Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

University of Padua

Department of Information Engineering

Master's Degree in Control Systems Engineering

Implementation of Flux Polar Control in IPM Synchronous Motor

Candidate: Monther Raed Suleiman Al Khoshman

Student ID: 2071303

Supervisor:

Prof. Nicola Bianchi

Co-supervisor:

Dr. Ludovico Ortombina

Academic Year 2025/2026

April 2026

Contents

List of Figures

List of Tables

Abstract

Synchronous motor drives are widely employed in high-performance electric applications due to their high torque density, efficiency, and wide operating range. However, achieving effective torque control remains challenging because of magnetic cross-coupling, parameter sensitivity, and the nonlinear behavior of the machine, especially in flux-weakening conditions.

Flux Polar Control (FPC) is a control strategy based on the direct regulation of the stator flux vector in polar coordinates through the independent control of flux magnitude and flux angle. This formulation provides a physically meaningful interpretation of torque production and offers a structured framework for the control of synchronous machines.

This thesis presents the modelling, implementation, and simulation-based validation of a Flux Polar Control strategy for synchronous motor drives. A linear Interior Permanent Magnet Synchronous Motor (IPMSM) model is first developed and used as a reference framework for controller design and initial validation. In this case, the stator flux vector is reconstructed through a voltage-model-based flux estimator derived from the stator voltage equations. The implemented control structure includes reference generation for Maximum Torque Per Ampere (MTPA) and flux-weakening operation, together with PI regulation loops, voltage limitation, and anti-windup compensation.

In a second stage, the same control architecture is applied to a nonlinear map-based synchronous reluctance motor model, where the machine behavior is described directly through flux-linkage, torque, and differential inductance lookup tables. This allows magnetic saturation and cross-coupling effects to be represented more realistically, providing a more demanding validation environment for the controller.

The obtained simulation results confirm that the proposed Flux Polar Control structure is capable of preserving stable and physically coherent behavior in both linear and nonlinear operating conditions. In particular, the nonlinear implementation reproduces the expected transition from the nominal operating region to flux-weakening conditions, while also highlighting the practical limitations introduced by voltage constraints and nonlinear machine behavior.

Overall, the work confirms the viability of Flux Polar Control as a structured and effective control approach for synchronous motor drives and highlights the importance of nonlinear machine modelling for a realistic assessment of control performance.

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Prof. Nicola Bianchi, for his guidance, support, and insightful feedback throughout the development of this thesis. His expertise in control systems and his encouragement were essential to shaping the direction and quality of this work.

I am also deeply grateful to Dr. Ludovico Ortombina for his valuable assistance, especially in the modelling of synchronous machines and the practical aspects of electric drive control. His advice and technical discussions greatly contributed to the depth of this research.

My sincere thanks extend to the University of Padova for providing an excellent academic environment and the tools necessary to carry out this project.

I would like to thank my family for their unwavering love, patience, and support. Their encouragement has been a constant source of strength throughout my studies. I am especially grateful to my wife, Sajeda, whose kindness, understanding, and continuous motivation made this journey possible. My children, Yahya and Noor , have brought joy and purpose to every step of the process.

Finally, I thank everyone who contributed, directly or indirectly, to the completion of this thesis.

Chapter 1

Introduction

Rapid electrification of transportation, industrial automation, and renewable energy systems has intensified the demand for high-performance electric drives [1, 2]. Modern applications require precise torque regulation, fast dynamic response, and reliable operation over extended speed ranges. Among synchronous motor technologies, the Interior Permanent Magnet Synchronous Motor (IPMSM) has become a preferred solution due to its high torque density, efficiency, and capability to operate effectively in the flux-weakening region [2, 3].

Despite these advantages, torque control of synchronous motors remains a challenging task. Magnetic saturation, cross-coupling between d- and q-axis dynamics, and parameter sensitivity introduce nonlinear effects that may degrade performance when conventional control strategies are applied outside nominal operating conditions [4, 5]. These phenomena become particularly relevant in high-performance applications where wide-speed operation and fast torque transients are required.

Field-Oriented Control (FOC) and Direct Torque Control (DTC) are among the most widely adopted strategies in industrial practice [6, 7]. FOC achieves satisfactory steady-state performance and provides a systematic framework for decoupled current control, but it relies strongly on accurate machine parameters and may require gain scheduling or compensation strategies when magnetic saturation is significant. DTC, on the other hand, provides rapid torque dynamics, but may suffer from torque ripple and sensitivity to inverter voltage constraints [7]. More recent approaches, such as Direct Flux Vector Control (DFVC), improve torque regulation by explicitly controlling the stator flux vector, thereby offering a more direct link between electromagnetic state variables and torque production [8].

Flux Polar Control (FPC) represents a different perspective on torque regulation. Instead of directly controlling current components, FPC regulates the stator flux vector in polar coordinates through its magnitude and flux angle [9, 10]. In this formulation, torque production becomes a geometric consequence of the orientation of the stator flux with respect to the rotor reference frame. This polar representation provides intuitive physical insight and enables a structured separation between flux regulation and torque dynamics.

The practical implementation of FPC requires two fundamental components. First, the stator flux vector must be reconstructed from measurable electrical quantities in a physically consistent manner. Second, the electromagnetic behavior of the machine must be represented with sufficient accuracy to evaluate the interaction between flux orientation and torque production. To address these requirements, this thesis adopts a progressive modeling approach.

The study begins with a linear dq representation of the IPMSM. In this model, the d- and

q-axis inductances are assumed constant and the permanent-magnet flux linkage is explicitly included [4, 2]. This analytical framework provides a controlled environment for developing and tuning the Flux Polar Control structure. The stator flux components are reconstructed from the stator voltage equation using a voltage-model-based estimator, ensuring consistency between electrical excitation and flux evolution [11].

After validating the control strategy within the linear framework, the analysis is extended to a nonlinear machine representation derived from experimentally provided flux maps. In this case, the electromagnetic behavior is described through lookup tables representing flux linkages and their Jacobian inductances, thereby capturing magnetic saturation and cross-coupling effects [12, 5]. The same FPC architecture developed for the linear model is then applied, without structural modification, to a nonlinear synchronous reluctance motor representation, enabling direct assessment of robustness under more realistic magnetic conditions.

This progression from analytical modeling to nonlinear map-based implementation allows the intrinsic properties of Flux Polar Control to be examined independently of idealized parameter assumptions, while also highlighting the practical limitations introduced by nonlinear magnetic behavior and operating constraints.

The remainder of this thesis is organized as follows. Chapter 2 reviews the main torque-control strategies for synchronous machines and positions Flux Polar Control within the existing literature. Chapter 3 presents the mathematical modeling of the synchronous motor in the dq reference frame and introduces the electromagnetic torque formulation in both linear and nonlinear contexts. Chapter 4 describes the stator flux estimation method and details the Flux Polar Control structure together with the associated reference-generation strategy. Chapter 5 presents the implementation of the proposed drive systems and the adopted simulation setup. Chapter 6 reports and discusses the obtained simulation results. Finally, Chapter 7 summarizes the main conclusions of the work.

Chapter 2

Literature Review

Torque control of AC machines has undergone continuous refinement as performance requirements have increased. In modern electric drives, high bandwidth torque response, operation in the flux-weakening region, and robustness to parameter variations are expected rather than exceptional [1, 2]. Various control structures have been proposed to meet these demands, each emphasizing different trade-offs between model dependence, implementation complexity, and dynamic behavior.

This chapter reviews the principal torque control approaches for synchronous machines and positions Flux Polar Control within this context.

2.1 Current Vector Control

Current Vector Control (CVC), commonly referred to as Field-Oriented Control (FOC), is the most established method for synchronous motor drives [6]. By transforming stator currents into a rotating dq reference frame aligned with rotor flux, CVC achieves decoupled control of flux-producing and torque-producing current components.

This structure enables linear controller design and predictable steady-state performance. However, its accuracy depends directly on knowledge of machine parameters. In IPMSMs, inductances vary with operating point due to magnetic saturation and cross-coupling, and compensation mechanisms or gain scheduling may therefore be required in high-performance applications [4, 5]. In practice, lookup tables are often introduced to preserve torque linearity and to manage operation in the flux-weakening region, increasing implementation complexity.

2.2 Direct Torque Control

Direct Torque Control (DTC) was introduced to improve torque dynamics by eliminating inner current loops [7]. In this approach, torque and stator flux are regulated directly through hysteresis comparators and inverter switching logic, leading to very fast transient response.

While conceptually simple and dynamically effective, classical DTC is characterized by variable switching frequency and noticeable torque ripple. PWM-based implementations improve switching regularity, but sensitivity to low-speed operation and magnetic nonlinearities remains a limitation [7, 1].

2.3 Direct Flux Vector Control

Direct Flux Vector Control (DFVC) represents an intermediate formulation in which the stator flux vector is explicitly controlled using a structured flux representation [8]. Compared with DTC, DFVC improves smoothness and provides a more systematic regulation of electromagnetic variables while preserving the physical interpretation associated with direct flux control.

Nevertheless, its performance remains influenced by variations in differential inductances. Under magnetic saturation, inductance changes complicate controller tuning, and additional logic is typically required to manage voltage constraints in the flux-weakening region [8, 5].

2.4 Flux Polar Control

Flux Polar Control (FPC) adopts a polar representation of the stator flux vector, using the flux magnitude λ and the flux angle γ as control variables [9, 10]. In this formulation, electromagnetic torque is governed by the orientation of the stator flux with respect to the rotor reference frame.

A notable feature of FPC is that the dynamics of the flux-magnitude and flux-angle loops are only weakly dependent on inductance variations. This property simplifies controller tuning and allows fixed gains to be maintained over a wide operating range [9]. In addition, optimization strategies such as Maximum Torque Per Ampere (MTPA) and flux weakening can be included within the reference generation stage without modifying the inner control structure [13, 3].

Because the formulation is based on flux orientation rather than direct current regulation, the same control architecture can be extended to different synchronous motor types with limited structural modification. This feature makes FPC particularly attractive for comparative validation between simplified analytical models and more realistic nonlinear machine representations.

2.5 Machine Representation in Torque Control

The effectiveness of a torque control strategy depends not only on its control structure but also on the fidelity of the machine model used for design and validation.

Linear models assume constant inductances and neglect magnetic saturation. They provide analytical transparency and are therefore well suited for controller development, preliminary tuning, and theoretical interpretation [4, 2].

Nonlinear models, by contrast, describe electromagnetic behavior through flux-linkage maps and differential inductances. These representations capture saturation and cross-coupling effects and are necessary when evaluating drive performance over a wide operating range [12, 5].

In this thesis, both modeling levels are employed. A linear IPMSM model is first used to develop and validate the Flux Polar Control structure under simplified assumptions. The same control architecture is then applied to a nonlinear map-based synchronous reluctance motor model in order to assess its robustness under more realistic magnetic conditions.

The following chapter presents the mathematical modeling framework adopted in this work.

Chapter 3

Machine Modelling

An accurate mathematical representation of the electrical machine is essential for the implementation and evaluation of the Flux Polar Control (FPC) scheme. In this work, two different machine models are considered.

The first is a *linear parameter model* of an Interior Permanent Magnet Synchronous Motor (IPMSM), used as an initial framework for controller development and validation. The second is a *nonlinear map-based model*, used to represent the machine under more realistic magnetic conditions by accounting for nonlinear flux–current relationships.

This chapter presents the mathematical formulation of both models and the mechanical dynamics adopted in the simulations.

3.1 Reference Frame and Definitions

The electrical dynamics are expressed in the synchronous rotating dq reference frame, with the d-axis aligned with the rotor magnetic axis [4, 2].

The electrical rotor angle is defined as

$$\theta_e = p\,\theta_m, \tag{3.1}$$

where p is the number of pole pairs and θ_m is the mechanical rotor position.

The corresponding electrical angular speed is

$$\omega_e = \frac{d\theta_e}{dt} = p\,\omega_m, \tag{3.2}$$

where ω_m is the mechanical angular speed.

Throughout this chapter, R_s denotes the stator resistance, while L_d and L_q denote the direct- and quadrature-axis inductances, respectively.

3.2 Linear Parameter Model

The first machine representation considered in this work is a linear parameter model of an Interior Permanent Magnet Synchronous Motor (IPMSM). This model assumes linear magnetic behaviour and constant machine parameters [4, 2].

In particular, the following assumptions are made:

- the inductances L_d and L_q are constant;
- magnetic saturation and cross-saturation effects are neglected;
- the stator flux linkages depend linearly on the stator currents.

Under these assumptions, the stator flux linkages are expressed as

$$\lambda_d = L_d i_d + \lambda_{PM}, \qquad \lambda_q = L_q i_q, \tag{3.3}$$

where λ_{PM} is the permanent-magnet flux linkage.

3.2.1 Voltage Equations

The stator voltage equations in the synchronous dq reference frame are

$$\begin{aligned} v_d &= R_s i_d + \frac{d\lambda_d}{dt} - \omega_e \lambda_q, \\ v_q &= R_s i_q + \frac{d\lambda_q}{dt} + \omega_e \lambda_d. \end{aligned} \tag{3.4}$$

By substituting the linear flux expressions into the above equations, the current dynamics become

$$\begin{aligned} \frac{di_d}{dt} &= \frac{1}{L_d}\left(v_d - R_s i_d + \omega_e L_q i_q\right), \\ \frac{di_q}{dt} &= \frac{1}{L_q}\left(v_q - R_s i_q - \omega_e (L_d i_d + \lambda_{PM})\right). \end{aligned} \tag{3.5}$$

These equations define the continuous-time electrical dynamics of the linear IPMSM model.

3.2.2 Electromagnetic Torque

For the linear IPMSM model, the electromagnetic torque is given by

$$T_e = \frac{3}{2} p \left(\lambda_{PM} i_q + (L_d - L_q) i_d i_q\right). \tag{3.6}$$

The first term represents the permanent-magnet torque contribution, while the second term corresponds to the reluctance torque contribution due to rotor saliency [2, 4].

3.3 Nonlinear Map-Based Model

To represent the machine under more realistic magnetic operating conditions, a nonlinear map-based model is also considered. In this case, the machine is described directly through nonlinear flux-linkage maps and a torque map.

Unlike the linear model, this representation accounts for magnetic nonlinearities and cross-coupling effects between the d- and q-axes.

