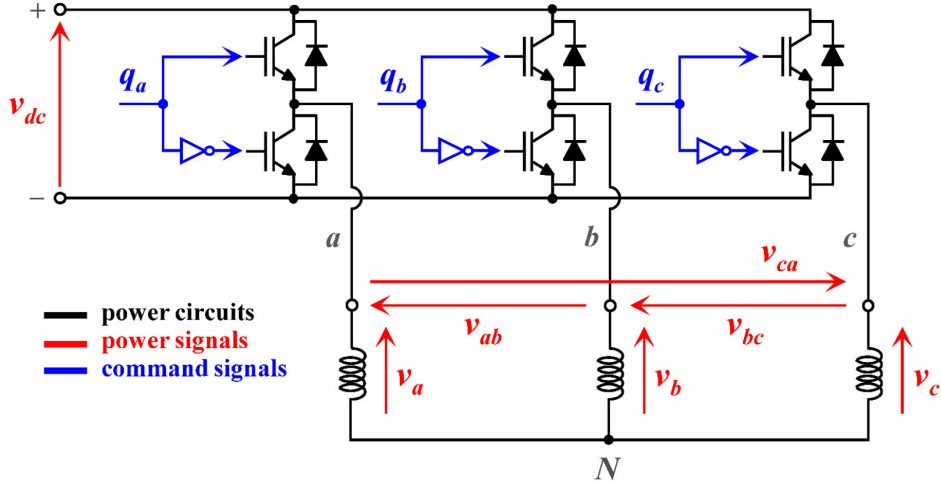
## 2.4 Inverter

An inverter is an electronic device that converts DC power into AC power with variable frequency and voltage output. In the context of electric motors, an inverter is commonly used to control the speed and torque of AC induction motors.

A three-phase inverter is a type of inverter that is specifically designed to work with three-phase AC motors like induction motors. Three-phase motors are commonly used in industrial applications because of their higher efficiency, better power factor, and lower torque ripple compared to single-phase motors. When a three-phase inverter is used to control the speed of an induction motor, it works by adjusting the frequency and voltage of the AC power supplied to the motor. The inverter produces a three-phase AC output waveform with variable frequency and voltage that is applied to the three-phase induction motor.

A three-phase inverter consists of six power switches (usually MOSFETs or IGBTs) arranged in three pairs, with each pair connected to one of the three phases of the AC motor. By switching these power switches on and off in a controlled sequence, the inverter can produce a three-phase AC output with variable frequency and voltage, allowing for precise control of the motor speed and torque.

The inverter also helps to improve the efficiency of the induction motor by controlling the flow of current through the motor windings. By controlling the frequency and voltage of the AC power supplied to the motor, the inverter can reduce the amount of energy wasted as heat, which can lead to significant energy savings and lower operating costs.



**Figure 2.5:** Circuit diagram of a voltage-source three-phase inverter [10].

### 2.4.1 Three-phase Inverter Model

The input of a three-phase inverter is typically a DC voltage or current source that provides the power necessary to drive the inverter circuit. The voltage or current level of the DC source depends on the rating of the inverter and the load it is driving, and it must be regulated to ensure the proper operation of the inverter.

The DC source is typically connected to a set of power electronics switches, such as MOSFETs or IGBTs [11]. The inverter circuit uses pulse-width modulation (PWM) to control the switching of the power electronics switches, which in turn generates the desired three-phase AC output waveform.

The command signal of a three-phase inverter is the reference waveform that is used to generate the desired three-phase AC output waveform, while the duty cycle is the percentage of time that the power switches (MOSFETs or IGBTs) in the inverter are turned on during each switching cycle. The duty cycle is adjusted based on the command signal and other feedback signals, such as the output voltage and current, to regulate the output waveform and ensure proper operation of the inverter and the connected load.

In a three-phase inverter, the pulse-width modulation (PWM) voltage laws are used to determine the switching patterns of the power electronics switches in order to generate the desired three-phase AC output waveform:



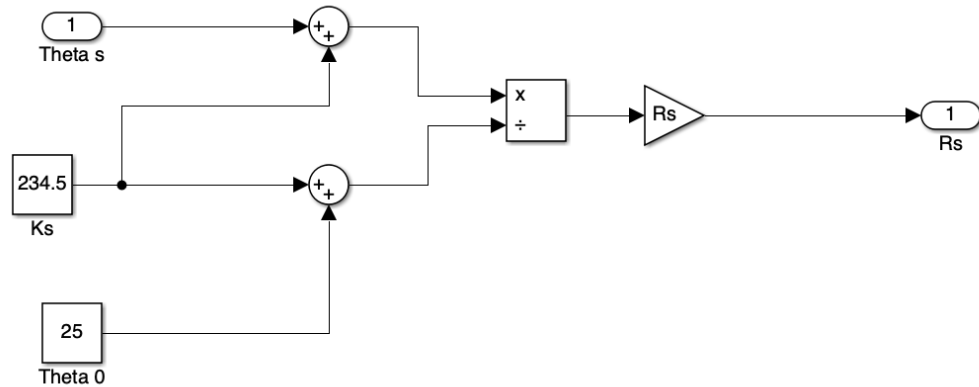
$$\begin{aligned}
 V_{aN}(\tau) &= \frac{V_{dc}}{3} \cdot [2 \cdot d_a(\tau) - d_b(\tau) - d_c(\tau)] \\
 V_{bN}(\tau) &= \frac{V_{dc}}{3} \cdot [2 \cdot d_b(\tau) - d_c(\tau) - d_a(\tau)] \\
 V_{cN}(\tau) &= \frac{V_{dc}}{3} \cdot [2 \cdot d_c(\tau) - d_a(\tau) - d_b(\tau)]
 \end{aligned} \tag{2.32}$$

Where:

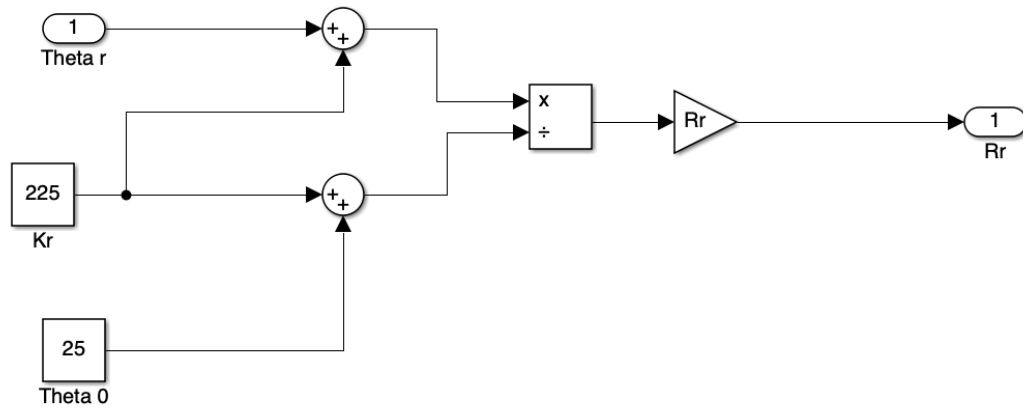
1.  $V_{xN}$  ( $x = a, b, c$ ) are the load voltages which represent the quantities that one is interested in controlling;
2.  $d_x$  ( $x = a, b, c$ ) are the duty cycles;
3.  $q_x$  ( $x = a, b, c$ ) are the instantaneous command functions;
4.  $V_{dc}$  is the dc-link voltage.

## 2.5 Simulink representation

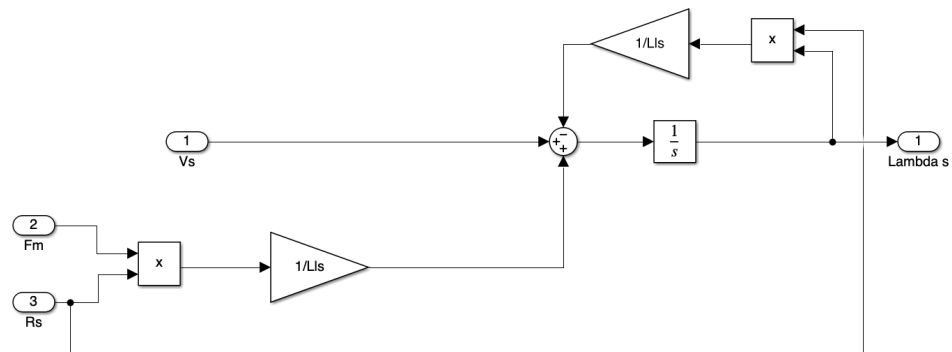
Below is shown graphically how the induction motor model, in all the aspects described in the previous sections, has been implemented on MATLAB/Simulink:



**Figure 2.6:** Stator resistance as a function of temperature.



**Figure 2.7:** Rotor resistance as a function of temperature.



**Figure 2.8:** Stator flux computation.

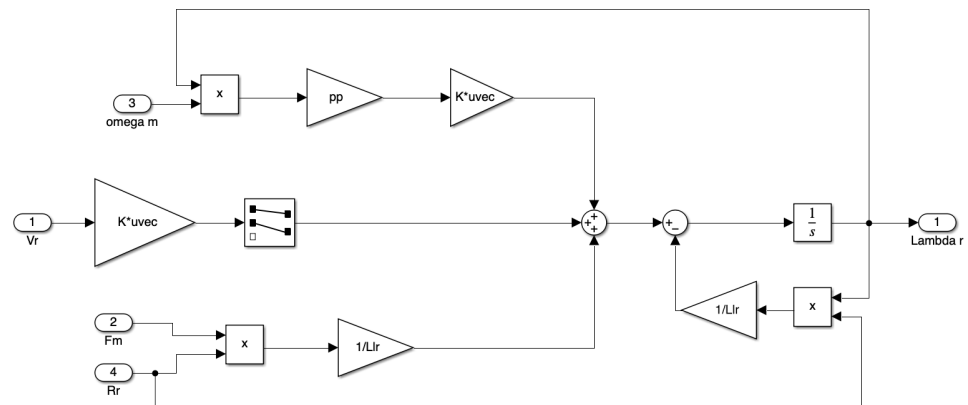


Figure 2.9: Rotor flux computation.

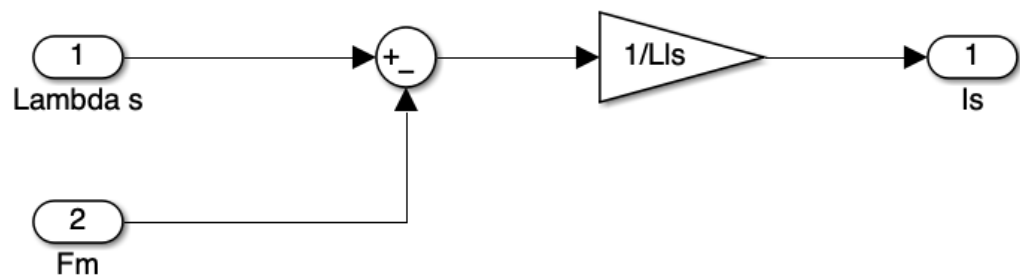


Figure 2.10: Stator current computation.

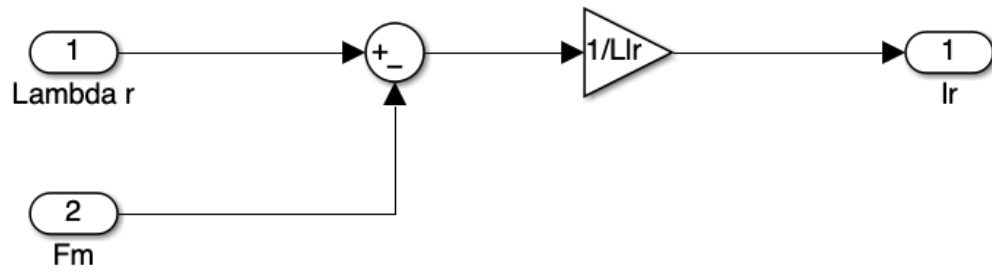


Figure 2.11: Rotor current computation.

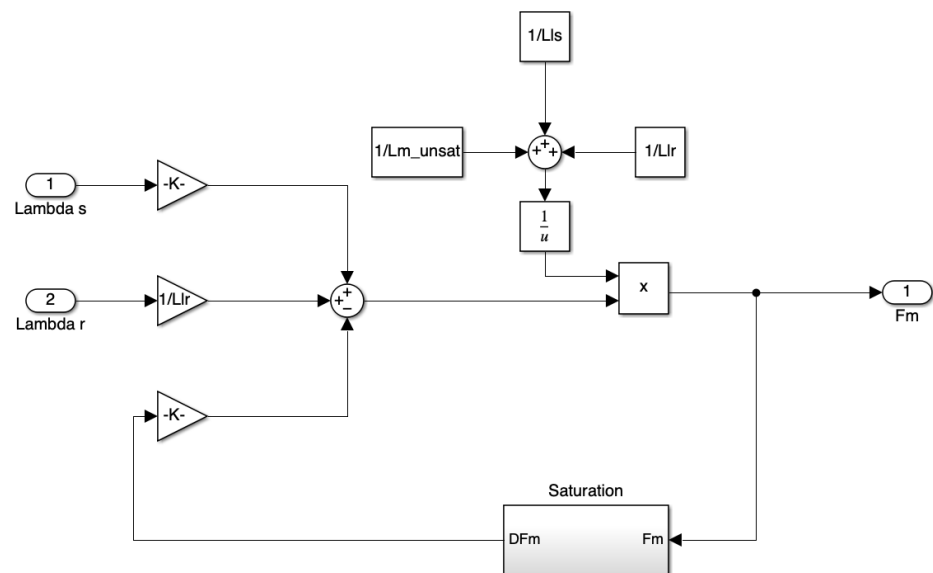
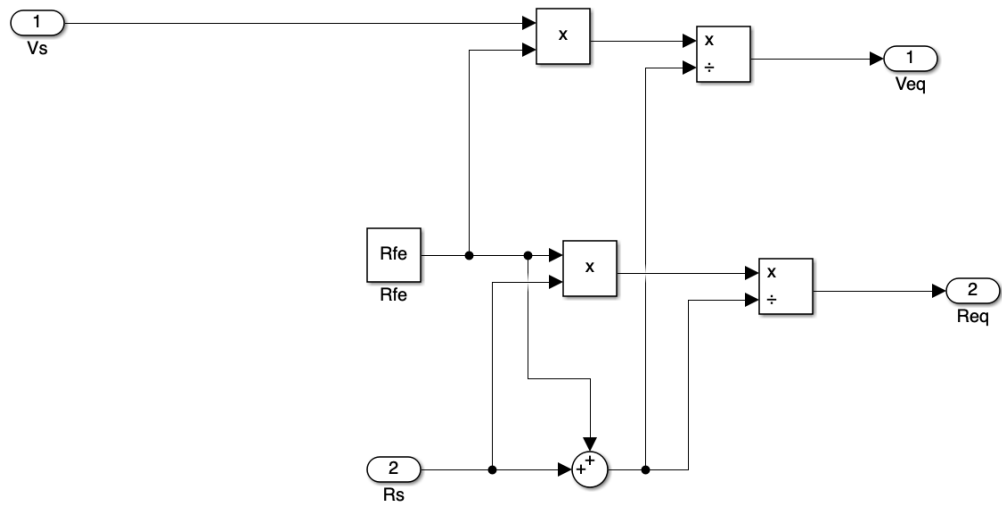
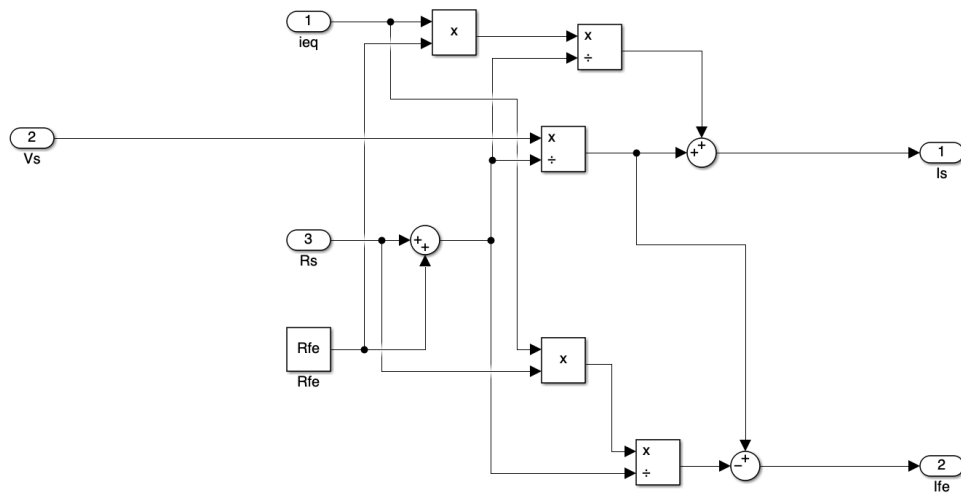


Figure 2.12: Magnetizing flux computation.



**Figure 2.13:** Adding of the iron resistance.



**Figure 2.14:** Iron current computation

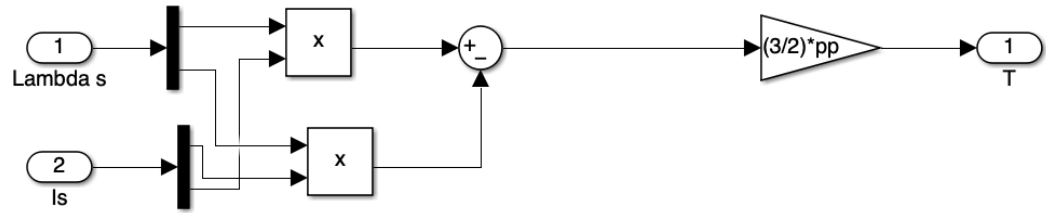


Figure 2.15: Electromagnetic torque computation.

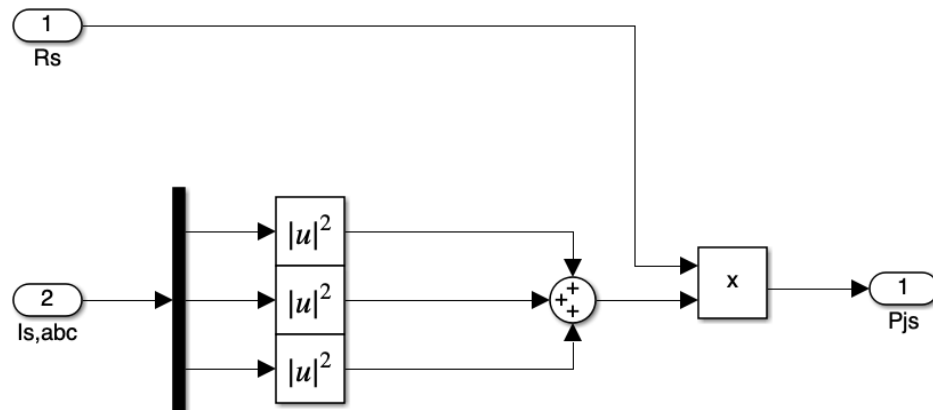


Figure 2.16: Stator Joule losses computation.

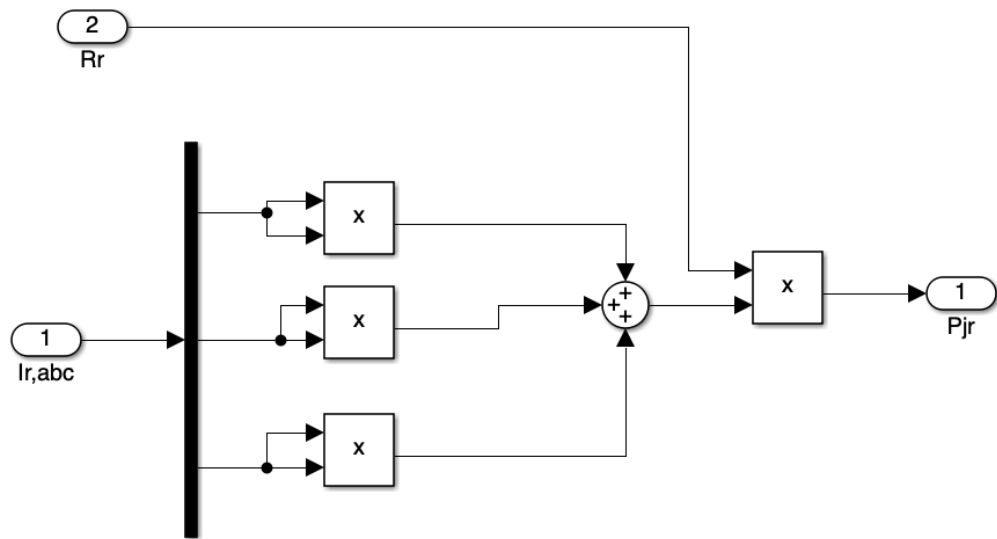


Figure 2.17: Rotor Joule losses computation.

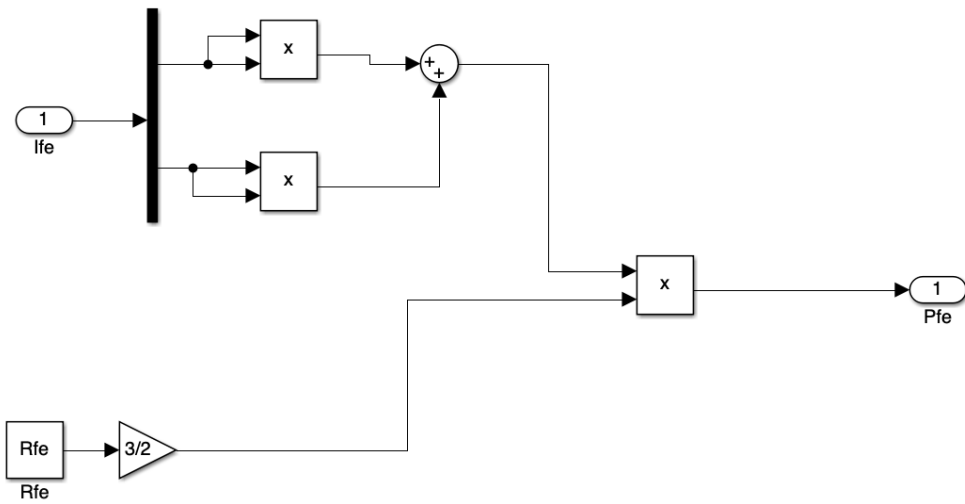
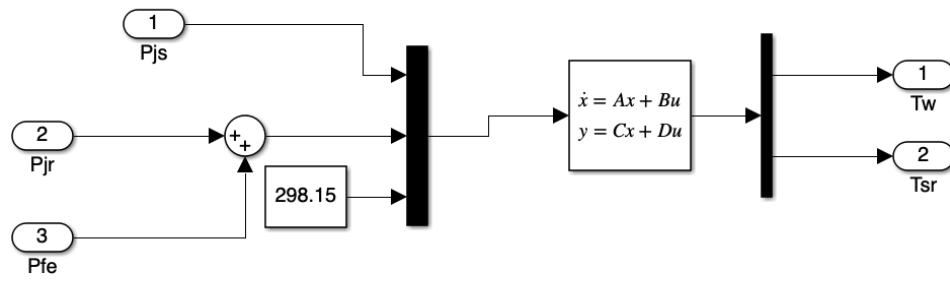


Figure 2.18: Iron losses computation.