The state variables of the model are chosen as the stator currents:

$$x = \begin{bmatrix} i_d \\ i_q \end{bmatrix}. \tag{3.7}$$

3.3.1 Flux-Linkage Maps

The stator flux linkages are represented as nonlinear functions of the stator currents:

$$\lambda_d = \lambda_d(i_d, i_q), \qquad \lambda_q = \lambda_q(i_d, i_q). \tag{3.8}$$

These relationships are implemented as two-dimensional lookup tables derived from the available machine magnetic maps.

3.3.2 Voltage Equations in Flux Form

Starting from the synchronous dq stator voltage equations,

$$\begin{aligned} v_d &= R_s i_d + \dot{\lambda}_d - \omega_e \lambda_q, \\ v_q &= R_s i_q + \dot{\lambda}_q + \omega_e \lambda_d, \end{aligned} \tag{3.9}$$

the flux derivatives can be written as

$$\dot{\lambda}_d = v_d - R_s i_d + \omega_e \lambda_q, \tag{3.10}$$

$$\dot{\lambda}_q = v_q - R_s i_q - \omega_e \lambda_d. \tag{3.11}$$

Defining the flux-derivative vector as

$$\dot{\boldsymbol{\lambda}} = \begin{bmatrix} \dot{\lambda}_d \\ \dot{\lambda}_q \end{bmatrix}, \tag{3.12}$$

these equations describe the required evolution of the stator flux components.

3.3.3 Differential Inductance Matrix

The nonlinear relationship between flux and current is locally described by the Jacobian matrix of the flux maps:

$$J = \begin{bmatrix} L_{dd} & L_{dq} \\ L_{qd} & L_{qq} \end{bmatrix} = \begin{bmatrix} \frac{\partial \lambda_d}{\partial i_d} & \frac{\partial \lambda_d}{\partial i_q} \\ \frac{\partial \lambda_q}{\partial i_d} & \frac{\partial \lambda_q}{\partial i_q} \end{bmatrix}. \tag{3.13}$$

The entries of this matrix are obtained offline from the flux-linkage maps and implemented as dedicated lookup tables.

The determinant of the Jacobian is

$$\Delta = L_{dd} L_{qq} - L_{dq} L_{qd}. \tag{3.14}$$

Assuming $\Delta \neq 0$, the inverse Jacobian is given by

$$J^{-1} = \frac{1}{\Delta} \begin{bmatrix} L_{qq} & -L_{dq} \\ -L_{qd} & L_{dd} \end{bmatrix}. \tag{3.15}$$

3.3.4 Current Dynamics

The relationship between flux derivatives and current derivatives is

$$\dot{\boldsymbol{\lambda}} = J\, \dot{\mathbf{i}}, \tag{3.16}$$

where

$$\dot{\mathbf{i}} = \begin{bmatrix} \dot{i}_d \\ \dot{i}_q \end{bmatrix}. \tag{3.17}$$

Solving for the current derivatives yields

$$\dot{\mathbf{i}} = J^{-1}\dot{\boldsymbol{\lambda}}. \tag{3.18}$$

Explicitly,

$$\begin{bmatrix} \dot{i}_d \\ \dot{i}_q \end{bmatrix} = \frac{1}{\Delta} \begin{bmatrix} L_{qq} & -L_{dq} \\ -L_{qd} & L_{dd} \end{bmatrix} \begin{bmatrix} \dot{\lambda}_d \\ \dot{\lambda}_q \end{bmatrix}. \tag{3.19}$$

These derivatives are integrated to obtain the stator current trajectories:

$$i_d(t) = \int \dot{i}_d \, dt, \qquad i_q(t) = \int \dot{i}_q \, dt. \tag{3.20}$$

3.3.5 Electromagnetic Torque

In the nonlinear model, the electromagnetic torque is obtained directly from a two-dimensional torque lookup table:

$$T_e = T_e(i_d, i_q). \tag{3.21}$$

This implementation preserves the nonlinear electromagnetic behavior captured in the machine maps and avoids relying on simplified analytical torque expressions.

In the nonlinear implementation adopted in this work, the flux maps correspond to a synchronous reluctance motor. Therefore, no permanent-magnet flux linkage term appears in the model, although the mathematical structure of the voltage equations remains unchanged.

3.4 Mechanical Dynamics

In addition to the electrical dynamics, the rotor mechanical behaviour is described through the standard equation of motion [4, 1]:

$$J\frac{d\omega_m}{dt} = T_e - T_L - B\omega_m, \tag{3.22}$$

where:

- J is the total moment of inertia,
- B is the viscous friction coefficient,
- T_e is the electromagnetic torque,
- T_L is the load torque.

The rotor position evolves according to

$$\frac{d\theta_m}{dt} = \omega_m. \tag{3.23}$$

These equations are used to describe the electromechanical coupling of the drive and are particularly relevant when speed-control operation is considered.

3.5 Summary

This chapter presented the machine models adopted in this work. A linear parameter IPMSM model was introduced as a simplified framework for controller development, while a nonlinear map-based model was used to capture more realistic magnetic behaviour through flux-linkage and torque maps. Finally, the rotor mechanical dynamics were included to describe the electromechanical interaction between the machine and the load.

These models provide the basis for the implementation of the Flux Polar Control strategy presented in the next chapter.

Chapter 4

Flux Polar Control

Flux Polar Control (FPC) regulates the stator flux vector in polar coordinates. Instead of directly controlling the stator current components, the control variables are the flux magnitude λ and the flux angle γ with respect to the rotor d-axis [9, 10]. Torque production is therefore governed by the geometric orientation of the stator flux vector.

The same FPC architecture is adopted for both the linear IPMSM model and the nonlinear map-based model. However, the realization of reference generation and flux estimation differs according to the machine representation. The following sections describe the control structure following the actual signal flow of the implementation.

4.1 Polar Representation of the Stator Flux

In the synchronous dq reference frame, the stator flux vector is written as

$$\boldsymbol{\lambda}_{dq} = \begin{bmatrix} \lambda_d \\ \lambda_q \end{bmatrix}. \tag{4.1}$$

The flux magnitude and flux angle are defined as

$$\lambda = \sqrt{\lambda_d^2 + \lambda_q^2}, \qquad \gamma = \operatorname{atan2}(\lambda_q, \lambda_d). \tag{4.2}$$

The angle γ represents the orientation of the stator flux vector relative to the rotor d-axis.

4.2 Polar Control Inputs

Following [9, 10], the FPC scheme introduces two polar control inputs:

$$u_\lambda = \frac{d\lambda}{dt}, \qquad u_\gamma = \frac{d\gamma}{dt}. \tag{4.3}$$

The input u_λ governs the stator flux magnitude dynamics, while u_γ governs the flux-angle dynamics.

4.3 Reference Generation

4.3.1 Linear IPMSM Case

For the linear IPMSM model, reference generation is based on analytical relationships between torque, flux, and current [13, 3].

The MTPA operating locus is computed offline and implemented through a one-dimensional lookup table:

$$\lambda^*_{\mathrm{MTPA}} = \mathrm{LUT}_\lambda(T^*), \tag{4.4}$$

where T^* is the imposed torque reference.

To enforce the voltage constraint at increasing speed, a flux-weakening limiter is introduced:

$$\lambda_{\max}(\omega_e) = \frac{v_{\max}}{|\omega_e| + \varepsilon}, \tag{4.5}$$

where $v_{\max}$ is the maximum available stator voltage and $\varepsilon > 0$ is a small positive constant introduced to avoid division by zero.

The final flux reference is therefore obtained as

$$\lambda^* = \min(\lambda^*_{\mathrm{MTPA}},\, \lambda_{\max}). \tag{4.6}$$

The corresponding flux-angle reference is obtained from a two-dimensional lookup table:

$$\gamma^* = \mathrm{LUT}_\gamma(T^*, \lambda^*). \tag{4.7}$$

For consistency with the polar control variables, the torque associated with a given reference pair (λ^*, γ^*) can be written as

$$T^* = \frac{3}{2} p \left[\frac{\lambda_{PM}}{L_d} \lambda^* \sin\gamma^* + \left(\frac{1}{L_q} - \frac{1}{L_d} \right) \lambda^{*2} \sin\gamma^* \cos\gamma^* \right]. \tag{4.8}$$

4.3.2 Nonlinear Map-Based Case

In the nonlinear implementation, reference generation is performed in the current domain.

The torque reference T^* is generated by the outer speed-control loop and then supplied to the reference-generation block. Based on the operating condition and voltage feasibility constraints, this block combines the MTPA and flux-weakening operating regions and produces the reference current components

$$i^*_d, \qquad i^*_q. \tag{4.9}$$

These reference currents are then applied to the nonlinear flux maps:

$$\lambda^*_d = \lambda_d(i^*_d, i^*_q), \qquad \lambda^*_q = \lambda_q(i^*_d, i^*_q), \tag{4.10}$$

implemented as two-dimensional lookup tables.

The corresponding flux magnitude and flux angle references are then computed as

$$\lambda^* = \sqrt{\lambda^{*2}_d + \lambda^{*2}_q}, \tag{4.11}$$

$$\gamma^* = \mathrm{atan2}(\lambda^*_q, \lambda^*_d). \tag{4.12}$$

In this case, the flux references are obtained directly from the nonlinear maps, without relying on analytical expressions in terms of (λ, γ).

4.4 Flux Estimation

4.4.1 Linear Model: $\alpha\beta$ Flux Observer with Magnetic Model Correction

In the linear IPMSM implementation, the stator flux is reconstructed using a voltage-model-based observer formulated in the stationary $\alpha\beta$ reference frame, including a magnetic-model correction term as described in [11].

Voltage Model The measured phase voltages and currents are first transformed to the $\alpha\beta$ frame. The voltage-model term is computed as

$$e_{\alpha\beta} = v_{\alpha\beta} - R_s i_{\alpha\beta}. \tag{4.13}$$

Magnetic Model A parallel magnetic-model branch evaluates the stator flux from the machine equations. The currents are transformed to the dq frame using the electrical angle θ_e,

$$i_{\alpha\beta} \rightarrow i_d, i_q, \tag{4.14}$$

and the flux components are computed as

$$\lambda_d^{model} = L_d i_d + \lambda_{PM}, \qquad \lambda_q^{model} = L_q i_q. \tag{4.15}$$

These components are then transformed back to the $\alpha\beta$ frame to obtain $\lambda_{\alpha\beta}^{model}$.

Observer Correction The estimation error is defined as

$$\tilde{\lambda}_{\alpha\beta} = \hat{\lambda}_{\alpha\beta} - \lambda_{\alpha\beta}^{model}. \tag{4.16}$$

The observer dynamics are given by

$$\dot{\hat{\lambda}}_{\alpha\beta} = e_{\alpha\beta} - g\,\tilde{\lambda}_{\alpha\beta}, \tag{4.17}$$

where g is the observer correction gain.

Flux Integration The estimated flux components are obtained through integration:

$$\hat{\lambda}_\alpha = \int \dot{\hat{\lambda}}_\alpha \, dt, \qquad \hat{\lambda}_\beta = \int \dot{\hat{\lambda}}_\beta \, dt. \tag{4.18}$$

Transformation to dq Frame The estimated stationary-frame flux is transformed to the rotating frame,

$$\hat{\lambda}_{\alpha\beta} \rightarrow \hat{\lambda}_d, \hat{\lambda}_q, \tag{4.19}$$

from which the quantities used by the FPC controller are computed:

$$\hat{\lambda} = \sqrt{\hat{\lambda}_d^2 + \hat{\lambda}_q^2}, \qquad \hat{\gamma} = \operatorname{atan2}(\hat{\lambda}_q, \hat{\lambda}_d). \tag{4.20}$$

This closed-loop structure mitigates integration drift and improves robustness with respect to parameter uncertainties in the voltage model.

4.4.2 Nonlinear Model: Direct Flux Computation

In the nonlinear implementation, the stator flux components are obtained directly from the nonlinear flux maps:

$$\lambda_d = \lambda_d(i_d, i_q), \qquad \lambda_q = \lambda_q(i_d, i_q). \tag{4.21}$$

The flux magnitude and flux angle are therefore computed directly as

$$\lambda = \sqrt{\lambda_d^2 + \lambda_q^2}, \qquad \gamma = \operatorname{atan2}(\lambda_q, \lambda_d). \tag{4.22}$$

No observer is required in this case, since the flux components are directly available from the implemented magnetic maps.

4.5 Speed Control Loop

In addition to the inner Flux Polar Control loops, a speed-control loop is introduced in the nonlinear implementation to regulate the rotor speed.

The mechanical dynamics of the machine are described by

$$J\frac{d\omega_m}{dt} = T_e - T_L - B\omega_m, \tag{4.23}$$

where J is the inertia, B is the viscous friction coefficient, T_e is the electromagnetic torque, and T_L is the load torque.

A PI controller is used to generate the torque reference from the speed error:

$$T^* = K_p^\omega(\omega_m^* - \omega_m) + K_i^\omega \int (\omega_m^* - \omega_m)\, dt, \tag{4.24}$$

where ω_m^* is the reference speed and ω_m is the measured speed.

To improve robustness under torque saturation, the speed PI controller is implemented with anti-windup action.

The generated torque reference is then supplied to the reference-generation block, which determines the corresponding current references according to the selected operating region. These are then converted into the flux-magnitude and flux-angle references used by the inner FPC loops.

4.6 Polar Control Loops

The polar control inputs are generated through two PI regulators acting on the flux-magnitude and flux-angle errors:

$$u_\lambda = K_p^\lambda(\lambda^* - \hat{\lambda}) + K_i^\lambda \int (\lambda^* - \hat{\lambda})\, dt, \tag{4.25}$$

$$u_\gamma = K_p^\gamma(\gamma^* - \hat{\gamma}) + K_i^\gamma \int (\gamma^* - \hat{\gamma})\, dt. \tag{4.26}$$

The estimated quantities $\hat{\lambda}$ and $\hat{\gamma}$ are obtained from the flux estimation block described previously.

In the implemented controller, both PI regulators include anti-windup action through back-calculation in order to prevent integrator windup when the commanded voltage vector is saturated. This is particularly important in flux-weakening operation and under high torque demand, where the voltage constraint becomes active.

4.7 Voltage Synthesis in the dq Frame

The polar inputs are converted into voltage references through

$$\begin{aligned} v_d^* &= R_s i_d + u_\lambda \cos\gamma - \lambda(u_\gamma + \omega_e)\sin\gamma, \\ v_q^* &= R_s i_q + u_\lambda \sin\gamma + \lambda(u_\gamma + \omega_e)\cos\gamma. \end{aligned} \tag{4.27}$$

These equations provide the interface between the polar control structure and the machine model, and correspond to the standard voltage synthesis of the FPC framework [9, 10].

4.8 Voltage Limitation

In this work, no power electronic stage or modulation process is explicitly modeled. Nevertheless, the stator voltage that can be applied to the machine is bounded by the available DC supply and must be respected to ensure physically admissible operation.

This constraint is enforced at the control level by limiting the magnitude of the voltage reference vector:

$$\sqrt{v_d^{*2} + v_q^{*2}} \leq v_{\max}, \tag{4.28}$$

where $v_{\max}$ represents the maximum available stator voltage derived from the DC supply.

If the voltage magnitude exceeds this limit, the voltage references are proportionally scaled to preserve the direction of the control vector. This ensures consistent controller behaviour while preventing unrealizable voltage commands.

4.9 Summary

This chapter presented the Flux Polar Control architecture following the actual signal flow of the implementation. The same polar control structure is employed for both the linear IPMSM and nonlinear map-based models.

The main difference lies in the realization of reference generation and flux estimation. In the linear case, analytical relationships and a voltage-model-based observer are used. In the nonlinear case, both the reference quantities and the flux components are obtained directly from the machine maps.

Chapter 5

Implementation and Simulation Setup

5.1 Introduction

This chapter presents the practical implementation of the drive systems developed in this thesis and the simulation setup adopted for their evaluation. The objective is to describe how the machine models and control structures introduced in the previous chapters were realized in the Simulink environment.

The implementation is presented for both the linear IPMSM model and the nonlinear map-based synchronous reluctance motor model. Particular attention is given to the overall control architecture, the structure of the implemented blocks, and the main simulation parameters used in the test campaigns discussed in the next chapter.

5.2 Overall Drive Architecture

The implemented drive system follows a cascaded electric-drive control structure in which the machine model, the reference-generation block, the Flux Polar Control loops, and the voltage-limitation stage are interconnected in closed loop.

Depending on the considered implementation, the control system receives either an imposed torque reference or a speed reference. The reference-generation block then produces the corresponding flux-magnitude and flux-angle references, which are regulated through the inner Flux Polar Control loops. The resulting voltage commands are finally applied to the machine model.

Figures 5.1 and 5.2 show the top-level Simulink realization adopted for the linear and nonlinear implementations, respectively.

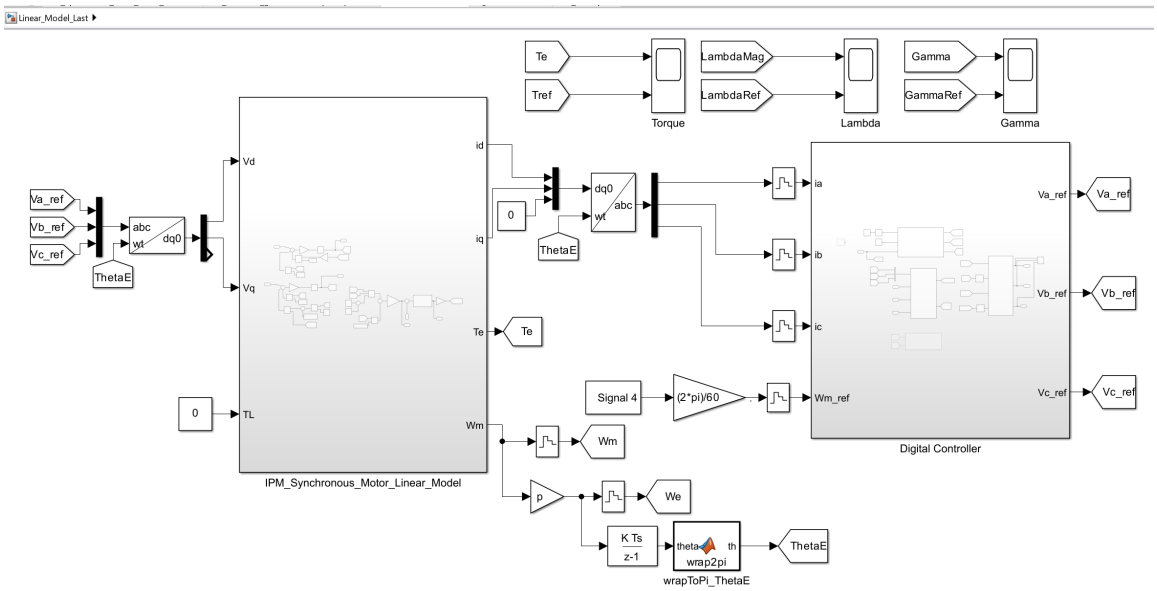


Figure 5.1: Top-level Simulink implementation of the linear IPMSM drive.

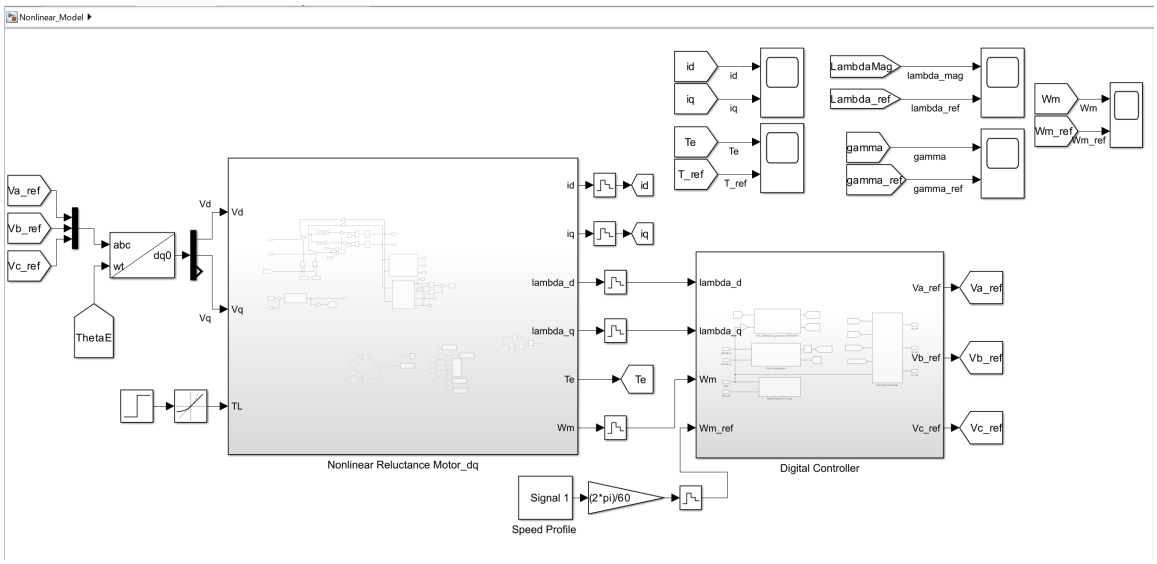
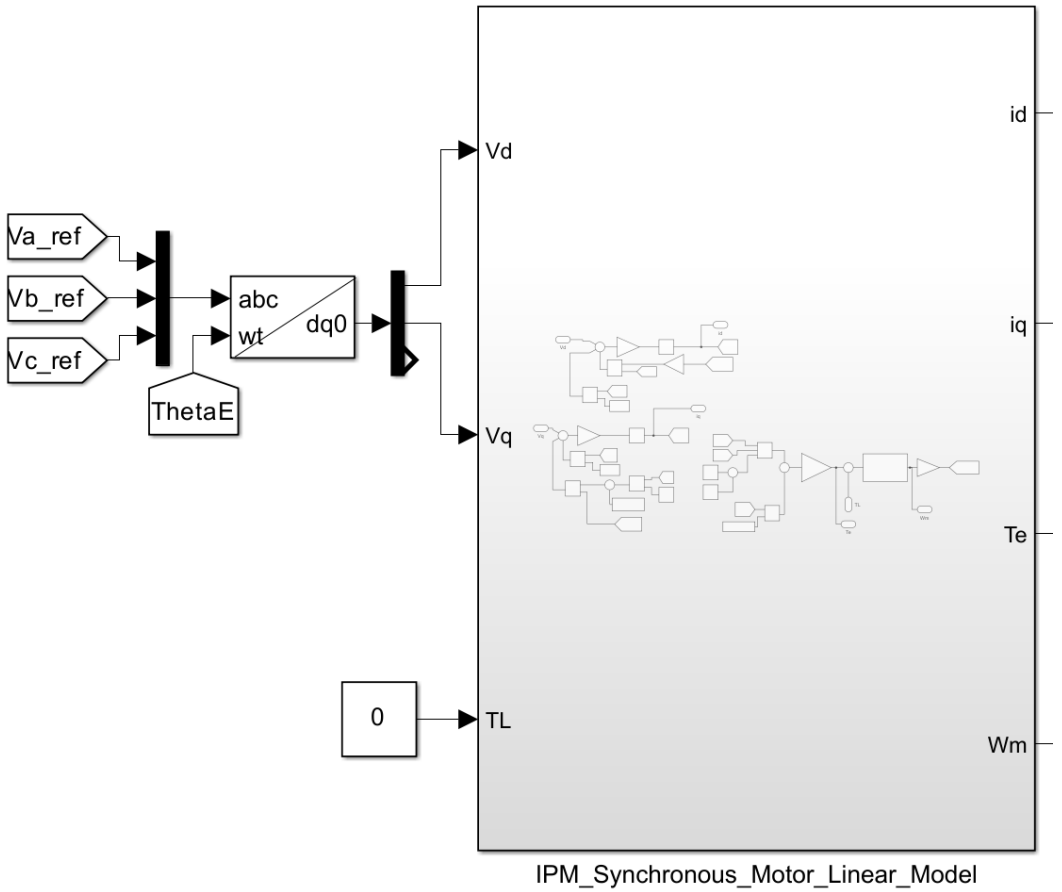


Figure 5.2: Top-level Simulink implementation of the nonlinear map-based drive.

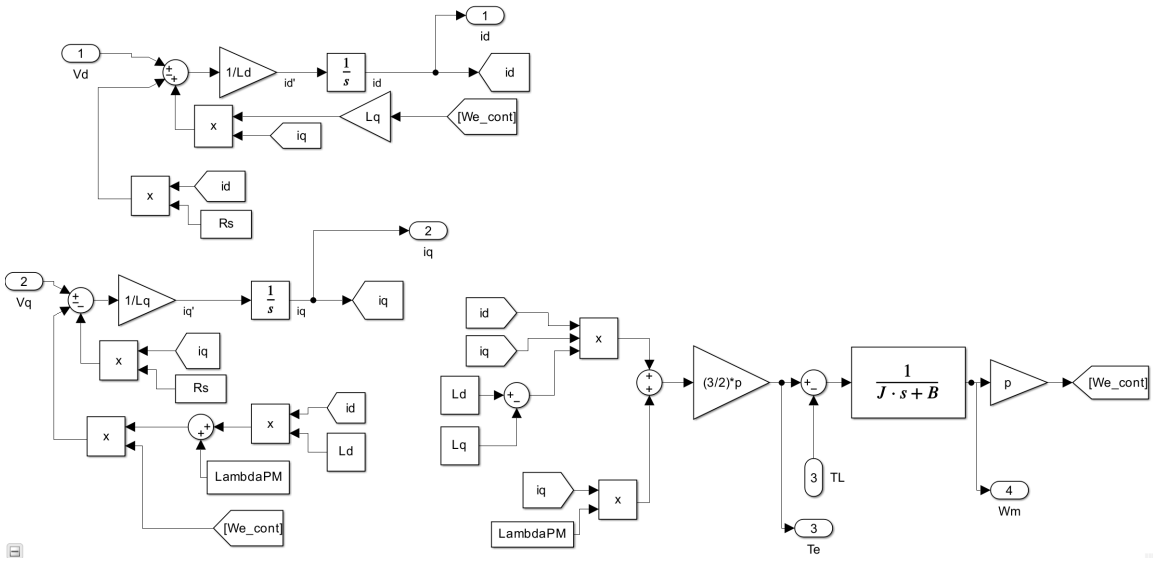
5.3 Implementation of the Linear IPMSM Drive

The linear IPMSM drive was implemented in Simulink using the analytical machine model described in Chapter 3 together with the Flux Polar Control structure presented in Chapter 4. In this implementation, the torque reference is imposed directly in order to evaluate the behavior of the inner FPC loops independently of outer-loop speed control.

5.3.1 Linear IPMSM Model Implementation



(a) Top-level linear IPMSM model block.



(b) Internal implementation of the linear IPMSM model.

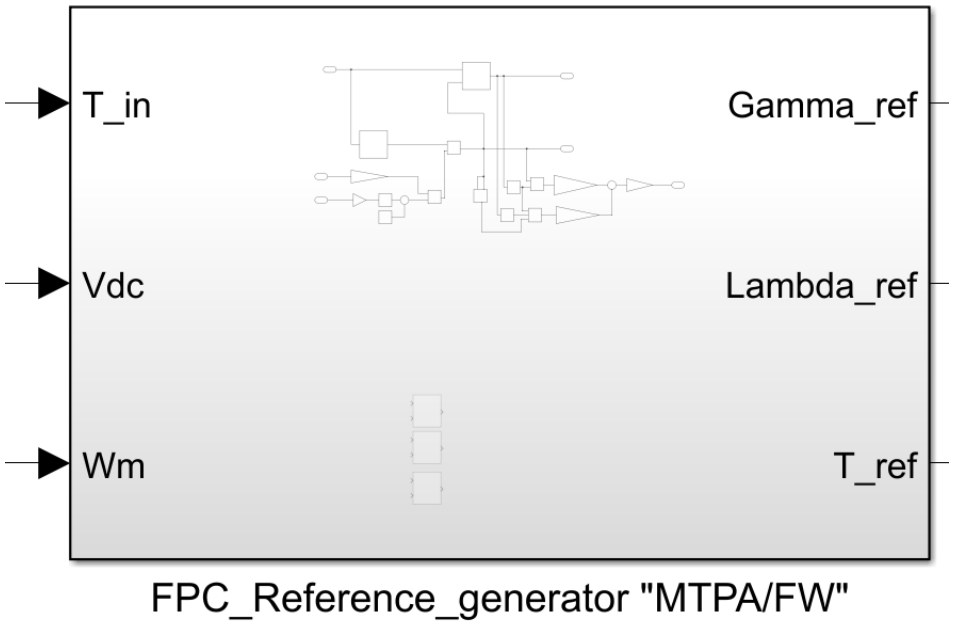
Figure 5.3: Simulink implementation of the linear IPMSM model.

Figure 5.3a shows the top-level implementation of the linear IPMSM model used in the Simulink environment. The three-phase voltage references generated by the controller are first transformed into the rotating dq frame using the electrical rotor position θ_e. The resulting v_d and v_q voltages are then applied to the machine model together with the load torque input T_L.