**Figure 2.19:** Second order thermal model system.



## Chapter 3

# Motor control

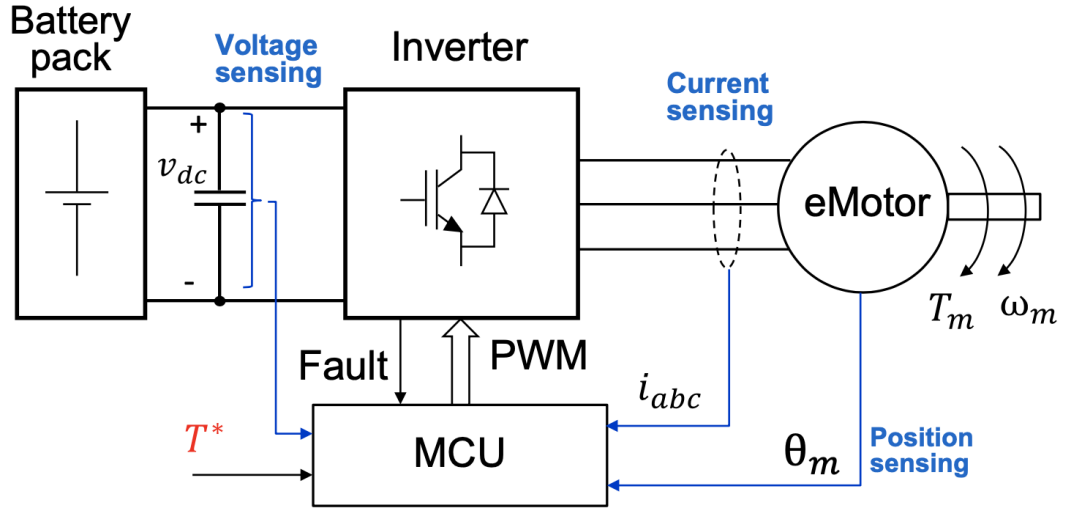
The control of an induction motor is important to achieve the desired performance and efficiency for various industrial applications. There are several methods used for the control of induction motors, including voltage control, frequency control, vector control and direct torque control.

1. **Voltage Control:** voltage control is a simple and effective method of controlling the speed of an induction motor. The speed of the motor is proportional to the voltage applied to its stator windings. Thus, by varying the voltage applied to the motor, the speed of the motor can be controlled.
2. **Frequency Control:** frequency control is another method used to control the speed of an induction motor. The speed of the motor is directly proportional to the frequency of the voltage applied to its stator windings. Thus, by varying the frequency of the voltage applied to the motor, the speed of the motor can be controlled.
3. **Vector Control:** vector control, also known as field-oriented control, is a more advanced method used to control the speed and torque of an induction motor. It is based on the concept of decoupling the motor's stator current into two orthogonal components: one component produces the magnetic field and the other component produces the torque. The vector control method uses a complex control algorithm to adjust the magnitude and phase of these two current components, which results in precise control of the motor speed and torque.
4. **Direct torque control (DTC):** is a control strategy used for induction motors that directly regulates the motor's torque and flux. It involves the use of hysteresis comparators to compare the actual and desired values of torque and flux and select the appropriate voltage vector to be applied to the motor.

DTC provides fast dynamic response and high accuracy, with the ability to operate without position sensors.

For the purpose of this thesis, the method used for the control of the motor is the Vector control and it has been implemented on MATLAB in C code. The motor control was initially implemented and tested on the previously created Simulink model through the "S-function" block. In the following, the vector control method will be introduced and explain in detail how it has been implemented.

### 3.1 Vector control



**Figure 3.1:** Block diagram of a motor control unit (MCU) functioning [12].

Vector control, also known as field-oriented control (FOC), is a popular, powerful and efficient technique for controlling the speed and torque of induction motors, and is widely used in many industrial applications such as electric vehicles, wind turbines, and industrial automation. It involves controlling the stator currents of the motor in a way that produces a rotating magnetic field that is oriented with respect to the rotor flux.

The basic principle behind vector control for induction motors is to decouple the torque and flux producing components of the stator current using mathematical transformations, such as Park's and Clarke's transformations, to transform the

three-phase stator currents into a two-phase stationary reference frame. In this frame, the torque and flux producing components of the stator current can be easily controlled independently [13].

The torque producing component of the stator current is controlled by using a proportional-integral (PI) controller that regulates the torque reference. The flux is instead obtained through voltage and magnetic models. By controlling these two components independently, the motor can be operated at any desired speed and torque. In vector control for induction motors, the motor is controlled with an external reference torque  $T^*$  by using feedback signals from sensors that measure the stator current and rotor speed [14]. These signals are used to calculate the required voltage vector that needs to be applied to a PWM modulator to obtain the switching functions of the converter power device in order to allow the motor to achieve the desired torque and speed.

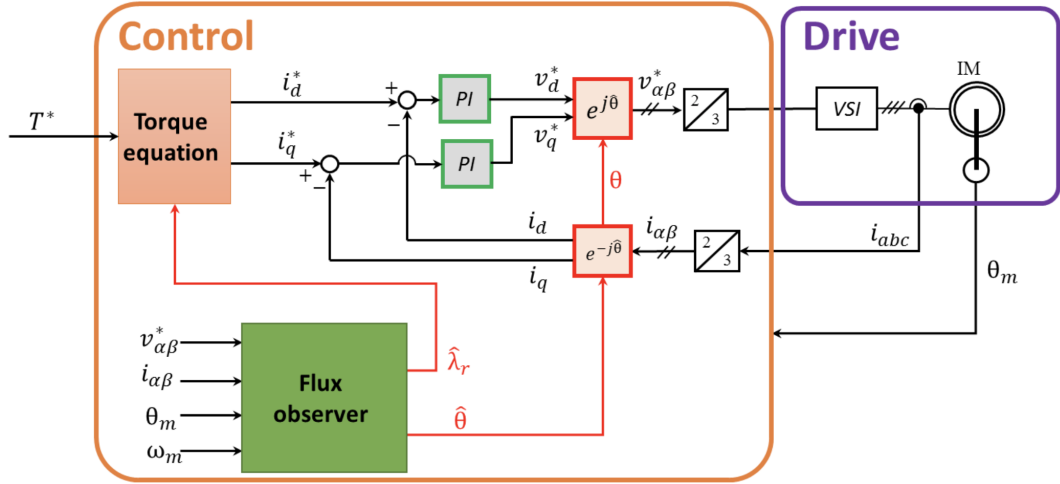


Figure 3.2: Motor control block diagram [12].

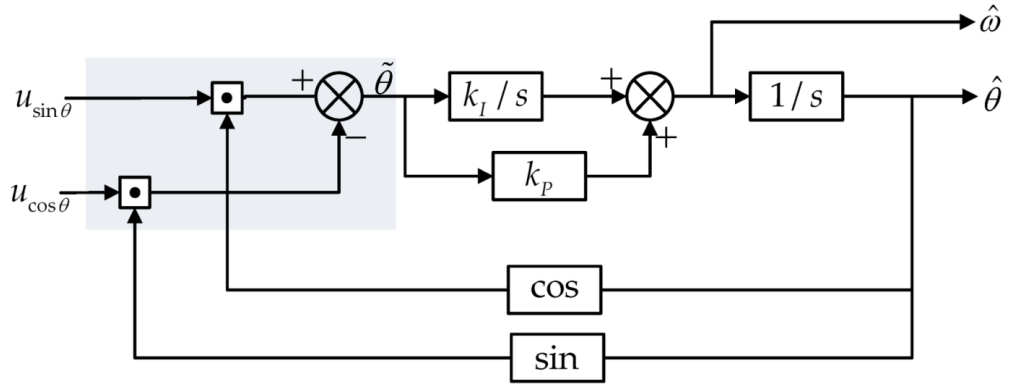
### 3.2 Speed estimator through PLL observer

As already mentioned previously, in field-oriented control, in order to control the motor it is necessary to measure the stator current and the rotor speed through some sensors.

A PLL (Phase-Locked Loop) observer is a technique used in induction motor

control to estimate the speed and position of the rotor without using physical sensors. The advantages of using a PLL observer in induction motor control include lower cost, reduced complexity, and increased reliability since it eliminates the need for physical sensors to measure rotor speed and position. It is widely used in applications where sensorless control is required, such as electric vehicles, robotics, and industrial automation [15].

It involves using a PLL circuit to lock onto the fundamental frequency of the stator current and then using this frequency to estimate the speed and position of the rotor. The PLL circuit in the observer generates a signal that is in phase with the stator current, and the phase difference between this signal and the rotor position is used to calculate the speed and position of the rotor. The observer continuously updates the speed and position estimates based on the feedback from the PLL circuit, and these estimates can then be used in the control algorithm for the induction motor [16][17].



**Figure 3.3:** PLL structure [18].

The main steps involved in implementing a PLL observer are:

1. Generate a reference signal: A reference signal is generated at the desired frequency and phase angle. In this case the frequency is set to 20  $Hz$  while the phase is set to  $60^\circ$ .
2. Calculate the PLL error: The observer calculates the phase error between the reference signal and the stator current phase angle. In a PLL control system, the accuracy of the error estimation is critical to achieving accurate rotor speed computation. Any error in the estimation will cause the PLL

to generate an incorrect reference signal, which can lead to reduced motor performance, instability, or even failure.

3. Filter the PLL error: the error signal is passed through a low-pass filter, which removes the high-frequency components and provides a smoothed estimate of the phase difference. The output of the low-pass filter is used to adjust the frequency and phase of the reference signal to track the phase angle of the stator current or voltage signal. To ensure accurate error estimation, it is essential to carefully select the low-pass filter parameters and tune the PLL control proportional and integral gain ( $k_p$  and  $k_i$ ).
4. Adjust the frequency and phase of the reference signal: The output of the low-pass filter is used to adjust the frequency and phase of the reference signal to track the phase angle of the stator current or voltage signal.
5. Estimate the rotor speed: The output of the PLL provides an estimate of the rotor speed, which can be used to adjust the control system's parameters or drive the motor.

### 3.2.1 Flux computation

Flux computation is an important aspect of motor control for an induction motor. In an induction motor, the stator winding produces a rotating magnetic field that induces a current in the rotor, which produces its own magnetic field. The interaction of these two magnetic fields produces torque that drives the motor.

The magnetic field produced by the stator winding is proportional to the current flowing through it. Therefore, controlling the stator current is a key factor in controlling the motor's torque and speed [19]. To control the stator current, the motor control system needs to estimate the motor's flux, which is the total magnetic field produced by both the stator and rotor. This estimation is typically done using a mathematical model of the motor that takes into account the physical parameters of the motor, such as the stator and rotor resistances, inductances, and the mechanical load on the motor.

In field-oriented control (FOC) systems, the flux estimation is typically performed using a mathematical model of the motor. The model takes into account the physical parameters of the motor, such as the stator and rotor resistances, inductances, and the mechanical load on the motor. The model is used to calculate the estimated flux and the stator current needed to produce the desired torque and speed.

A flux observer is a model-based technique used for estimating the magnetic flux in the motor control of induction motors. The observer estimates the flux based on the measured motor currents and voltages and the mathematical model of the induction motor. The flux observer is particularly useful in sensorless control of induction motors, where direct measurement of the flux is not possible. Instead, the observer estimates the flux by solving a set of differential equations that describe the behavior of the motor. The observer uses the measured stator currents and voltages to update the state of the motor model and estimate the flux. The observer typically consists of two parts: a voltage model and a flux model. The voltage model estimates the stator voltage based on the measured stator currents and the rotor speed. The flux model estimates the flux based on the estimated stator voltage and the measured stator currents.