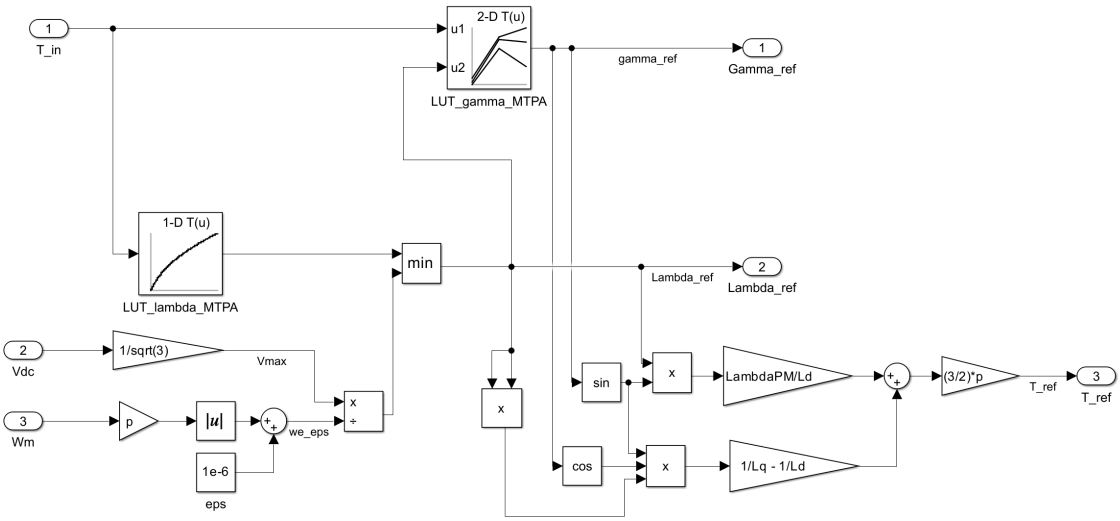
Figure 5.3b shows the internal realization of the machine dynamics. The stator current dynamics are implemented according to the linear dq-model using constant parameters R_s, L_d, L_q, and λ_{PM}. The electromagnetic torque is computed from the resulting currents and coupled with the mechanical dynamics through the inertia J and viscous friction coefficient B, allowing the rotor speed ω_m and electrical speed ω_e to evolve consistently.

The implementation provides the machine outputs i_d, i_q, electromagnetic torque T_e, and mechanical speed ω_m, which are used by the control and estimation blocks.

5.3.2 Reference Generation Block



(a) Top-level reference-generation block.



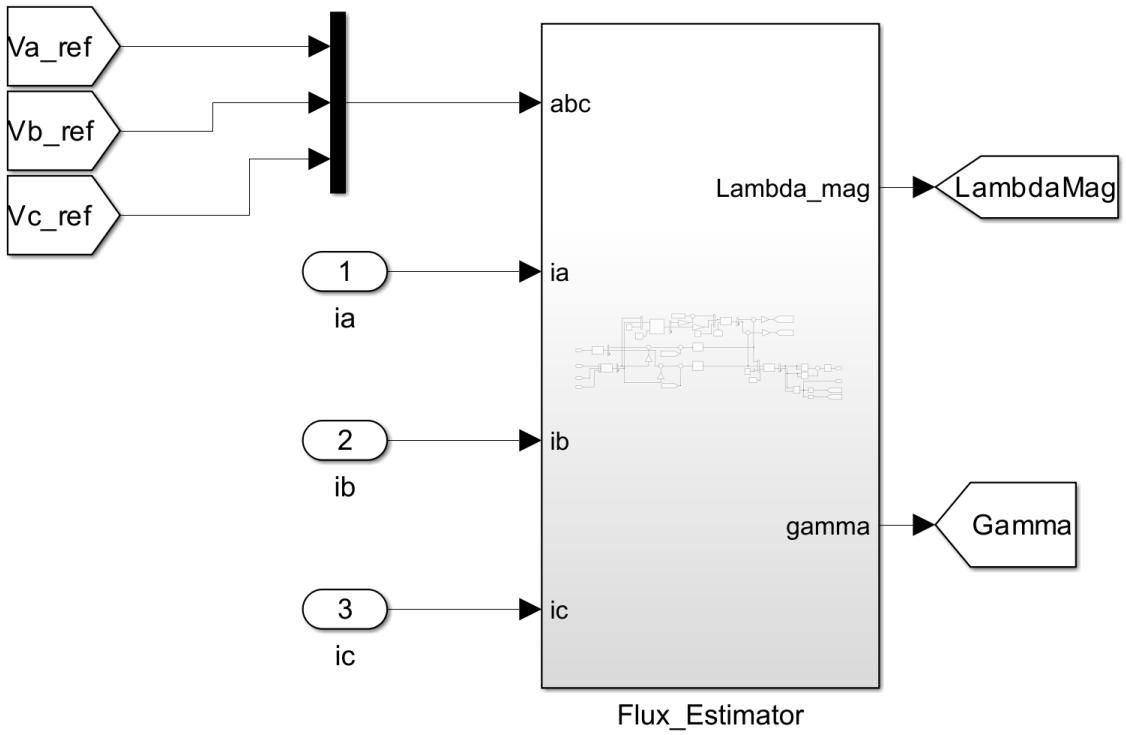
(b) Internal implementation of the linear IPMSM reference generator.

Figure 5.4: Reference-generation structure adopted in the linear implementation.

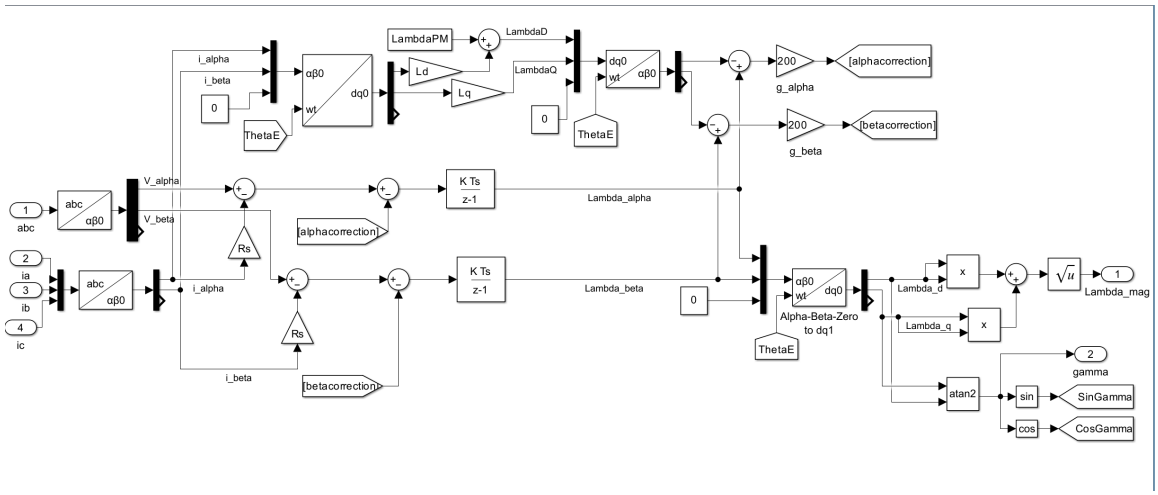
The reference-generation block receives the imposed torque reference, the DC-link voltage, and rotor speed, and produces the flux-magnitude reference λ^*, the flux-angle reference γ^*, and the corresponding torque signal used in the control structure.

In the linear IPMSM implementation, the block combines MTPA lookup tables with a flux-weakening limiter. The flux reference is first obtained from the MTPA locus and then limited according to the available voltage, while the flux-angle reference is generated from a two-dimensional lookup table.

5.3.3 Flux Estimator



(a) Top-level flux-estimation block.



(b) Internal implementation of the linear-model flux estimator.

Figure 5.5: Flux-estimation structure adopted in the linear implementation.

The flux-estimation block receives the applied stator voltages and measured phase currents and provides the estimated flux magnitude $\hat{\lambda}$ and flux angle $\hat{\gamma}$ required by the Flux Polar Controller.

In the linear implementation, the estimator is realized as a voltage-model observer in the stationary $\alpha\beta$ frame with magnetic-model correction. The estimated flux components are then transformed to the rotating dq frame.

5.3.4 Flux Polar Controller

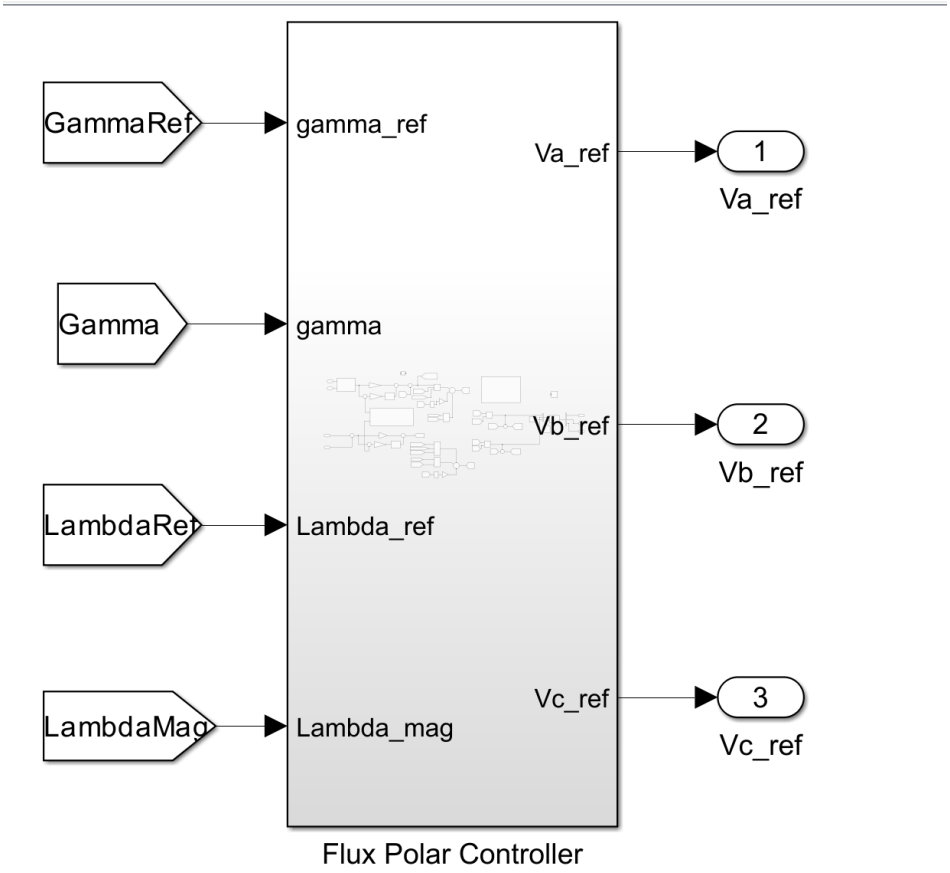


Figure 5.6: Top-level Flux Polar Controller block.

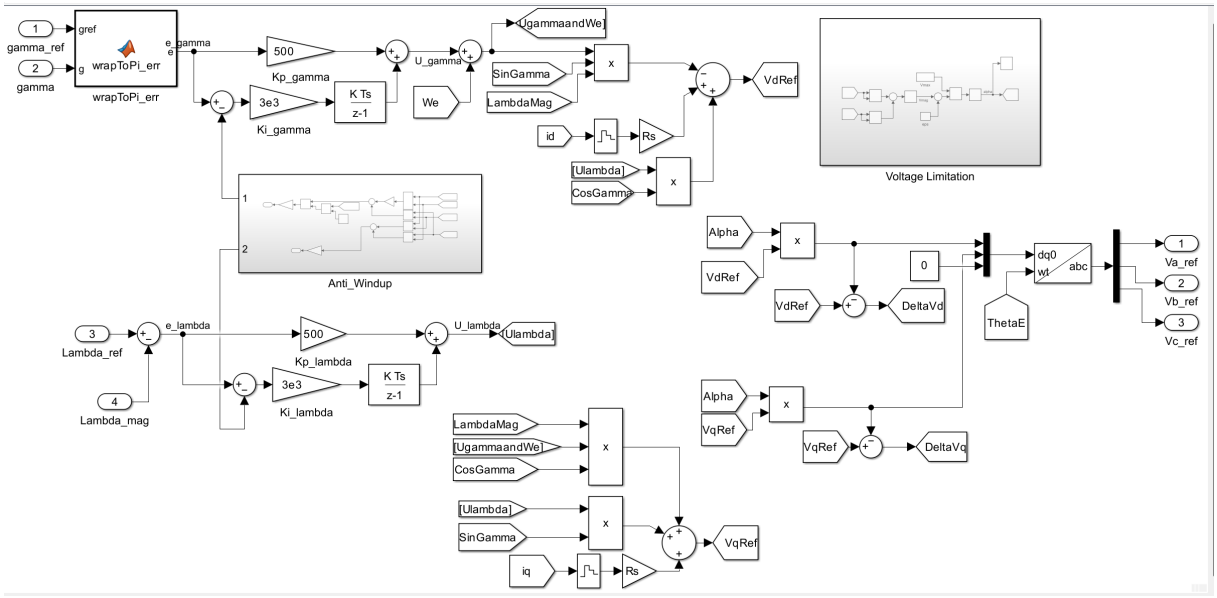


Figure 5.7: Internal implementation of the Flux Polar Controller.

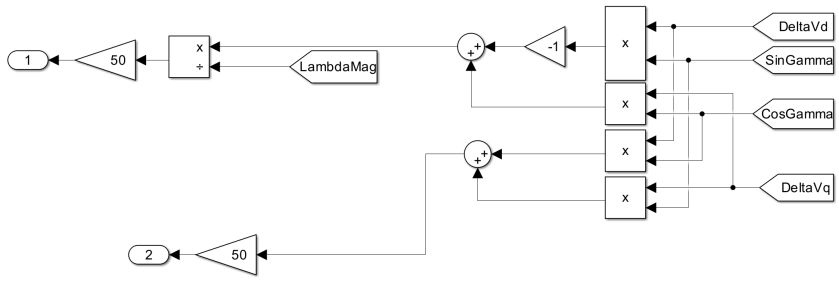


Figure 5.8: Anti-windup compensation adopted in the PI loops.

The Flux Polar Controller receives the reference and estimated values of the flux magnitude and flux angle and generates the corresponding stator voltage references through two PI control loops acting on the flux-magnitude and flux-angle errors.

To improve the behavior in the presence of voltage saturation, a back-calculation anti-windup mechanism is included in both PI regulators.

5.3.5 Voltage Limitation

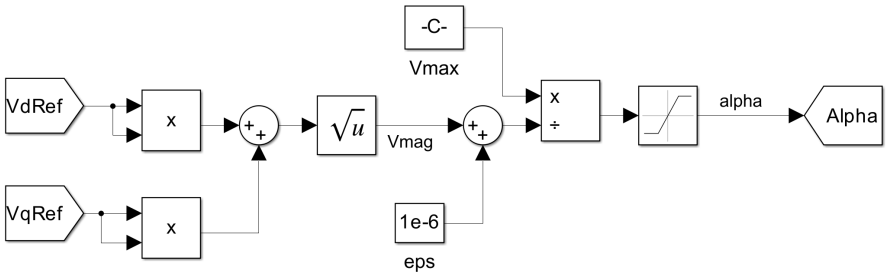


Figure 5.9: Implementation of the voltage-limitation block.

The voltage-limitation block computes the magnitude of the unconstrained voltage reference vector from the dq-frame components v_d^* and v_q^*. A scaling factor α is then generated by comparing the voltage magnitude with the maximum available voltage $V_{\max}$.

When the voltage magnitude exceeds the admissible limit, the voltage references are proportionally scaled by α, preserving the direction of the voltage vector while keeping its magnitude physically realizable.

5.4 Implementation of the Nonlinear Map-Based Drive

The nonlinear implementation was realized using the map-based synchronous reluctance motor model presented in Chapter 3. In this case, the machine behavior is described directly through flux-linkage and torque lookup tables.

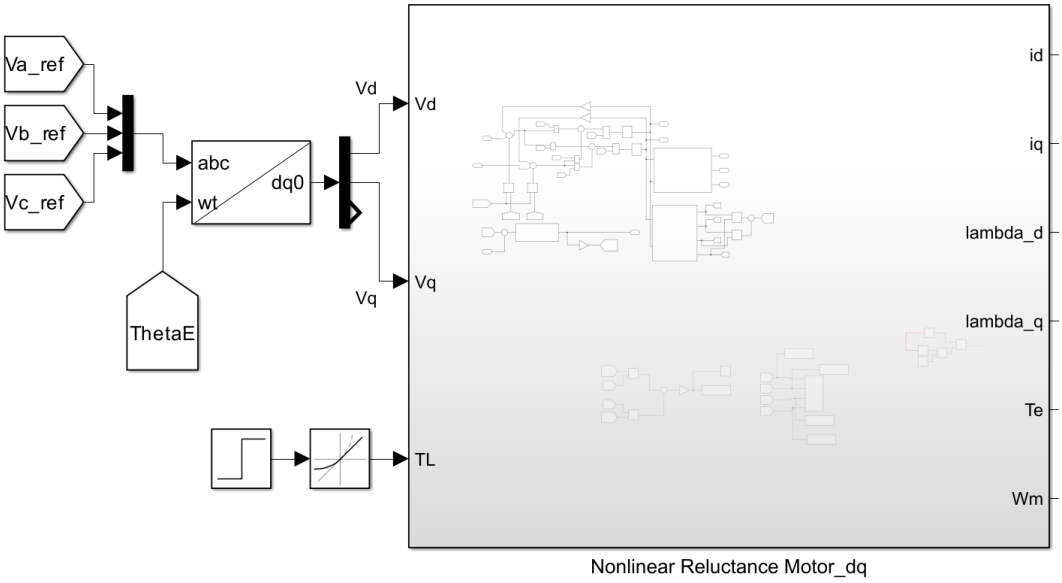
The machine model was implemented using the stator currents as state variables. The flux components and electromagnetic torque were obtained directly from the nonlinear maps, while

the differential inductance matrix was evaluated through dedicated lookup tables used to compute the current derivatives.

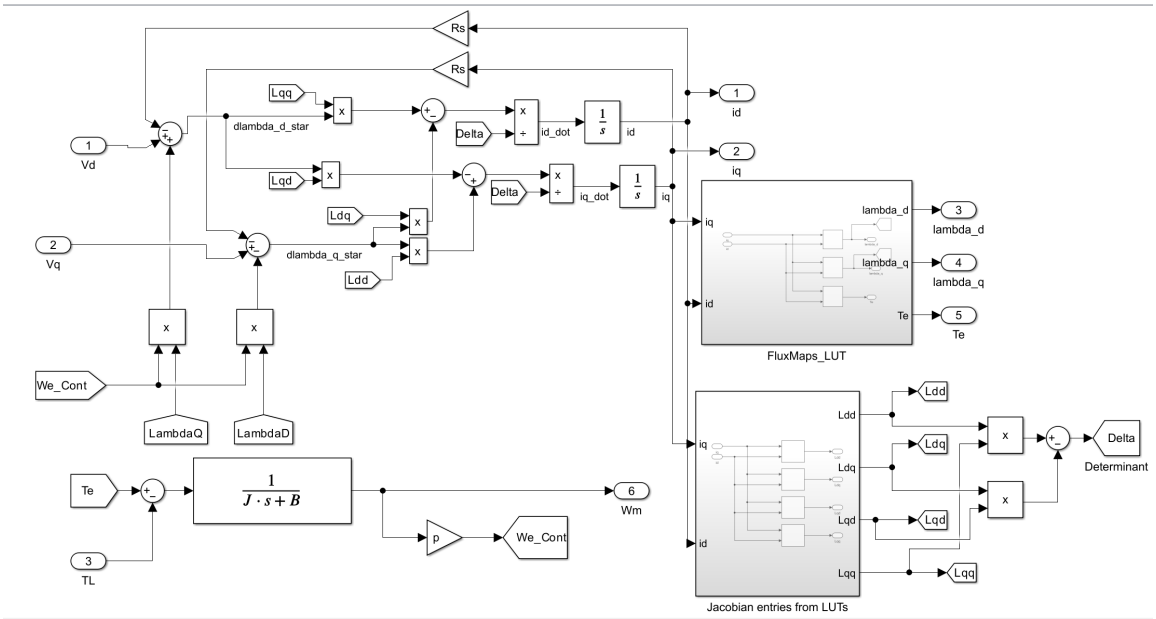
The nonlinear drive also includes an outer speed-control loop, which generates the torque reference supplied to the reference-generation block. The latter determines the corresponding current references for MTPA and flux-weakening operation, from which the flux-magnitude and flux-angle references are computed.

The same inner Flux Polar Control structure used in the linear case was then applied to regulate the flux magnitude and flux angle, while the resulting voltage commands were limited according to the available voltage constraint before being applied to the nonlinear machine model.

5.4.1 Nonlinear Machine Model Implementation



(a) Top-level nonlinear map-based motor block.



(b) Internal implementation of the nonlinear machine model.

Figure 5.10: Simulink implementation of the nonlinear map-based synchronous reluctance motor model.

Figure 5.10a shows the top-level Simulink realization of the nonlinear map-based machine model. The three-phase voltage references generated by the controller are transformed into the rotating dq frame through the electrical angle θ_e, producing the applied stator voltage components v_d and v_q. These voltages, together with the load torque input T_L, are applied to the nonlinear motor block.

Figure 5.10b shows the internal implementation of the nonlinear machine model. The electro-

magnetic behavior is described through lookup tables that provide the flux-linkage components $\lambda_d(i_d, i_q)$, $\lambda_q(i_d, i_q)$, the electromagnetic torque $T_e(i_d, i_q)$, and the entries of the differential inductance matrix.

The stator currents i_d and i_q are treated as the electrical state variables of the model. Their derivatives are computed from the nonlinear voltage equations using the Jacobian terms of the flux maps and then integrated to obtain the current evolution. The electromagnetic torque is coupled with the mechanical dynamics through the inertia J and viscous friction coefficient B, yielding the rotor speed ω_m and the electrical speed ω_e.

The implementation provides the machine outputs i_d, i_q, λ_d, λ_q, electromagnetic torque T_e, and mechanical speed ω_m.

5.4.2 Flux and Torque Map Evaluation

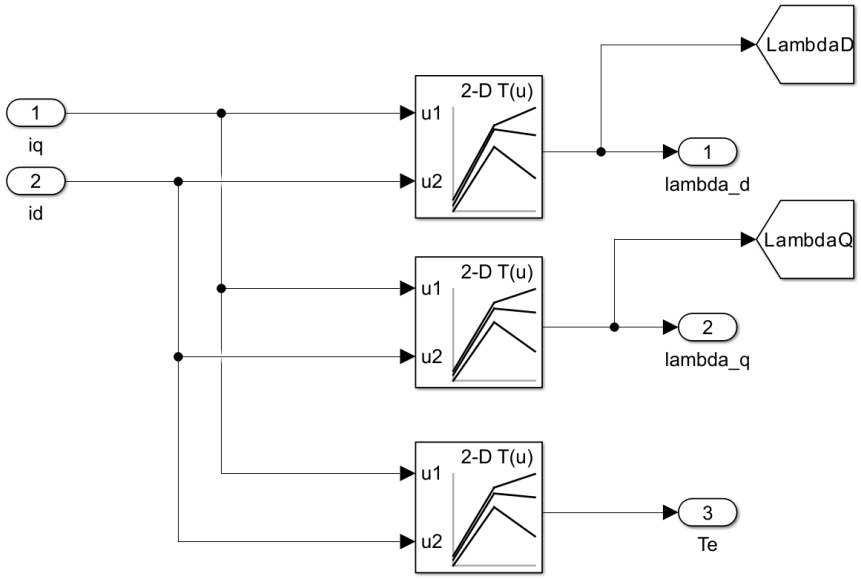


Figure 5.11: Evaluation of flux-linkage and torque maps from the nonlinear lookup tables.

Figure 5.11 shows the lookup-table block used to evaluate the nonlinear electromagnetic quantities of the machine. The stator current components i_d and i_q are used as inputs to three two-dimensional lookup tables that provide the flux-linkage components $\lambda_d(i_d, i_q)$, $\lambda_q(i_d, i_q)$, and the electromagnetic torque $T_e(i_d, i_q)$.

These outputs are used in the electrical dynamics of the nonlinear machine model and in the flux-related quantities required by the control structure.

5.4.3 Differential Inductance Matrix and Current Dynamics

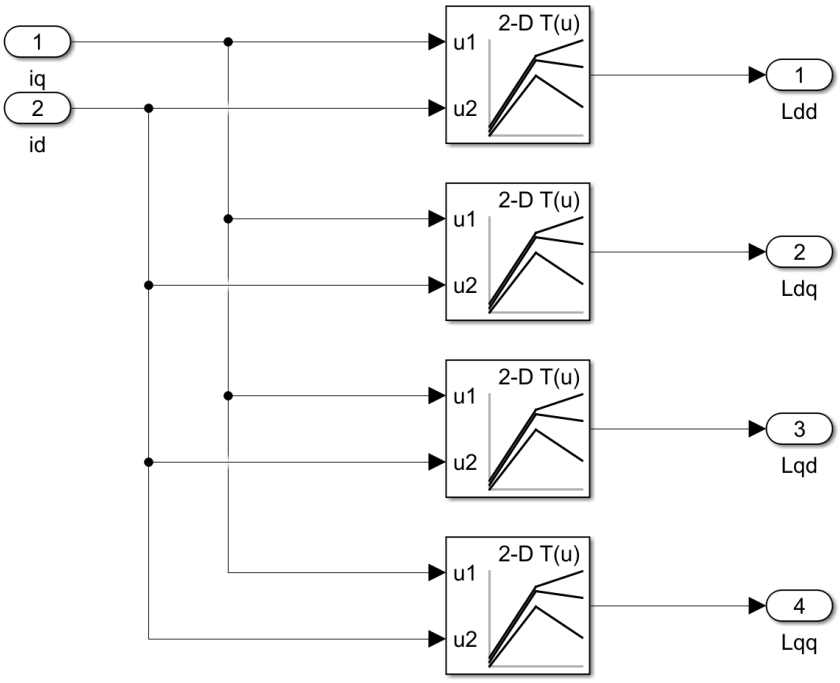


Figure 5.12: Evaluation of the differential inductance matrix entries from lookup tables.

Figure 5.12 shows the evaluation of the differential inductance matrix entries using two-dimensional lookup tables. The stator currents i_d and i_q are used as inputs to obtain the four partial derivatives of the flux-linkage functions:

$$L_{dd} = \frac{\partial \lambda_d}{\partial i_d}, \quad L_{dq} = \frac{\partial \lambda_d}{\partial i_q}, \quad L_{qd} = \frac{\partial \lambda_q}{\partial i_d}, \quad L_{qq} = \frac{\partial \lambda_q}{\partial i_q}.$$

These quantities form the differential inductance matrix

$$\mathbf{L}(i_d, i_q) = \begin{bmatrix} L_{dd} & L_{dq} \\ L_{qd} & L_{qq} \end{bmatrix}.$$

The electrical dynamics are expressed in terms of the stator current derivatives by inverting the differential inductance matrix:

$$\begin{bmatrix} \dot{i}_d \\ \dot{i}_q \end{bmatrix} = \mathbf{L}^{-1}(i_d, i_q) \left(\begin{bmatrix} v_d \\ v_q \end{bmatrix} - \begin{bmatrix} R_s i_d \\ R_s i_q \end{bmatrix} - \begin{bmatrix} -\omega_e \lambda_q \\ \omega_e \lambda_d \end{bmatrix} \right).$$

This formulation enables the current dynamics to reflect the nonlinear magnetic coupling of the machine.

5.4.4 Speed Control Loop

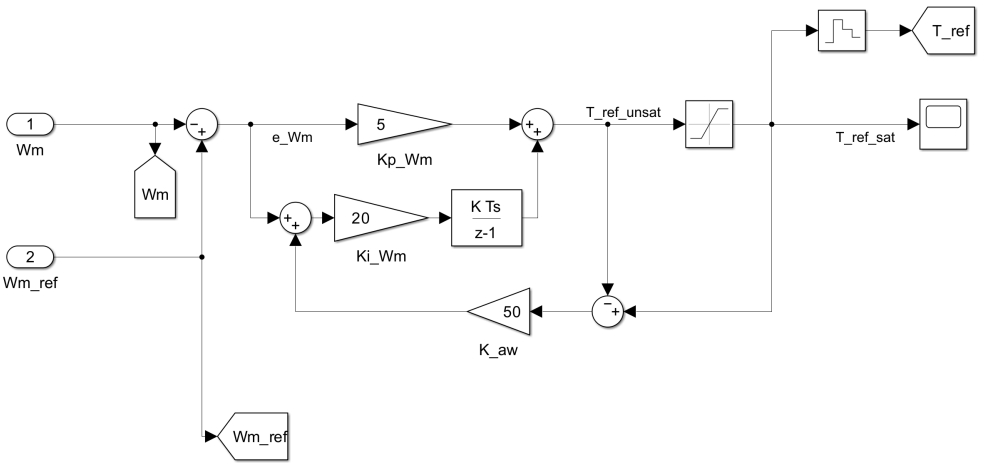


Figure 5.13: Implementation of the outer speed-control loop.