The advantages of the flux observer are that it can provide accurate flux estimation even at low speeds, it is insensitive to parameter variations, and it can be used in sensorless control of induction motors. However, the observer requires a precise motor model, which can be computationally intensive. Additionally, the observer may suffer from noise and measurement errors, which can affect the accuracy of the estimated flux.

### 3.2.2 Flux computation for low and high frequency

To obtain robust and accurate flux estimation for an induction motor across the entire frequency range, a combination of different techniques can be used. At low frequencies, the flux equation is relatively simple and can be estimated based on machine voltage, current and eventually rotor position (or speed). From rotor voltage equation (2.11) and rotor flux equation (2.15):

$$\mathbf{v}_r = \mathbf{0} = R_r \cdot \mathbf{i}_r + \frac{d\lambda_r}{dt} \quad (3.1)$$

$$\mathbf{i}_r = \frac{\lambda_r - L_m \cdot \mathbf{i}_s}{L_r} \quad (3.2)$$

From these two equations it is possible to obtain:

$$\mathbf{0} = \frac{R_r}{L_r} \cdot \lambda_r - \frac{L_m}{L_r} \cdot R_r \cdot \mathbf{i}_s + \frac{d\lambda_r}{dt} \quad (3.3)$$

Defining  $\tau_r = \frac{L_r}{R_r}$  as the rotor time constant:

$$\begin{aligned}
 0 &= \frac{R_r}{L_r} \cdot \lambda_r - \frac{L_m}{L_r} \cdot R_r \cdot \mathbf{i}_s + \frac{d\lambda_r}{dt} \\
 &= \frac{1}{\tau_r} \cdot \lambda_r - \frac{L_m}{L_r} \cdot R_r \cdot \mathbf{i}_s + \frac{d\lambda_r}{dt}
 \end{aligned} \tag{3.4}$$

$$\begin{aligned}
 \tau_r \cdot \frac{L_m}{L_r} \cdot R_r \cdot \mathbf{i}_s &= \lambda_r + \tau_r \cdot \frac{d\lambda_r}{dt} \\
 \frac{L_r}{R_r} \cdot \frac{L_m}{L_r} \cdot R_r \cdot \mathbf{i}_s &= \lambda_r + \tau_r \cdot \frac{d\lambda_r}{dt} \\
 L_m \cdot \mathbf{i}_s &= \lambda_r + \tau_r \cdot \frac{d\lambda_r}{dt}
 \end{aligned} \tag{3.5}$$

The equation of the rotor flux as a function of the stator current is thus obtained:

$$\lambda_r = \frac{L_m}{1 + s\tau_r} \cdot \mathbf{i}_s \tag{3.6}$$

Starting from Eq.(2.14) and Eq.(3.2) and given the rotor flux, it is possible to derive the stator flux through the following steps:

$$\begin{aligned}
 \lambda_s &= L_s \cdot \mathbf{i}_s + L_m \cdot \mathbf{i}_r \\
 &= L_s \cdot \mathbf{i}_s + \frac{L_m}{L_r} \cdot (\lambda_r - L_m \cdot \mathbf{i}_s) \\
 &= \left(L_s - \frac{L_m^2}{L_r}\right) \cdot \mathbf{i}_s + \frac{L_m}{L_r} \cdot \lambda_r \\
 &= \sigma L_s \cdot \mathbf{i}_s + k_r \cdot \lambda_r
 \end{aligned} \tag{3.7}$$

Where  $\sigma = (1 - \frac{L_m^2}{L_r L_s})$  is the total leakage factor and  $k_r = \frac{L_m}{L_r}$  is the coupling factor [20].

At high frequencies, the flux equation for an induction motor becomes more complex due to the presence of eddy currents and other electromagnetic effects. As a result, it may be necessary to use a more advanced technique to estimate the flux and achieve robust and accurate motor control. From Eq.(2.10):

$$\mathbf{v}_s = R_s \cdot \mathbf{i}_s + \frac{d\lambda_s}{dt} \Rightarrow \frac{d\lambda_s}{dt} = \mathbf{v}_s - R_s \cdot \mathbf{i}_s \tag{3.8}$$

Calling  $\lambda_{s,l}$  the flux at low frequencies computed before and  $G_{obs}$  the observer gain, it is possible to adjust the model's parameters and match the actual motor behavior through a feedback loop:

$$\frac{d\lambda_s}{dt} = \mathbf{v}_s - R_s \cdot \mathbf{i}_s + G_{obs} \cdot (\lambda_{s,l} - \lambda_s) \quad (3.9)$$

$$\lambda_s = \frac{\mathbf{v}_s - R_s \cdot \mathbf{i}_s + G_{obs} \cdot (\lambda_{s,l} - \lambda_s)}{s} \quad (3.10)$$

From Eq.(3.7):

$$\lambda_r = \frac{\lambda_s - \sigma L_s \cdot \mathbf{i}_s}{k_r} \quad (3.11)$$

### 3.2.3 PI regulator

In motor control of an induction motor, a PI (proportional-integral) regulator is a type of control system that is used to regulate the motor's speed or torque. To implement a PI regulator in motor control of an induction motor, the motor's electrical and mechanical signals, such as the stator currents and voltages, are measured and fed back to the regulator. The regulator calculates the error between the desired and actual values and generates a control signal that adjusts the motor's input to minimize the error.

The PI regulator consists of two components: a proportional component and an integral component. The proportional component generates a control signal that is proportional to the error between the desired and actual values. The integral component generates a control signal that is proportional to the integral of the error over time [21]. The output is calculated using the PI controller formula:

$$\frac{K_i + s \cdot K_p}{s} \quad (3.12)$$

Where  $K_p$  is the proportional gain and  $K_i$  is the integral gain. These gains determine how the controller responds to changes in the error signal and how it compensates for any steady-state error. The proportional gain determines the amount of output change for a given change in the error signal. The integral gain is used to eliminate steady-state error. It accumulates the error over time and adjusts the output accordingly. The PI controller gains are calculated through frequency response of the system.

In a PI regulator with feedforward functioning, the control signal is also influenced by an estimate of the external disturbances that affect the motor's performance. This estimate is generated by a feedforward path that takes into account the external disturbances based on a model of the system. The feedforward path can improve the performance of the PI regulator by anticipating the effect of external



disturbances on the motor and compensating for them in advance. This can result in faster response times, better disturbance rejection, and more accurate control of the motor.

In a PI controller with overspeed protection, the controller is designed to limit the maximum output of the proportional term to prevent the process variable from exceeding a certain threshold. This is particularly important in systems where overspeeding can be dangerous or cause damage to equipment. One way to implement overspeed protection in a PI controller is to add a saturation function to the proportional term. The saturation function limits the output of the proportional term to a certain maximum value, which is determined based on the maximum allowable speed of the system.

The PI regulator is widely used in motor control of an induction motor because it is simple, robust, and effective. The regulator can be easily tuned to achieve the desired performance, such as fast response time, low overshoot, and good stability.

### **3.2.4 Speed control and torque estimation**

Like shown in Fig.(3.2) the reference torque is directly related to the stator current which is the input to a PI regulator that calculates the reference PWM voltage required to achieve the desired speed. In order to calculate this torque, a speed control is implemented through another PI regulator.

The PI regulator takes the difference between the desired speed setpoint and the actual speed of the motor as the error signal and computes the required reference torque to achieve the desired speed. The desired speed is set from the user interface while the actual speed is the rotor speed estimated through the PLL observer.

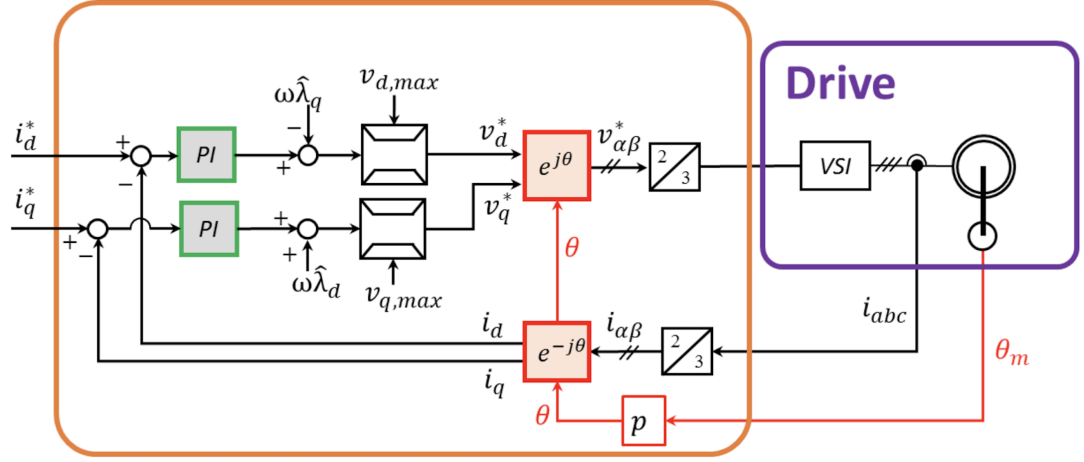
The feedforward function is not implemented, while the maximum output of the proportional term is limited through a saturation function which limit the Torque to a fixed value equal to  $24Nm$ .

For the temperature tests the reference torque is set manually from the user interface like the desired speed.

### **3.2.5 Current control and reference voltage computation**

In motor control of an induction motor the desired motor speed is achieved through the control of the voltage to be applied to a PWM modulator which regulates the switching functions of the converter power device [22]. A PI (proportional-integral)

regulator is a commonly used control technique for this purpose. A PI regulator can be designed to regulate the stator current in order to obtain the desired reference voltage at the output. We speak of current control due to the fact that the input variable to the PI to be controlled is a current [23]. Like shown in Fig.(3.4) the control of the stator current vector is in  $(dq)$  coordinates (rotating reference frame) through two PI regulators: one for the  $d$  component and one for the  $q$  component.



**Figure 3.4:** Current control block diagram [12].

Like said before, the regulator calculates the error between the desired (reference) and actual values. In d-axis the reference current ( $i_d^*$ ) is set equal to 6.351A, which is the value of the magnetizing current. In q-axis instead the reference current  $i_q^* = \frac{T^*}{\frac{3}{2} \cdot p k_r \lambda_r}$ , where:

1.  $T^*$  is the reference Torque computed through the speed control discussed before;
2.  $p$  is the pole-pairs number and is equal to 2;
3.  $\lambda_r$  is the magnitude of the rotor flux vector.

Feedforward is a control strategy in which a system's output is adjusted in anticipation of a known or expected disturbance or change in input. The feedforward component ( $ffw$ ) in current control is given by the motional term of Eq.(2.10) ( $ffw_d = -\omega \lambda_q$  and  $ffw_q = +\omega \lambda_d$ ). The feedforward component is also shown in Fig.(3.4).