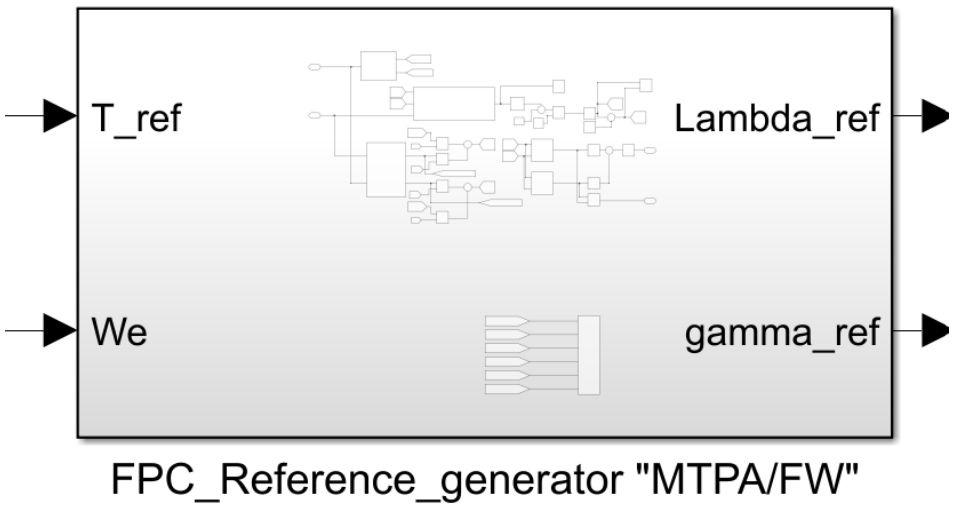
Figure 5.13 shows the outer speed-control loop adopted in the nonlinear map-based drive. In contrast to the linear implementation, where the torque reference is imposed directly, the nonlinear drive includes a speed regulation stage to generate the electromagnetic torque reference required by the inner control structure.

The measured mechanical speed ω_m is compared with the reference speed ω_m^*, producing the speed error used as input to a discrete PI controller.

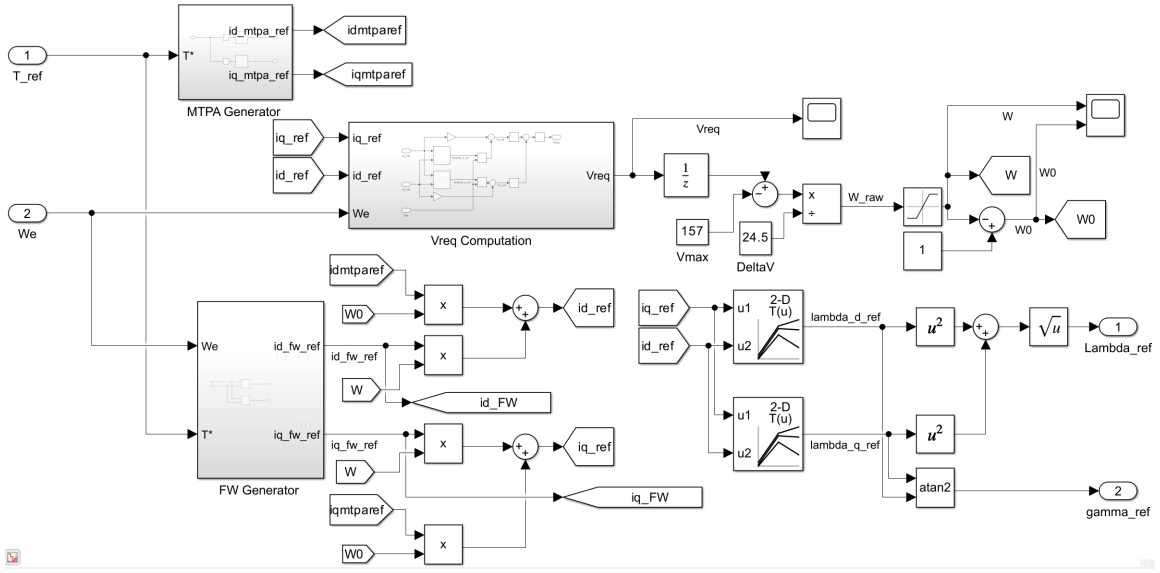
The output of the PI controller corresponds to the unsaturated torque reference $T_{\text{ref,unsat}}$. This signal is then limited through a saturation block, and the resulting limited signal $T_{\text{ref,sat}}$ is used as the torque reference supplied to the reference-generation block.

To improve the controller behavior in the presence of torque saturation, a back-calculation anti-windup mechanism is included in the integral path.

5.4.5 Reference Generation



(a) Top-level nonlinear reference-generation block.



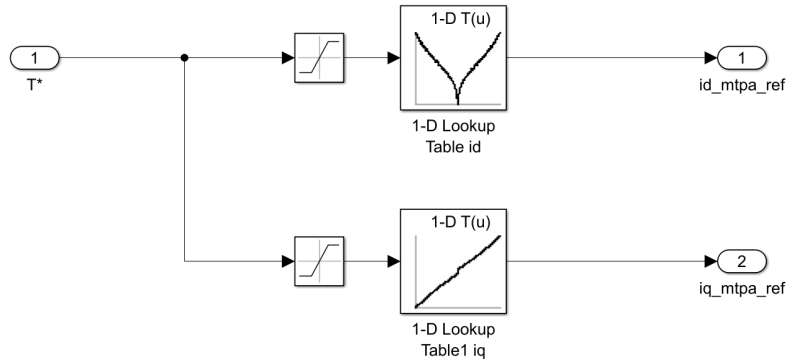
(b) Internal implementation of the nonlinear reference generator.

Figure 5.14: Reference-generation structure adopted in the nonlinear implementation.

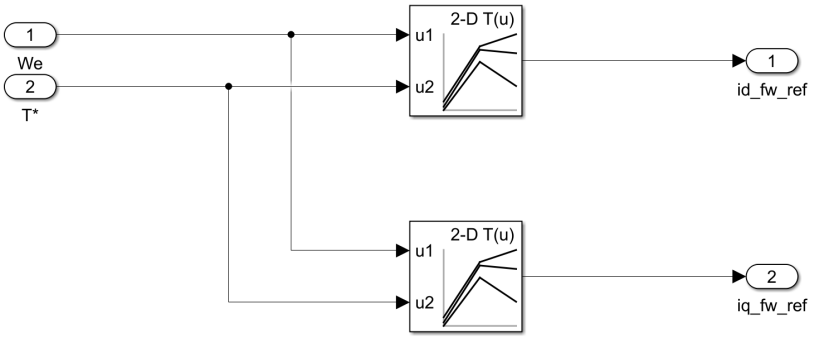
The reference-generation block used in the nonlinear implementation receives the torque reference T_{ref} generated by the outer speed loop together with the electrical speed ω_e, and produces the flux-magnitude reference λ^* and the flux-angle reference γ^* required by the inner Flux Polar Control loops.

The nonlinear reference generator is based on current-reference scheduling. It first determines the current references associated with MTPA and flux-weakening operation, and then combines them according to the operating condition.

MTPA and Flux-Weakening Current References



(a) MTPA current-reference generation.



(b) Flux-weakening current-reference generation.

Figure 5.15: Current-reference generation used in the nonlinear reference block.

Figure 5.15a shows the MTPA current-reference generation used in the nonlinear drive. The torque reference is used as input to one-dimensional lookup tables that provide the corresponding current references i_d^{MTPA} and i_q^{MTPA}.

Figure 5.15b shows the flux-weakening current-reference generation. In this case, two-dimensional lookup tables are used to determine the corresponding current references i_d^{FW} and i_q^{FW} as functions of the electrical speed ω_e and the torque reference.

Required Voltage Evaluation and Reference Blending

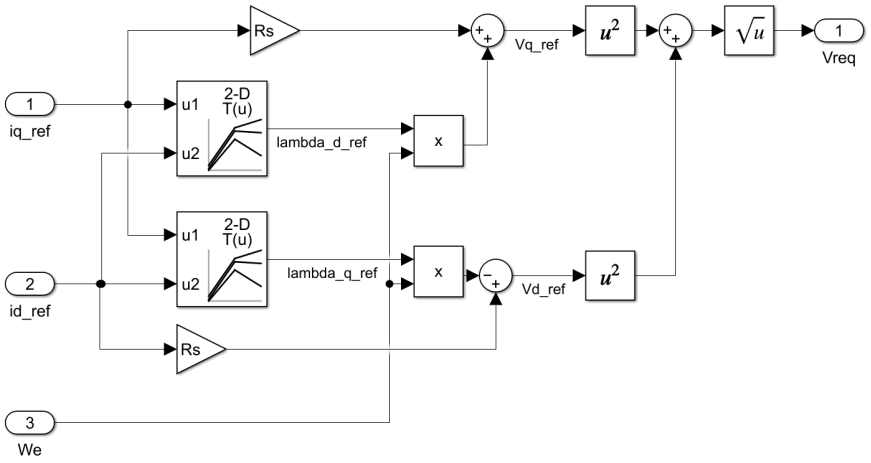


Figure 5.16: Computation of the required stator voltage magnitude.

To determine whether flux weakening is required, the voltage demand associated with the scheduled current references is evaluated through the block shown in Figure 5.16. The reference current components are first mapped into the corresponding flux-linkage references $\lambda_d(i_d^*, i_q^*)$ and $\lambda_q(i_d^*, i_q^*)$ through the nonlinear flux maps. These quantities are then used together with the stator resistance and electrical speed to estimate the corresponding steady-state voltage components:

$$v_d^* = R_s i_d^* - \omega_e \lambda_q^*, \qquad v_q^* = R_s i_q^* + \omega_e \lambda_d^*.$$

The required voltage magnitude is then computed as

$$V_{\text{req}} = \sqrt{(v_d^*)^2 + (v_q^*)^2}.$$

This quantity is compared with the available voltage limit in order to determine the operating region of the machine.

Rather than switching abruptly between the two operating modes, the implemented reference generator uses a smooth blending strategy based on weighting factors. In this way, the final current references are obtained as a continuous combination of the MTPA and FW current references.

Flux-Magnitude and Flux-Angle References

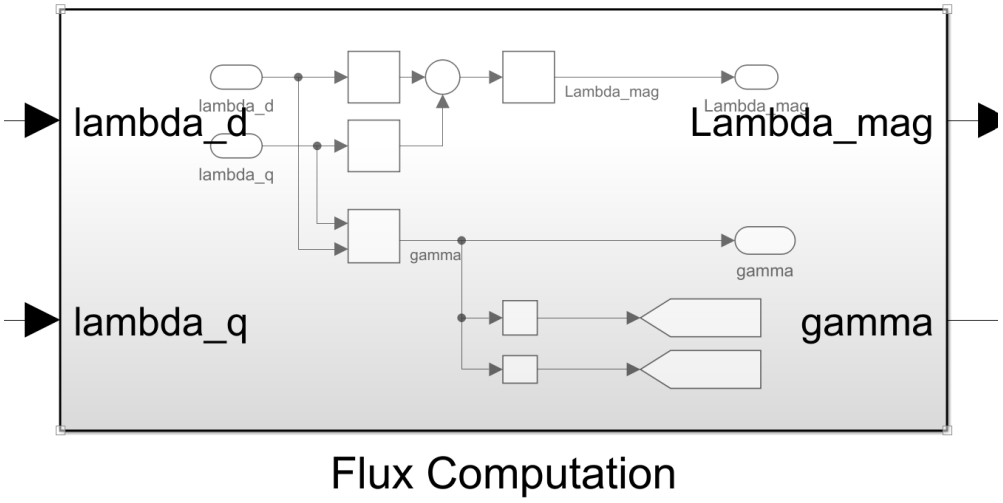
Once the final current references i_d^* and i_q^* have been obtained, they are converted into the corresponding flux-linkage references through the same nonlinear lookup tables used in the machine model:

$$\lambda_d^* = \lambda_d(i_d^*, i_q^*), \qquad \lambda_q^* = \lambda_q(i_d^*, i_q^*).$$

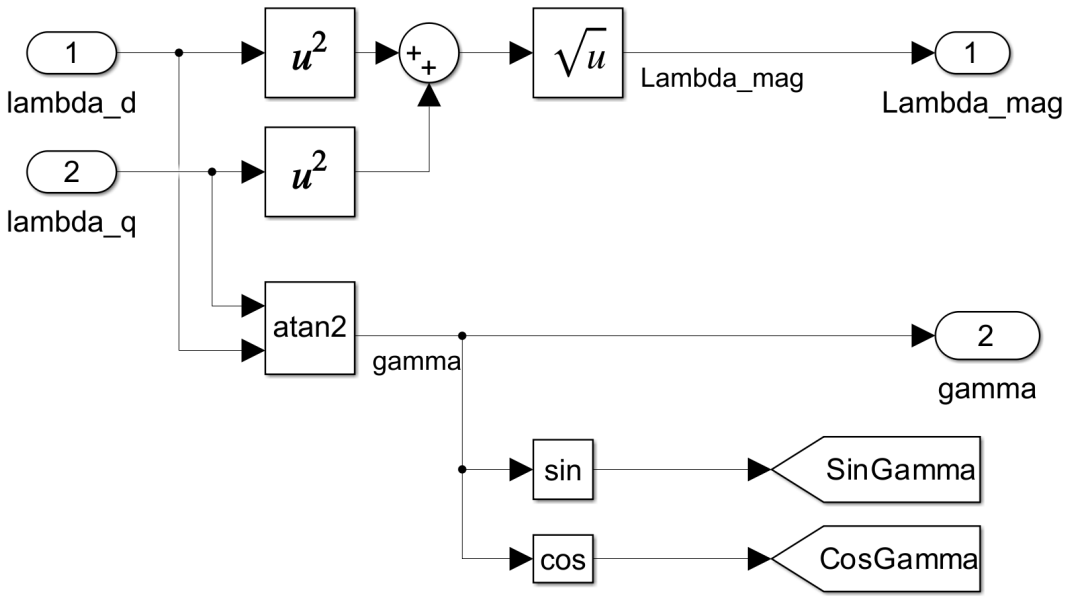
From these quantities, the flux-magnitude and flux-angle references supplied to the Flux Polar Controller are finally computed as

$$\lambda^* = \sqrt{(\lambda_d^*)^2 + (\lambda_q^*)^2}, \qquad \gamma^* = \operatorname{atan2}(\lambda_q^*, \lambda_d^*).$$

5.4.6 Flux Computation



(a) Top-level flux-computation block.