For any drive, every control variable must be limited to work in a safe and reliable operating range. For this reason the overvoltage protection, which limits the maximum voltage of the proportional term, is included. By limiting the voltage, the motor can be protected from excessive speed and associated damage:

1.  $v_{d,max} = \frac{v_{dc}}{\sqrt{3}}$  is the voltage limit set in d-axis;
2.  $v_{q,max} = \sqrt{(\frac{v_{dc}}{\sqrt{3}})^2 - (v_d^*)^2}$  is the voltage limit set in q-axis;

The current control output is therefore the voltage vector:  $v_{dq}^* = \begin{bmatrix} v_d^* \\ v_q^* \end{bmatrix}$

### 3.2.6 Duty Cycle computation

To obtain the command signals of PWM from the reference voltage of a motor control system for an induction motor it is necessary to calculate the instantaneous duty cycles. The reference voltage vector  $v_{dq}^*$  in rotating reference frame is transformed in  $v_{abc}^*$  in three-phase time-domain in order to compute the duty cycle in the following way:

$$\begin{aligned} d_a &= \frac{1}{2} + \frac{v_a^* + v_{cm}}{v_{dc}} \\ d_b &= \frac{1}{2} + \frac{v_b^* + v_{cm}}{v_{dc}} \\ d_c &= \frac{1}{2} + \frac{v_c^* + v_{cm}}{v_{dc}} \end{aligned} \tag{3.13}$$

Where  $v_{cm}$  is the common mode voltage which depends only on the chosen modulation technique. The modulation technique used in this case is the "MinMax modulation" with which the common mode is calculated in order to balance the positive and negative envelope of the reference voltages [11]:

$$v_{cm} = -\frac{\max(v_a^*, v_b^*, v_c^*) + \min(v_a^*, v_b^*, v_c^*)}{2} \tag{3.14}$$

The PWM command signal is generated by comparing the instantaneous duty cycle to the carrier signal using a comparator circuit. The output of the comparator is a series of pulses that have a width proportional to the duty cycle. The PWM signal is then used to control the switching of the power electronics in the inverter, which generates the three-phase output voltage that drives the induction motor.

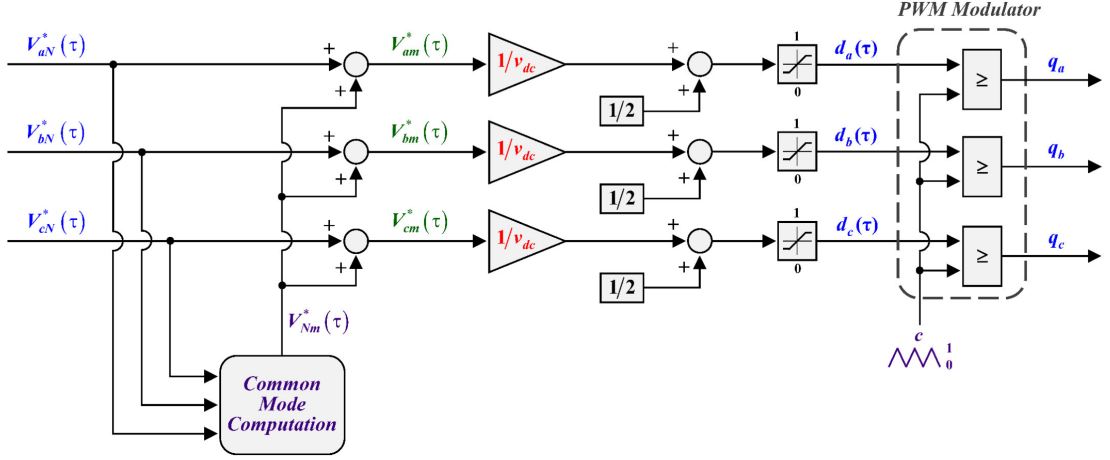


Figure 3.5: MinMax modulation block diagram [10].

### 3.3 Thermal model and temperature estimation

The purpose of this thesis, as already mentioned, is the implementation of a thermal model that is able to simulate the thermal behavior of the stator windings. A temperature "estimator" has also been implemented which is capable of predicting the temperature reached by the windings after a certain period of time. The thermal model is a first order model. The values of the model parameters  $C_{eq,w}$  and  $R_{eq,w}$  are the same of the second order model implemented on the Simulink model, and are obtained, like shown in [7], [8] and [9], considering the first minutes samples.

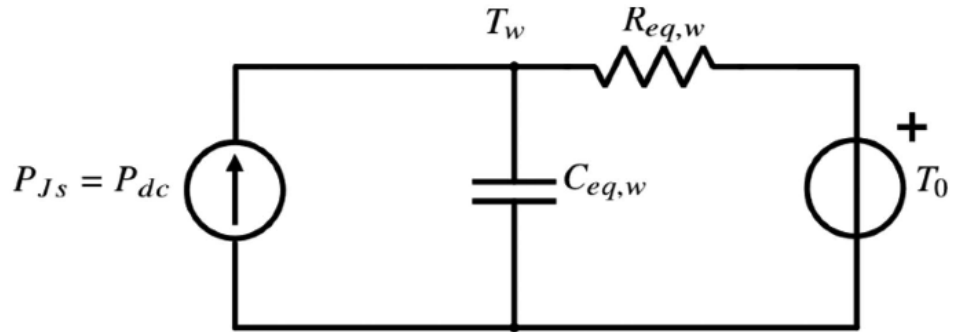


Figure 3.6: First order thermal model equivalent circuit [7].

From Fig.(3.6) is possible to obtain the overtemperature equation of the stator windings, with respect to the initial temperature, updated after each sampling instant and therefore the temperature reached after each sampling instant:

$$\Delta_T = \frac{P_{js}}{C_{eq,w}} \cdot T_s + \Delta_T \cdot \left(1 - \frac{T_s}{C_{eq,w} R_{eq,w}}\right) \quad (3.15)$$

$$T_w = \Delta_T + T_0 \quad (3.16)$$

In this way it is also possible to calculate the temperature that the windings will reach after a given period of time  $T_{final}$  chosen by the user interface at the time of the tests:

$$T_f = \left(1 - e^{-\frac{T_{imef}}{\tau}}\right) \cdot P_{js} \cdot R_{eq,w} + T_0 \quad (3.17)$$

Where:

1.  $\Delta_T$  is the overtemperature with respect to the initial temperature  $T_0$ ;
2.  $T_s$  is the sampling time: the overtemperature estimation is calculated after each  $T_s$  time interval which is set equal to 0.5 seconds;
3.  $\tau = C_{eq,w} \cdot R_{eq,w}$  is the thermal time constant.
4.  $T_f$  is the temperature that the windings should reach after a period of time  $T_{final}$

### 3.4 Motor Control implementation in C code

Here it is shown how the motor control has been implemented in C code in the various parts explained above.

The PLL observer for calculating the speed has been implemented via a structure, its parameters are calculated through a function and then used as follows:

**PLL observer structure:**

```

1 typedef struct { double ref_sin;
2                 double ref_cos;
3                 double pll_sin;
4                 double pll_cos;
5                 double error;
6                 double kp;

```

```

7         double ki;
8         double prop;
9         double integral;
10        double out;
11        double ffw;
12        double angle;
13    } Xpll;

```

### PLL observer function:

```

1
2 void PllSpeedCompute (Xpll *pll) {
3
4     // Compute PLL Error
5     pll->error = pll->ref_sin * pll->pll_cos - pll->ref_cos * pll->
pll_sin;
6
7     // Compute Proportional (with ffw)
8     pll->prop = pll->kp * pll->error;
9
10    // Compute Integral
11    pll->integral += Ts * pll->ki * pll->error;
12
13    // Compute PLL Speed
14    pll->out = pll->prop + pll->integral + pll->ffw;
15
16    // Compute PLL Angle
17    pll->angle += Ts * pll->out;
18
19    // Normalization of PLL Angle
20    pll->angle -= double_pi * floor(pll->angle * 0.5 * one_over_pi);
21    if (pll->angle > pi)
22        pll->angle -= double_pi;
23    if (pll->angle < -pi)
24        pll->angle += double_pi;
25
26    // Compute PLL Angle SinCos
27    pll->pll_sin = sin(pll->angle);
28    pll->pll_cos = cos(pll->angle);
29
30 }

```

### Mechanical Speed Computation:

```

1
2     // PLL Execution
3     PllMechSpeed.ref_sin = SinCosMec.sin;

```

```

4      PllMechSpeed.ref_cos = SinCosMec.cos;
5      PllMechSpeed.kp      = Kp_PLL_speed;
6      PllMechSpeed.ki      = Ki_PLL_speed;
7      PllMechSpeed.ffw     = 0.0;
8      PllSpeedCompute(&PllMechSpeed);
9
10     // Update Mech Speed
11     MotorSpeed             = PllMechSpeed.integral;
12     MotorSpeedAbs          = fabs(MotorSpeed);
13     MotorSpeedRpm          = MotorSpeed * rad2rpm;

```

The Flux is computed through two functions: one for the low frequency and another one for the high frequency as explained before.

### Flux computation (low frequency):

```

1
2 void Lambda_low (Xabc *Aabc) {
3     Xabc Isabc;
4     Isabc.a = Aabc->a;
5     Isabc.b = Aabc->b;
6     Isabc.c = Aabc->c;
7     Xsc SinCos;
8     Xalphabeta Ialphabeta;
9     Xdq Isdq;
10
11     SinCos.cos = SinCosElt.cos;
12     SinCos.sin = SinCosElt.sin;
13
14     DirectClarke00(&Isabc, &Ialphabeta);
15     DirectRot_dq(&Ialphabeta, &SinCos, &Isdq);
16
17     Lambda_Rdq.d += ((Lm*Isdq.d - Lambda_Rdq.d)/tau_r)*Ts;
18     Lambda_Rdq.q += ((Lm*Isdq.q - Lambda_Rdq.q)/tau_r)*Ts;
19
20     InverseRot_dq(&Lambda_Rdq, &SinCos, &Lambda_r);
21     Lambda_s.alpha = Lambda_r.alpha*Kr + sigLs*Ialphabeta.alpha;
22     Lambda_s.beta  = Lambda_r.beta*Kr + sigLs*Ialphabeta.beta;
23 }

```

### Flux computation (high frequency):

```

1
2 void Lambda_high (Xabc *dabc, Xabc *Iabc) {
3     Xabc Isabc;
4     Xabc Eabc;

```

```

5   Xabc V_abc;
6   Xalphabeta Ealphabeta;
7   Xalphabeta Ialphabeta;
8
9   Isabc.a = Iabc->a;
10  Isabc.b = Iabc->b;
11  Isabc.c = Iabc->c;
12  V_abc.a = (vdc/3)*(2*dabc->a - dabc->b - dabc->c);
13  V_abc.b = (vdc/3)*(2*dabc->b - dabc->c - dabc->a);
14  V_abc.c = (vdc/3)*(2*dabc->c - dabc->a - dabc->b);
15  Eabc.a = V_abc.a-Rs*Isabc.a;
16  Eabc.b = V_abc.b-Rs*Isabc.b;
17  Eabc.c = V_abc.c-Rs*Isabc.c;
18
19  DirectClarke00(&Eabc, &Ealphabeta);
20
21  Lambda_sh.alpha = Lambda_sh_next.alpha;
22  Lambda_sh.beta = Lambda_sh_next.beta;
23
24  Lambda_sh_next.alpha += (Ealphabeta.alpha + obs_gain * (Lambda_s.
alpha-Lambda_sh.alpha))*Ts;
25  Lambda_sh_next.beta += (Ealphabeta.beta + obs_gain * (Lambda_s.
beta-Lambda_sh.beta))*Ts;
26
27  DirectClarke00(&Isabc, &Ialphabeta);
28
29  Lambda_rh.alpha = (Lambda_sh.alpha - sigLs*Ialphabeta.alpha)/Kr;
30  Lambda_rh.beta = (Lambda_sh.beta - sigLs*Ialphabeta.beta)/Kr;
31 }

```

### Flux computation:

```

1   Lambda_low(&iabc);
2   Lambda_high(&duty,&iabc);
3

```

The desired speed is set from the user interface by changing the value of the "mech ref" parameter:

### Speed setpoint:

```

1   // Saturation of Speed Reference
2   SpeedSetPoint = two_level_saturation(mech_ref, SPEED_LIMIT, -
3   SPEED_LIMIT);
4

```



```

5 // Slew Rate of Speed Reference
6 OmegaRef = slew_rate_limit(OmegaRef, (mech_rate * rpm2rad *
Ts), (SpeedSetPoint * rpm2rad));

```

The reference torque is calculated through a PI regulator. The PI regulator is implemented through a structure, its parameters are calculated through a function and then used in the "main" as already done for the PLL observer:

### PI regulator structure:

```

1
2 typedef struct { double ref;
3                 double ref_old;
4                 double out;
5                 double actual;
6                 double error;
7                 double prop;
8                 double integral;
9                 double lim_high;
10                double lim_low;
11                double int_lim_high;
12                double int_lim_low;
13                double ffw;
14                double kp;
15                double ki;
16            } XPIReg;

```

### PI regulator function:

```

1
2 // PI with ffw and anti wind-up
3 void PIReg (XPIReg *Reg) {
4
5     // Compute Error
6     Reg->error = Reg->ref - Reg->actual;
7
8     // Compute Proportional (plus FFw)
9     Reg->prop = (Reg->kp * Reg->error) + Reg->ffw;
10
11    // Saturation of Proportional
12    if (Reg->prop > Reg->lim_high)
13        Reg->prop = Reg->lim_high;
14    if (Reg->prop < Reg->lim_low)
15        Reg->prop = Reg->lim_low;
16
17    // Compute Limits of Integral

```

```

18 Reg->int_lim_high = Reg->lim_high - Reg->prop;
19 Reg->int_lim_low = Reg->lim_low - Reg->prop;
20
21 // Compute Integral
22 Reg->integral += Ts * Reg->ki * Reg->error;
23
24 // Saturation of Integral
25 if (Reg->integral > Reg->int_lim_high)
26     Reg->integral = Reg->int_lim_high;
27 if (Reg->integral < Reg->int_lim_low)
28     Reg->integral = Reg->int_lim_low;
29
30 // Compute Unsaturated PI output
31 Reg->out = Reg->prop + Reg->integral;
32
33 // Saturation of PI output
34 if (Reg->out > Reg->lim_high)
35     Reg->out = Reg->lim_high;
36 if (Reg->out < Reg->lim_low)
37     Reg->out = Reg->lim_low;
38
39 }

```

### Torque reference computation:

```

1
2 // Set Parameters of PI Speed
3 speedPI.ref = OmegaRef;
4 speedPI.actual = MotorSpeed;
5 speedPI.kp = Kp_PI_speed;
6 speedPI.ki = Ki_PI_speed;
7 speedPI.lim_high = T_max;
8 speedPI.lim_low = -T_max;
9
10 // Start PI Speed Code
11 PIReg (&speedPI);
12
13 // Load Speed from PI
14 T_ref = speedPI.out;

```

As said before the reference torque, for the temperature tests, is set from user interface, like the desired speed, by changing the value of the "mech rate" parameter:

### Torque setpoint:

```

1

```

```

2 // Saturation of Torque Reference
3 TorqueSetPoint = two_level_saturation(mech_ref, T_max, 0.0);
4
5 // Slew Rate of Torque Reference
6 T_contr = slew_rate_limit(T_contr, (mech_rate * Ts), (
TorqueSetPoint));

```

The current control is implemented through the PI regulator shown before and then used in the "main":

### Current control:

```

1
2 // Current Control Loop
3 // Set PI d Current Control Parameters
4
5 dcurrentreg.ref = Imagnet;
6 dcurrentreg.actual = idq.d;
7 dcurrentreg.kp = Kp;
8 dcurrentreg.ki = Ki;
9 dcurrentreg.ffw = Rs * idq.d - (puls_obs * Lambda_sh_dq.q);
10 dcurrentreg.lim_high = vdc*sqrt3_inv;
11 dcurrentreg.lim_low = -dcurrentreg.lim_high;
12
13 // Start PI d Control Code
14 PIReg(&dcurrentreg);
15
16 // Load d Reference Voltage
17 vdqRef.d = dcurrentreg.out;
18
19 // Set PI q Current Control Parameters
20
21 qcurrentreg.ref = T_contr / (1.5*p*Lambda_r_abs*Kr);
22 qcurrentreg.actual = idq.q;
23 qcurrentreg.kp = Kp;
24 qcurrentreg.ki = Ki;
25 qcurrentreg.ffw = Rs * idq.q + (puls_obs * Lambda_sh_dq.d);
26 qcurrentreg.lim_high = sqrt(pow(vdc*sqrt3_inv,2.0)-pow(vdqRef
.d,2.0));
27 qcurrentreg.lim_low = -qcurrentreg.lim_high;
28
29 // Start PI q Control Code
30 PIReg(&qcurrentreg);
31
32 // Load q Reference Voltage
33 vdqRef.q = qcurrentreg.out;

```

The duty cycle are computed through a function:

### Duty cycle computation:

```

1 void DutyComp(Xabc *VrefInv, Xabc *Dutyprova) {
2     float Cm;
3     float max;
4     float min;
5
6
7     // Common Mode Voltage Computation
8     max=VrefInv->a;
9     if (VrefInv->b>max)
10         max=VrefInv->b;
11     if (VrefInv->c>max)
12         max=VrefInv->c;
13
14     min=VrefInv->a;
15     if (VrefInv->b<min)
16         min=VrefInv->b;
17     if (VrefInv->c<min)
18         min=VrefInv->c;
19
20     Cm = -((max+min)/2.0);
21     Dutyprova->a = 0.5 + ((VrefInv->a + Cm)/ vdc);
22     Dutyprova->b = 0.5 + ((VrefInv->b + Cm)/ vdc);
23     Dutyprova->c = 0.5 + ((VrefInv->c + Cm)/ vdc);
24
25     // Duty-Cycles Saturation
26     if (Dutyprova->a > 1.0)
27         Dutyprova->a = 1.0;
28     if (Dutyprova->a < 0.0)
29         Dutyprova->a = 0.0;
30
31     if (Dutyprova->b > 1.0)
32         Dutyprova->b = 1.0;
33     if (Dutyprova->b < 0.0)
34         Dutyprova->b = 0.0;
35
36     if (Dutyprova->c > 1.0)
37         Dutyprova->c = 1.0;
38     if (Dutyprova->c < 0.0)
39         Dutyprova->c = 0.0;
40 }

```

Finally the thermal model and the estimation of the final temperature have been implemented as follows.

**Pjs computation:**

```

1 void PJS_function (Xalphabet * Iab) {
2   Rs_T = Rs*((234.5 + (T_wire - 273.15)) / (234.5 + 25));
3   IpeakSquare = pow(Iab->alpha, 2.0) + pow(Iab->beta, 2.0);
4   Pjs = (1.5)*Rs_T*IpeakSquare;
5   Irms = sqrt(IpeakSquare)/sqrt(2);
6 }

```

**Thermal control:**

```

1 TMIS = (RMISFilt/R0)*(234.5 + (Ta0-273.15)) - 234.5 + 273.15;
2 TMIS_MA = (RMIS_MA/R0)*(234.5 + (Ta0-273.15)) - 234.5 + 273.15;
3
4
5 ThresholdIrmsStart = Tselivector*0.15;
6 if (abs(Tselivector - TselivectorOLD) < Thres_Tobs && flag == 0 &&
7     abs(TorqueT40Filt - Tselivector) < ThresholdIrmsStart)
8 {
9     tau_th = R1*C1;
10    T0 = Ta;
11    DeltaT_wire = 0.0;
12    T_f = (1.0 - exp(-(Time_fin/tau_th)))*Pjs*R1 + T0;
13    flag = 1;
14    counter_th = 0;
15    ExecThMod = 0;
16 }
17 if (flag == 1 && abs(Tselivector) < Thres_Tobs)
18     flag = 0;
19
20 if (flag == 1)
21 {
22     counter_th++;
23     if (counter_th == (int) floor(Ts_th*fs)) {
24         DeltaT_wire = Pjs/C1*Ts_th + DeltaT_wire*(1.0 - Ts_th/R1/
25         C1);
26         T_wire = DeltaT_wire + T0;
27         counter_th = 0;
28         ExecThMod = 1;
29     }
30     else
31     {
32         ExecThMod = 0;
33     }
34 }

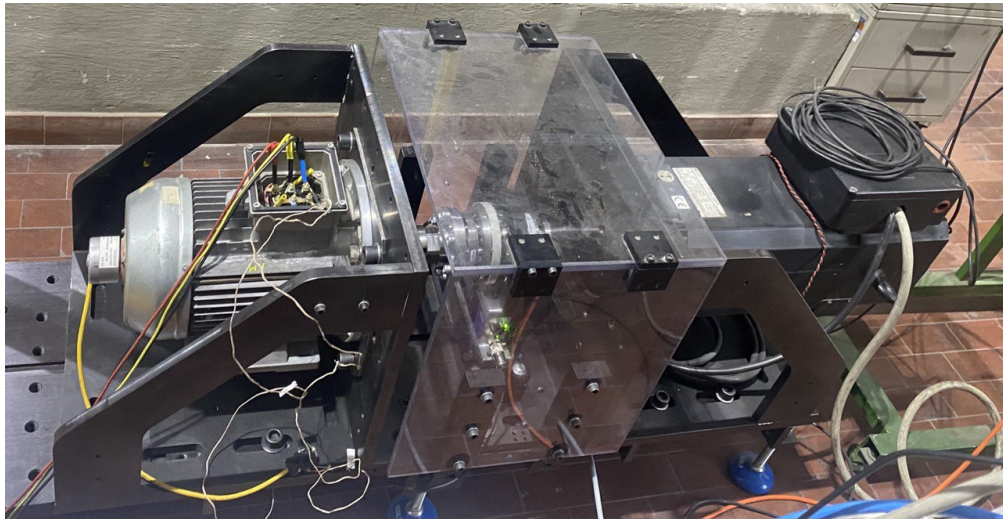
```

}

## Chapter 4

# Results

The motor control was designed by testing it initially on the MATLAB/Simulink model developed in chapter 2: in this way it was possible to verify that the input data, such as speed and current, were measured correctly and that, step by step, each variable behaved as desired and expected. After checking and verifying that the motor control worked correctly on the motor model, it was finally possible to perform bench tests and verify that the motor control made was actually able to control a real induction motor as desired. This chapter will present the results obtained from the experiments carried out to evaluate the effectiveness of this work.



**Figure 4.1:** Induction motor.

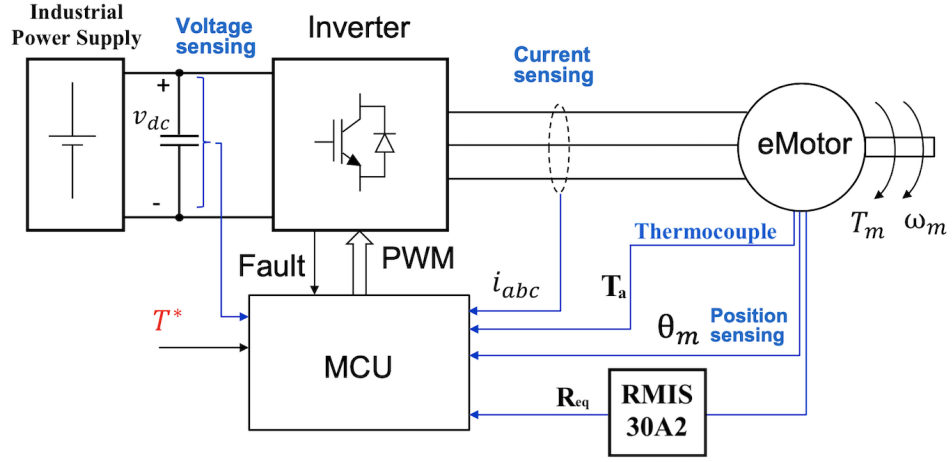


Figure 4.2: Block diagram of the motor bench.