(b) Internal implementation of the flux-computation block.

Figure 5.17: Flux-computation structure adopted in the nonlinear implementation.

The flux-computation block receives the nonlinear machine outputs λ_d and λ_q and converts them into the feedback quantities required by the Flux Polar Controller, namely the flux magnitude λ and the flux angle γ.

The block also evaluates the auxiliary quantities $\sin(\gamma)$ and $\cos(\gamma)$, which are used internally by the control law.

Since the nonlinear machine model directly provides the flux-linkage components, no additional

flux observer is required in this implementation.

5.4.7 Flux Polar Control Loops

The inner Flux Polar Control structure adopted in the nonlinear implementation is the same as the one previously described for the linear IPMSM drive in Section 5.3.4.

In the nonlinear drive, the reference values λ^* and γ^* are generated from the nonlinear lookup-table-based reference generator, while the feedback quantities λ and γ are computed directly from the machine flux-linkage components λ_d and λ_q.

5.5 Simulation Parameters and Test Conditions

This section summarizes the simulation parameters and operating conditions used to evaluate the implemented drive systems. Since the linear and nonlinear implementations are based on different machine models and operating constraints, their parameters are reported separately for clarity.

5.5.1 Linear IPMSM Simulation Parameters

Table 5.1 summarizes the main parameters used for the simulation of the linear IPMSM drive.

Table 5.1: Simulation parameters for the linear IPMSM model.

Parameter	**Symbol**	**Value**
Sampling time	T_s	$100\,\mu$s
Stator resistance	R_s	$0.3\,\Omega$
d-axis inductance	L_d	$4\,$mH
q-axis inductance	L_q	$28\,$mH
Permanent magnet flux	λ_{PM}	$61.4\,$mWb
Pole pairs	p	2
Mechanical inertia	J	$0.1\,\mathrm{kg\,m^2}$
Viscous friction coefficient	B	$0.01\,\mathrm{N\,m\,s}$
Maximum current	$I_{\max}$	$24.75\,$A
Maximum voltage	$V_{\max}$	$240\,$V
Input torque command	T_{in}	$30\,\mathrm{N\,m}$
Maximum torque reference	$T_{\max}$	$34\,\mathrm{N\,m}$

In the linear implementation, the imposed quantity is the input torque command supplied to the reference-generation block, rather than the electromagnetic torque itself. Starting from this input, the reference generator produces the corresponding flux-magnitude and flux-angle references.

5.5.2 Nonlinear Map-Based Drive Simulation Parameters

Table 5.2 summarizes the parameters used for the nonlinear map-based drive.

Table 5.2: Simulation parameters for the nonlinear map-based drive.

Parameter	**Symbol**	**Value**
Sampling time	T_s	$100\,\mu$s
Stator resistance	R_s	$0.0286\,\Omega$
Pole pairs	p	2
Mechanical inertia	J	$0.05\,\mathrm{kg\,m^2}$
Viscous friction coefficient	B	$0.001\,\mathrm{N\,m\,s}$
Maximum voltage	$V_{\max}$	$157\,\mathrm{V}$
Voltage transition band	ΔV	$24.5\,\mathrm{V}$

In this case, the electromagnetic behavior of the machine is obtained directly from nonlinear flux-linkage and torque maps.

5.5.3 Reference Profiles and Load Conditions

The implemented drive systems were tested under different operating conditions in order to evaluate both steady-state and transient performance.

For the linear implementation, tests were carried out by imposing torque references, focusing on the validation of the Flux Polar Control loops.

For the nonlinear implementation, the adopted test conditions included both speed-controlled and load-disturbance scenarios. In this case, the outer speed loop, the nonlinear MTPA/FW reference-generation block, and the inner Flux Polar Control loops were tested together as part of the complete drive system.

The main simulation scenarios considered in the analysis include:

- operation in the constant-torque region, below or around nominal speed;
- operation in the flux-weakening region, above nominal speed;
- no-load operation;
- loaded operation under different values of load torque;
- transient speed-reference variations;
- disturbance rejection under applied load-torque steps.

In the nonlinear speed-controlled tests, different load-torque values were considered in order to evaluate the dynamic response and robustness of the drive under increasingly demanding operating conditions, with particular attention to the transition between MTPA and flux-weakening operation.

The corresponding simulation results are presented and discussed in Chapter 6.

Chapter 6

Results and Discussion

6.1 Introduction

This chapter presents the simulation results obtained from the implemented drive systems and discusses their main dynamic and steady-state characteristics. The objective is to assess the behavior of the Flux Polar Control strategy under linear and nonlinear machine representations and under different operating conditions.

The chapter is organized progressively. First, the linear IPMSM implementation is examined in order to validate the inner Flux Polar Control structure under controlled operating conditions. Then, the nonlinear map-based drive is evaluated under no-load, loaded, and flux-weakening conditions. The chapter concludes with a comparative discussion of the obtained results.

6.2 Linear IPMSM Results

This section presents the simulation results obtained with the linear IPMSM implementation. The analysis first considers the two inner control loops individually and then examines the behavior of the complete drive under a speed-varying operating condition.

6.2.1 Validation of the Inner Flux Polar Control Loops

The first set of simulations was carried out in order to evaluate the dynamic behavior of the two inner loops composing the Flux Polar Controller, namely the flux-magnitude loop and the flux-angle loop.

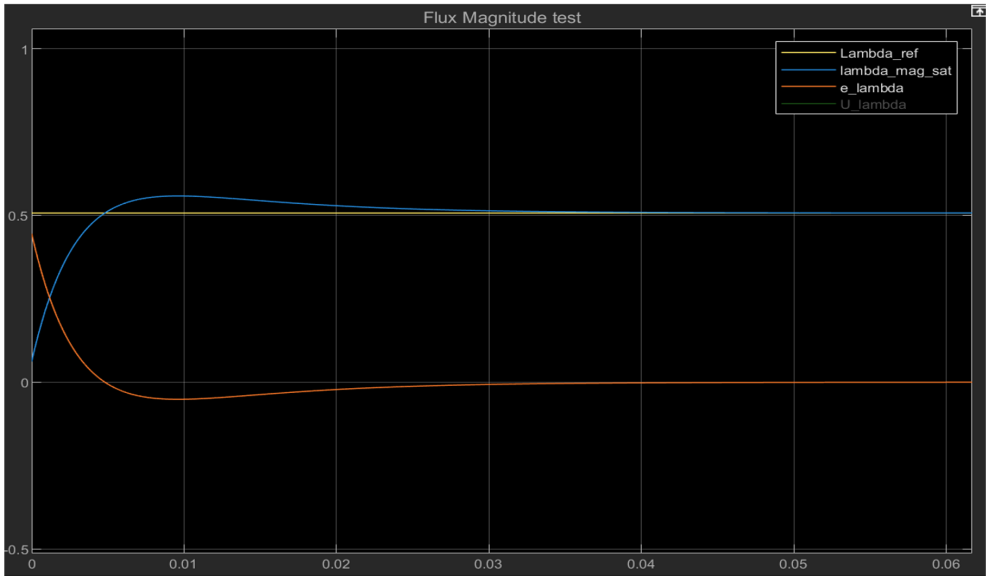


Figure 6.1: Flux-magnitude loop response in the linear IPMSM implementation.

Figure 6.1 shows the response of the flux-magnitude loop. The estimated flux magnitude follows the imposed reference with a stable and well-damped transient. A limited overshoot is observed during the initial response, after which the flux converges toward the reference value with negligible steady-state error. The corresponding error signal rapidly decreases to zero, confirming the correct operation of the flux PI regulator.

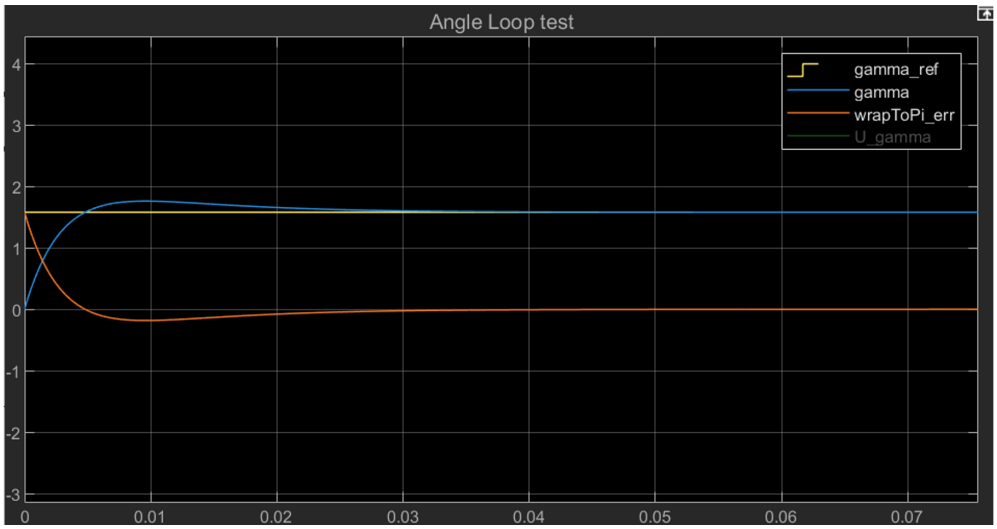


Figure 6.2: Flux-angle loop response in the linear IPMSM implementation.

Figure 6.2 shows the response of the flux-angle loop. Also in this case, the controlled variable converges to the reference with a stable transient and small steady-state error. The wrapped angular error remains well behaved during the transient, which confirms the effectiveness of the adopted angular-error formulation in the controller.

Overall, these results show that the two inner Flux Polar Control loops are able to regulate the controlled quantities independently and with satisfactory dynamic performance.

6.2.2 Behavior Under Speed-Varying Operation

After validating the inner control loops individually, the complete linear drive was evaluated under a speed-varying operating condition.

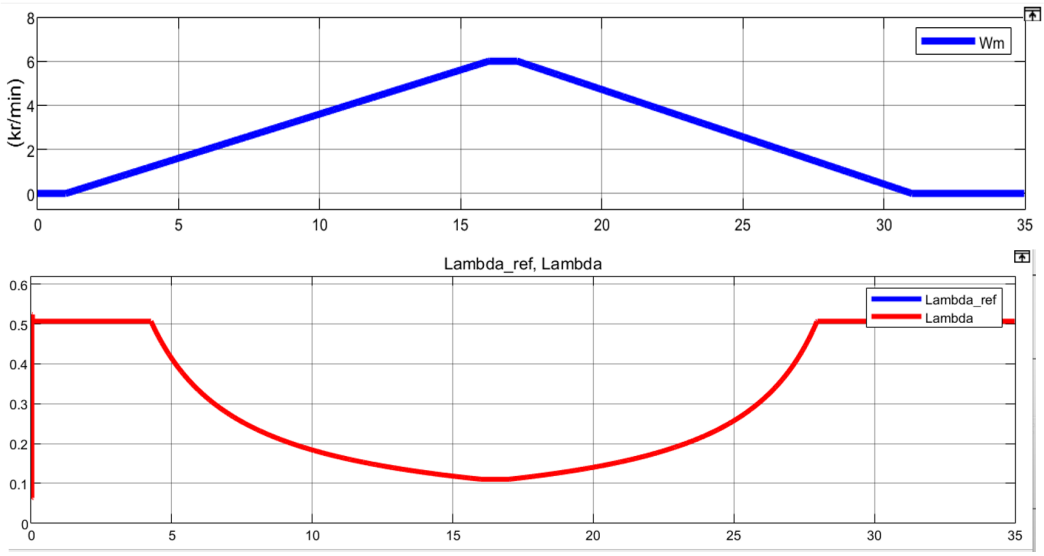


Figure 6.3: Imposed speed profile and corresponding flux-magnitude evolution in the linear IPMSM drive.

Figure 6.3 shows the imposed speed profile together with the evolution of the flux magnitude. At low speed, the flux magnitude remains close to its reference value, corresponding to operation in the constant-torque region. As the speed increases, the reference generator progressively reduces the flux reference, and the controlled flux follows this variation accordingly. This behavior is consistent with the expected transition toward flux-weakening operation.

As the speed decreases again, the flux magnitude returns toward its initial value, showing that the controller is able to recover the higher-flux operating condition when the voltage constraint becomes less restrictive.

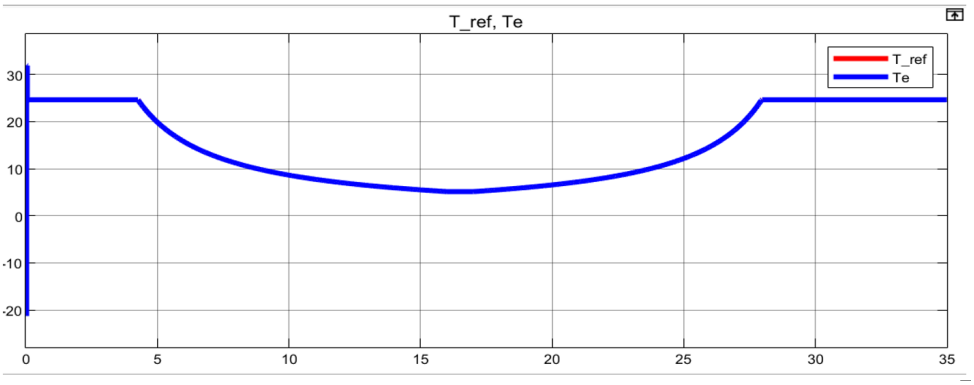


Figure 6.4: Torque behavior under the imposed speed-varying condition in the linear IPMSM drive.

Figure 6.4 shows the corresponding electromagnetic torque behavior. The reference torque decreases as the speed enters the flux-weakening region, reflecting the reduction in the achievable torque under voltage-limited operation. The electromagnetic torque follows the same general

trend, indicating that the implemented drive remains consistent with the reference-generation strategy over the considered operating range.