The data capturing, visualization and layout designing is done by using the dSPACE Control Desk which is a system identification toolbox that can be used for Rapid control prototyping (RCP) and testing purposes. The Real time interfacing (RTI) is done by using dSPACE controller board and controller panel. In order to better evaluate the collected measurement data, these are placed in the Measurement Data Manager and exported in standard formats such as MAT format.

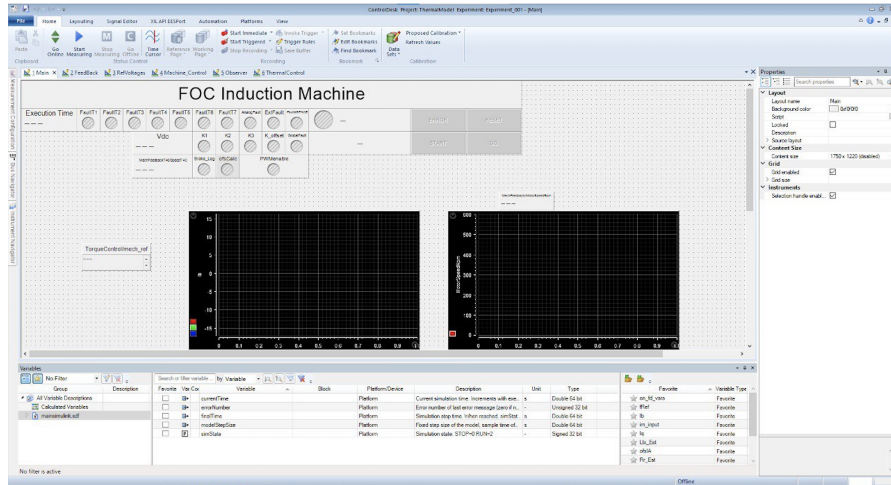


Figure 4.3: dSPACE Control Desk interface.



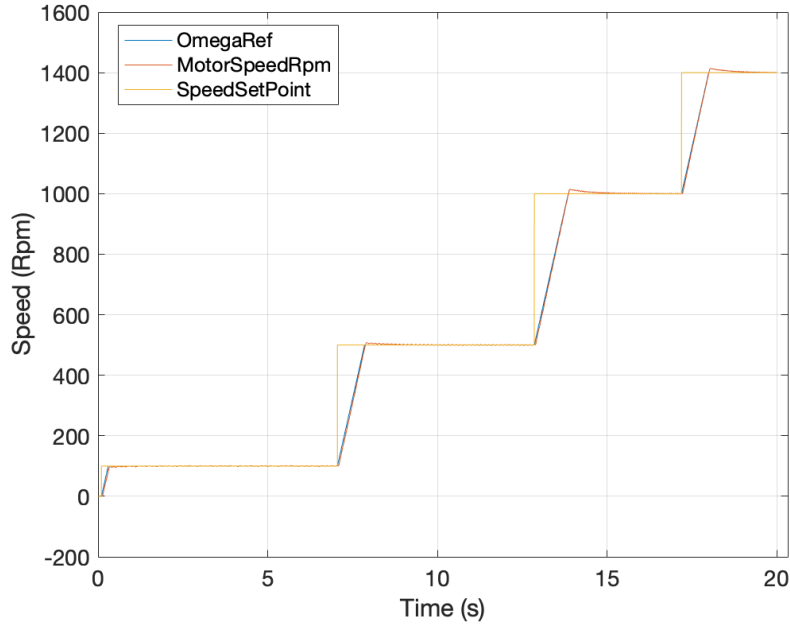
## 4.1 Results validation

The first step to ensure that the tests were carried out correctly, even before performing the thermal tests on the windings, was to verify that the motor control was actually able to control the dynamic behavior of the motor in terms of speed and torque. For this purpose, tests were carried out in which the desired speed was changed several times from the user interface. The purpose of these tests is to verify that the speed calculated from the input data through the PLL observer followed the trend of the manually set speed. The reference torque estimated by the first PI regulator was also compared with the torque observed from the input stator current and flux in order to verify that the two parameters had the same behaviour. This latter torque is obtained from Eq.(2.22):

$$T_e = \frac{3}{2} \cdot p \cdot (\lambda_s \wedge \mathbf{i}_s) \quad (4.1)$$

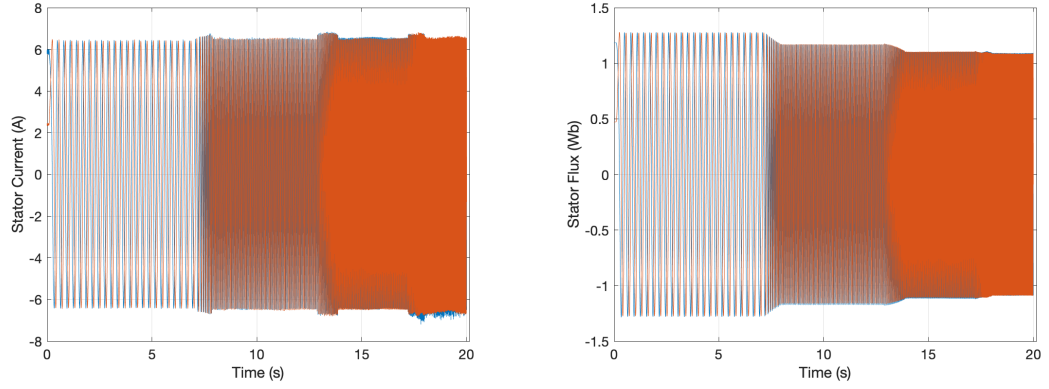
Exploiting  $\alpha\beta$  reference frame:

$$T_e = \frac{3}{2} \cdot p \cdot (\lambda_{s,\alpha} \cdot i_{s,\beta} - \lambda_{s,\beta} \cdot i_{s,\alpha}) \quad (4.2)$$

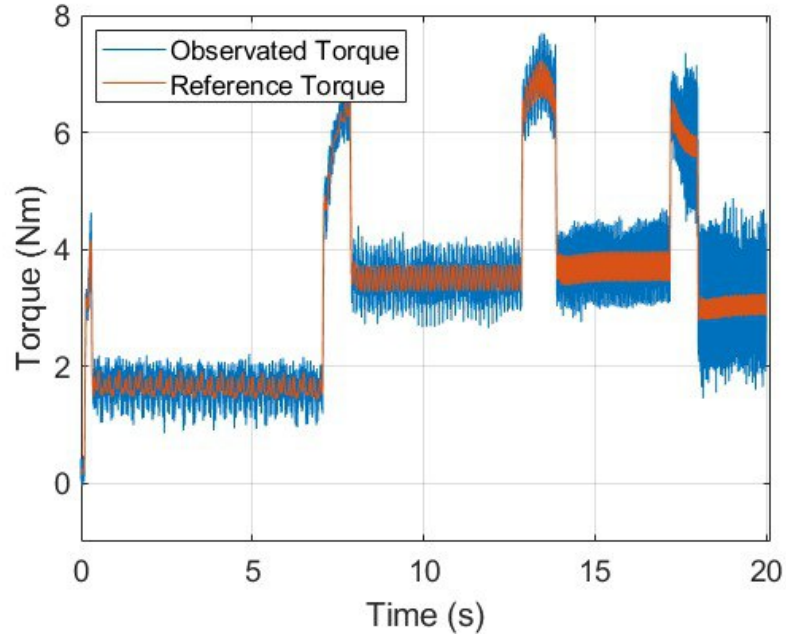


**Figure 4.4:** Speed control test.

The speeds profile of Fig.(4.4) shows the correct functioning of the speed controller. In fact the value of the "SpeedSetPoint" parameter is the one set from the user interface and changes immediately with a step response, followed after a short time by the value of the desired speed, set directly by the "SpeedSetPoint" parameter, and by the speed value reached by the motor "MotorSpeedRpm".



**Figure 4.5:** Stator current (left) and stator flux (right) in  $\alpha\beta$  reference frame.



**Figure 4.6:** Comparison of torques behavior.

Also the trend of the torque of Fig.(4.6) shows a correct operation of the motor control. In fact the observed torque tracks well the reference torque provided by the speed controller.

Obtained this data and obtained the confirmation that the motor follows the desired behavior and set by the motor control, it was possible to continue with the tests and carry out those relating to the thermal control.

As already mentioned several times, the goal of this thesis is to be able to reproduce and predict the thermal behavior of the stator windings, which are the most sensitive part to thermal overloads in the induction motor. For this purpose, the temperature measurement was obtained in the laboratory using a thermocouple and compared with the trend estimated by the model implemented in the motor control and with the final temperature predicted by the same model.

What is expected to obtain are two functions, representing the thermal transient obtained by measuring the stator windings and the one estimated by the thermal model, with the same trend and which reach the predicted temperature after the evaluated time interval. The tests are performed by setting the motor speed to 600 rpm. The load torque is initially set to 0 Nm, data acquisition is started by changing the load torque value and measurements are acquired for a duration of 10 minutes.

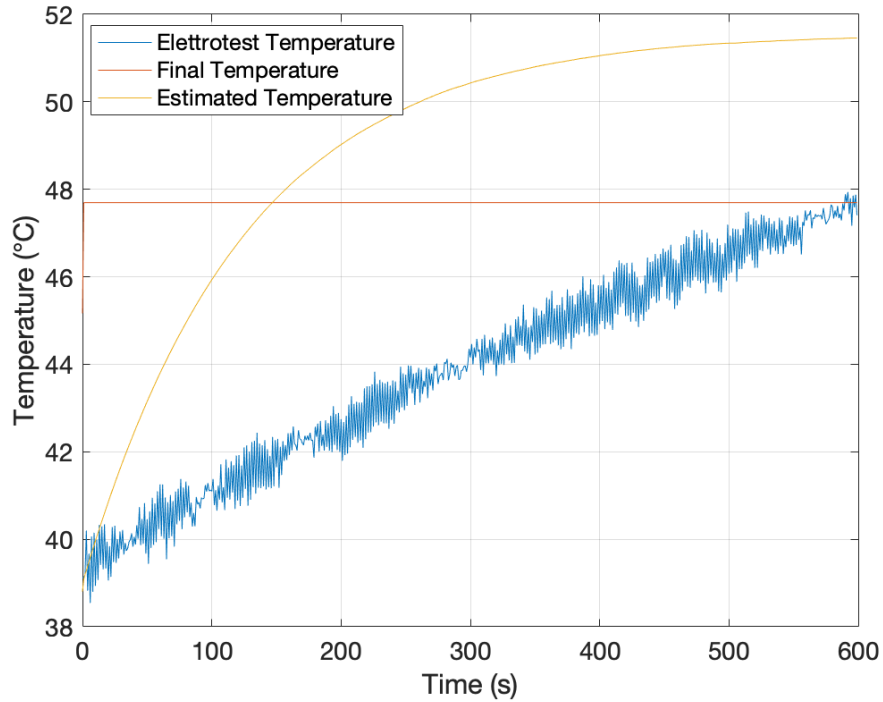
At this point, after some initial tests, the following considerations were made:

1. In the thermal model, the estimate of the temperature that the windings should reach after 10 minutes is calculated taking into consideration the initial value of the joule losses of the stator ( $P_{js}$ ), while the calculation of the temperature of the windings updated every 0.5 seconds takes into consideration the joule losses of that instant. This implies that as the joule losses vary, the temperature of the windings increases more or less rapidly and does not reach the final temperature exactly in the 10 minutes predicted.
2. The equivalent thermal resistance of the thermal model is initially set to 0.07 K/W, however this thermal model is a first order model and does not take into account some motor cooling factors such as ventilation and forced convection. For this reason it is decided to lower the value of the thermal resistance to 0.063 K/W.
3. The temperature measurement of the windings was initially acquired through a thermocouple, however this measurement is not directly on the windings but on the motor frame. This does not allow a correct measurement of the

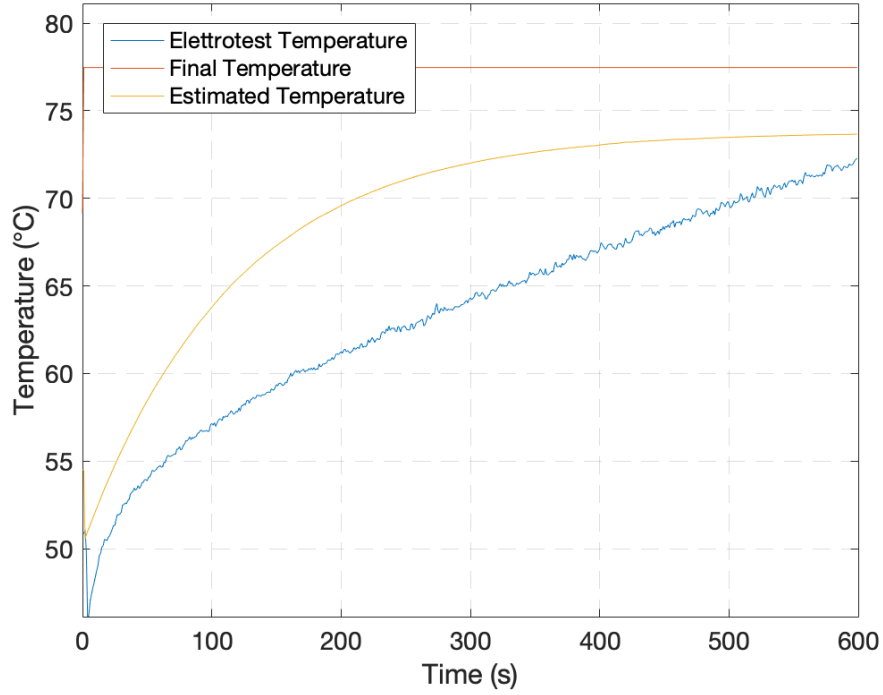
temperature of the windings which is lower than it actually is. To obtain a more accurate measurement it was decided to proceed with the measurement through the elettrotest RMIS 30A2 which directly measures the thermal resistance of the windings and calculates the temperature.

Taking into account the above considerations and making possible modifications to the measurement system and the thermal model, new tests were performed. By setting the load torque value to 8 Nm, the results shown in Fig.(4.7) are obtained, while by setting the value to 26 Nm, the results shown in Fig.(4.8) are obtained.

In Fig.(4.7), as explained above, the thermal transient estimated by the thermal model reaches the predicted temperature after 2 minutes instead of 10 minutes. However the measured temperature of the windings reaches the temperature predicted by the thermal model in exactly 10 minutes, furthermore the value of the temperature measured at the end of the transient deviates by only 4 degrees from the temperature estimated from the thermal model at the end of the transient.



**Figure 4.7:** Thermal model validation with a load torque equal to 8 Nm.



**Figure 4.8:** Thermal model validation with a load torque equal to 26 Nm.

In Fig.(4.8) the thermal transient estimated by the thermal model doesn't reaches the predicted temperature in 10 minutes. Not even the measured temperature of the windings reaches the temperature predicted by the thermal model in 10 minutes, however, at the end of the transient, the value of the measured temperature and the value of the temperature estimated from the thermal model deviate from each other by only 1 degree and by about 5 degrees from the value they were predicted to reach.

## Chapter 5

# Conclusion

Induction motors are often subject to stress caused by repeated fast and short torque transients. These working conditions could lead to overheating some parts of the motor. The most sensitive parts of an induction motor to thermal overloads are the stator windings. This thesis proposes a thermal model of the stator windings which is able to simulate the thermal trend due to heating in induction motor working conditions. A temperature "estimator" has also been implemented which is capable of predicting the temperature reached by the windings after a certain period of time. This model is based on the relationship between motor power loss and temperature, which can be obtained from experimental data or computer simulations. Obtaining an accurate estimate of the thermal behavior also allows us to understand how much it is possible to exploit the motor in certain conditions without reaching temperatures that could damage the windings and therefore the motor itself.

For this purpose, a first-order thermal model is proposed in this thesis. The obtained results are shown in Fig.(4.7) and in Fig.(4.8). The expected results should show the three temperature curves which, at the end of the transient, converge to the same value. The results obtained from the tests show instead that these three temperature curves never reach the same precise value. This failure is attributable to the simplicity of the first order thermal model which does not take into account some thermal phenomena. However, since at the end of the thermal transient all three values considered are within a range of a few degrees, the results obtained are considered acceptable.

In order to make the estimation of the temperature of the stator windings more accurate, it would have been possible to implement a thermal model of the second order or higher which also takes into consideration the motor cooling phenomena. However, implementing an high-order thermal model would have been

much more complex and unnecessary for the purposes of this thesis, given that for the application of an induction motor an uncertainty of a few degrees is considered acceptable.





# Bibliography

- [1] Dmitry Levkin. *Single-phase induction motor*. <https://en.engineering-solutions.ru/motorcontrol/induction1ph/>. Accessed: 2023-01-30 (cit. on p. 2).
- [2] Radu Bojoi and Paolo Pescetto. *Fundamentals of vector control*. University Lecture (cit. on p. 5).
- [3] Tripti Rai and Prashant Debre. «Generalized Modeling Model Of three phase Induction Motor». In: *2016 International Conference on Energy Efficient Technologies for Sustainability (ICEETS)*. IEEE (2016), pp. 927–931 (cit. on p. 6).
- [4] Gordon R. Slemon. «Modelling of Induction Machines for Electric Drives». In: *IEEE Transactions on Industry Applications* vol.25, no.6, (Nov. 1989), pp. 1126–1131 (cit. on p. 7).
- [5] Yin Jun, Lu Xiaoli, Wei Yunbing, and Cui Guangzhao. «Simulation and Research of Induction Motor Considering Iron Loss in Stationary Reference Frame». In: *2010 International Conference on Computing, Control and Industrial Engineering* (2010), pp. 446–449 (cit. on p. 11).
- [6] Nikola Z. Popov, Slobodan N. Vukosavic, and Emil Levi. «Motor Temperature Monitoring Based on Impedance Estimation at PWM Frequencies». In: *IEEE Transactions on Energy Conversion* vol.29, no.1, (Mar. 2014), pp. 215–223 (cit. on p. 12).
- [7] Eric Armando, Aldo Boglietti, Fabio Mandrile, Enrico Carpaneto, Sandro Rubino, and devi Geetha Nair. «Definition and Experimental Validation of a Second-Order Thermal Model for Electrical Machines». In: *IEEE Transactions on Industry Applications* vol.57, no.6, (Sept. 2021), pp. 5969–5982 (cit. on pp. 13, 14, 36).
- [8] Aldo Boglietti, Silvio Vaschetto, Marco Cossale, and Thiago Dutrae. «Winding Thermal Model for Short-Time Transient: Experimental Validation in Operative Conditions». In: *IEEE Transactions on Industry Applications* vol.54, no.2, (Mar. 2018), pp. 1312–1319 (cit. on pp. 14, 36).

- [9] Aldo Boglietti, Enrico Carpaneto, Marco Cossale, and Silvio Vaschetto. «Stator-Winding Thermal Models for Short-Time Thermal Transients: Definition and Validation». In: *IEEE Transactions on Industrial Electronics* vol.63, no.5, (May 2016), pp. 2713–2721 (cit. on pp. 14, 36).
- [10] Sandro Rubino. *Componenti e tecnologie per l'elettificazione dei sistemi di trazione nei veicoli stradali*. University Lecture. 2020 (cit. on pp. 16, 36).
- [11] Alex Ruderman. «Three-phase multi-level PWM rectifier multi-carrier discontinuous voltage modulation strategy». In: *2007 European Conference on Power Electronics and Applications* (2007), pp. 1–9 (cit. on pp. 16, 35).
- [12] Radu Bojoi and Paolo Pescetto. *Field Oriented Control (FOC) of eDrives*. University Lecture (cit. on pp. 26, 27, 34).
- [13] Ibrahim Senol, Nur Bekiroglu, and Selin Ozcira. «Design and Application of A New Sensorless Induction Motor Drive Implemented by Using Field Oriented Vector Control Method». In: *4th International Conference on Power Engineering, Energy and Electrical Drives* (May 2013), pp. 1543–1547 (cit. on p. 27).
- [14] Sandro Rubino, Radu Bojoi, Emil Levi, and Obrad Dordevic. «Vector Control of Multiple Three-Phase Permanent Magnet Motor Drives». In: *IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society* (2018), pp. 5866–5871 (cit. on p. 27).
- [15] Michael H. Bierhoff. «A General PLL-Type Algorithm for Speed Sensorless Control of Electrical Drives». In: *IEEE Transactions on Industrial Electronics* vol.64, no.12, (Dec. 2017), pp. 9253–9260 (cit. on p. 28).
- [16] Mihai Comanescu and Longya Xu. «An Improved Flux Observer Based on PLL Frequency Estimator for Sensorless Vector Control of Induction Motors». In: *IEEE Transactions on Industrial Electronics* vol.53, no.1, (Feb. 2006), pp. 50–56 (cit. on p. 28).
- [17] Prasun Mishra, Cristian Lascu, Michael Møller Bech, Bjørn Rannestad, and Stig Munk-Neilsen. «Design and Analysis of PLL Speed Estimator for Sensorless Rotor-Flux Oriented Control of Induction Motor Drives». In: *2021 IEEE Energy Conversion Congress and Exposition (ECCE)* (2021), pp. 4743–4747 (cit. on p. 28).
- [18] Haoye Qin and Zhong Wu. *Angle Tracking Observer with Improved Accuracy for Resolver-to-Digital Conversion*. <https://www.mdpi.com/2073-8994/11/11/1347>. Accessed: 2023-01-30 (cit. on p. 28).

- [19] Myoung-Ho Shin, Dong-Seok Hyun, Soon-Bong Cho, and Song-Yul Choe. «An improved stator flux estimation for speed sensorless stator flux orientation control of induction motors». In: *PESC 98 Record. 29th Annual IEEE Power Electronics Specialists Conference* vol.2, (1998), pp. 1581–1586 (cit. on p. 29).
- [20] Amit Kumar and Tejavathu Ramesh. «Direct Field Oriented Control of Induction Motor Drive». In: *2015 Second International Conference on Advances in Computing and Communication Engineering* (2015), pp. 219–223 (cit. on p. 31).
- [21] Eun-Chul Shin, Tae-Sik Park, Won-Hyun Oh, and Ji-Yoon Yoo. «A design method of PI controller for an induction motor with parameter variation». In: *IECON'03. 29th Annual Conference of the IEEE Industrial Electronics Society* vol.1, (2003), pp. 408–413 (cit. on p. 32).
- [22] Siamak Masoudi, Mohammad Reza Feyzi, and Mohammad Bagher Bannae Sharifian. «Speed control in vector controlled induction motors». In: *2009 44th International Universities Power Engineering Conference (UPEC)* (2009), pp. 1–5 (cit. on p. 33).
- [23] Marian P. Kazmierkowski and Luigi Malesani. «Current Control Techniques for Three-Phase Voltage-Source PWM Converters: A Survey». In: *IEEE Transactions on Industrial Electronics* vol.45, no.5, (Oct. 1998), pp. 691–703 (cit. on p. 34).