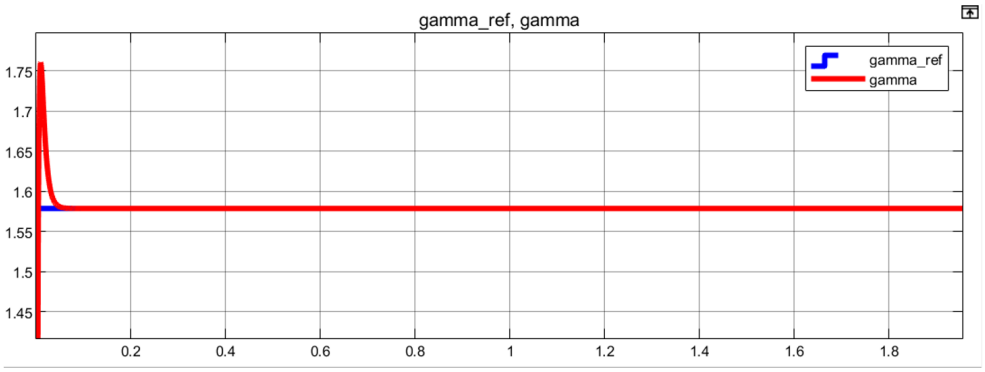


Figure 6.5: Reference and actual flux angle in the linear IPMSM drive.

Figure 6.5 shows the evolution of the flux angle during the same test. After a short initial transient, the actual flux angle rapidly converges to the reference and remains closely aligned with it over the considered time interval.

Taken together, these results confirm that the linear implementation provides a coherent validation of the Flux Polar Control strategy. The controller remains consistent with the expected torque and flux behavior as the operating point moves from the constant-torque region toward the flux-weakening region.

6.3 Nonlinear Map-Based Drive Results

This section presents the simulation results obtained with the nonlinear map-based machine model, where the electromagnetic behavior is described through flux-linkage, torque, and Jacobian lookup tables.

6.3.1 No-Load Operation Under Positive Speed Profile

The first nonlinear simulation was carried out under no-load conditions ($T_L = 0$ N.m) using a positive speed reference profile.

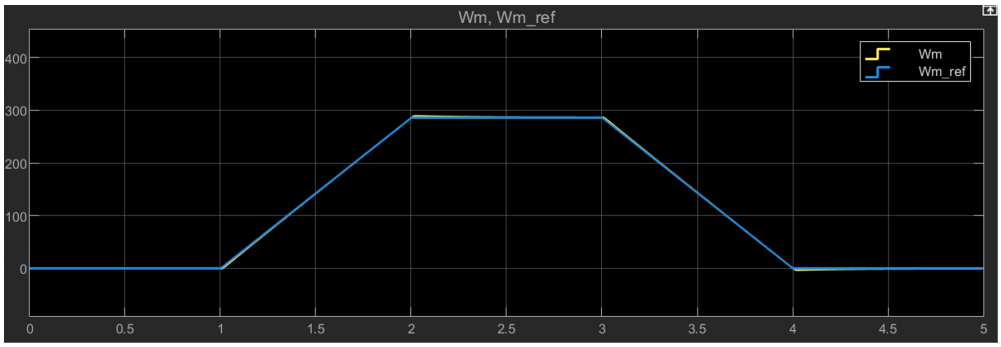


Figure 6.6: Mechanical speed response under no-load conditions and positive speed profile.

Figure 6.6 shows the mechanical speed response. The actual speed closely follows the imposed reference throughout the entire test, including the acceleration, constant-speed, and deceleration intervals. The absence of significant oscillations or tracking delay indicates stable and well coordinated operation under no-load conditions.

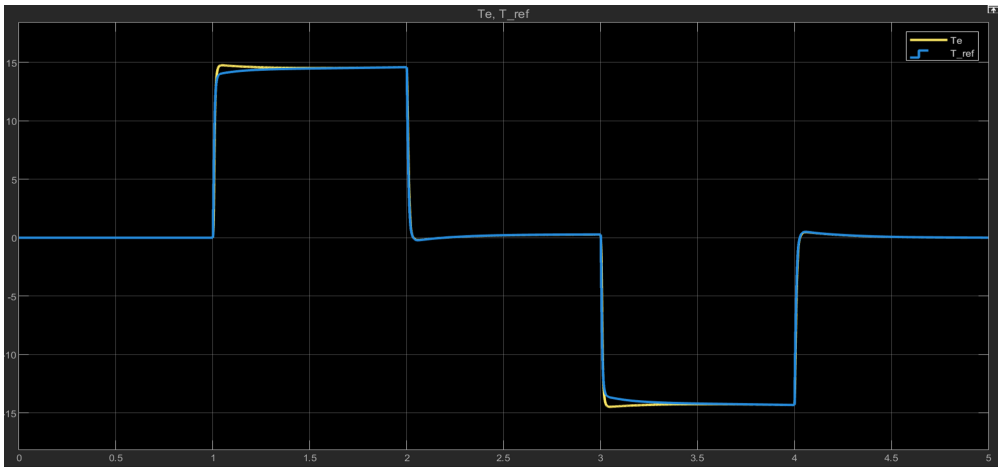


Figure 6.7: Electromagnetic torque and torque reference under no-load conditions and positive speed profile.

Figure 6.7 shows the electromagnetic torque and its reference. The torque follows the reference with a fast transient and limited overshoot at the main operating-point transitions. Positive torque is produced during acceleration, negative torque appears during deceleration, and the torque remains close to zero during the constant-speed intervals, which is physically consistent with no-load operation.

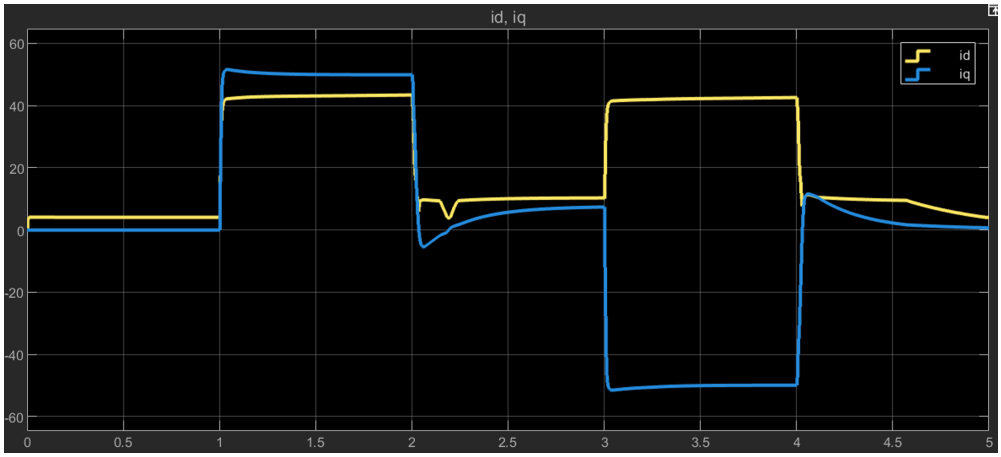


Figure 6.8: Current components i_d and i_q under no-load conditions and positive speed profile.

Figure 6.8 shows the evolution of the current components. The i_q component changes sign according to the sign of the requested torque, while i_d varies consistently with the operating-point changes imposed by the reference-generation block.

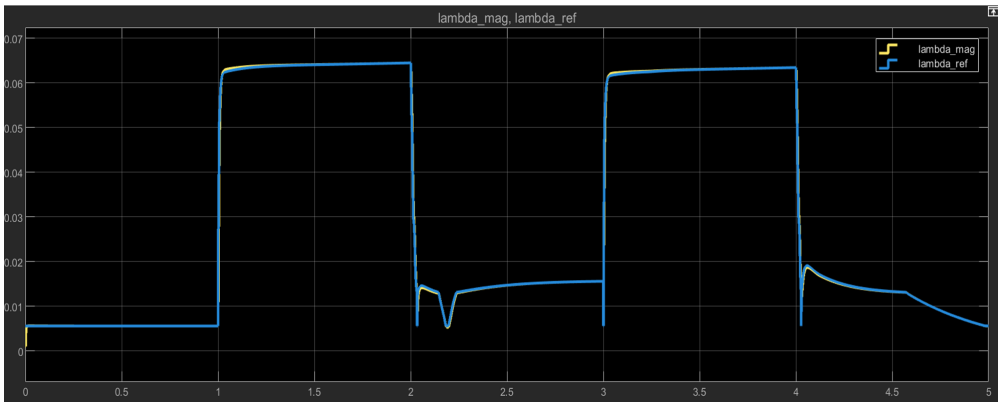


Figure 6.9: Flux magnitude and flux-magnitude reference under no-load conditions and positive speed profile.

Figure 6.9 shows the flux-magnitude response. The actual flux magnitude follows the reference very closely over the whole simulation interval, with only small transient deviations during the operating-point transitions.

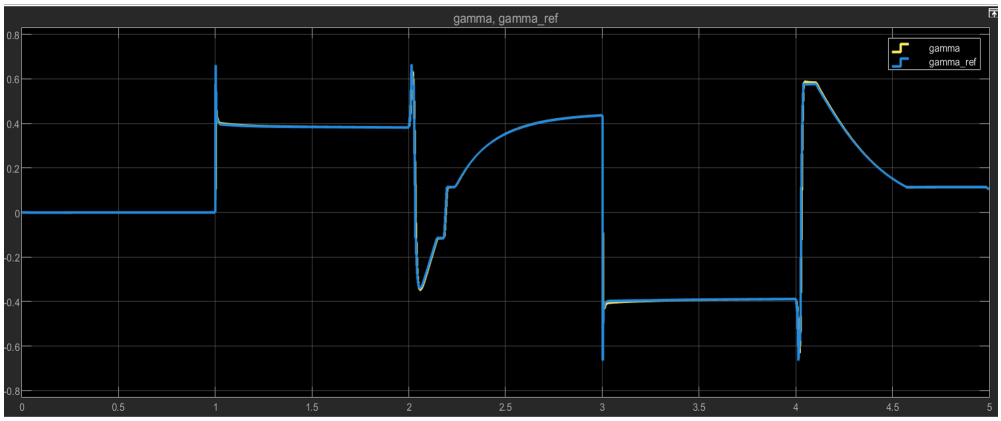


Figure 6.10: Flux angle and flux-angle reference under no-load conditions and positive speed profile.

Figure 6.10 shows the evolution of the flux angle. The actual flux angle tracks the reference with good accuracy in all operating intervals. The main transients are observed during abrupt reference changes, but the controlled variable rapidly converges to the desired value.

Overall, these results confirm that the nonlinear map-based drive is able to operate correctly under no-load conditions and positive speed variation, with no undesired activation of flux weakening in this operating range.

6.3.2 Loaded Operation Under Negative Speed Profile

A second nonlinear simulation was carried out under a constant load torque of $T_L = 100$ N m using a negative speed reference profile.

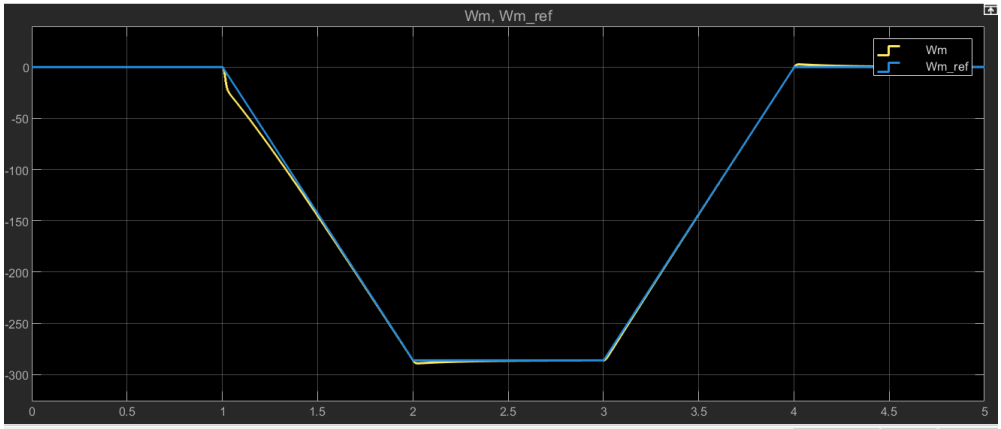


Figure 6.11: Mechanical speed response under $T_L = 100$ N.m and negative speed profile.

Figure 6.11 shows the mechanical speed response. The actual speed follows the imposed negative reference with good accuracy during the acceleration, constant-speed, and deceleration intervals. A small transient deviation is visible during the initial speed reversal, but the response rapidly stabilizes and remains closely aligned with the reference.

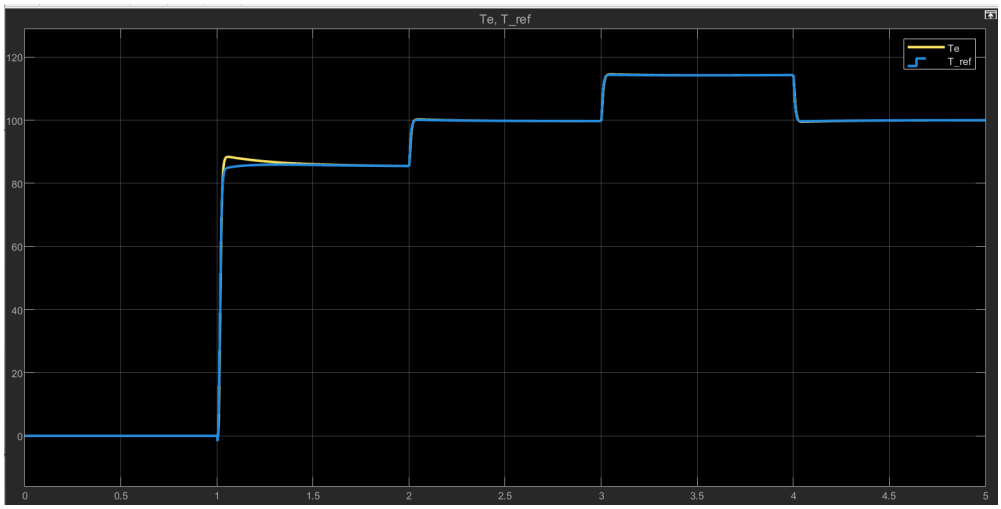


Figure 6.12: Electromagnetic torque and torque reference under $T_L = 100$ N.m and negative speed profile.

Figure 6.12 shows the electromagnetic torque response. The torque follows the reference with good accuracy, with only a limited overshoot during the main step transitions. The controller is therefore able to generate the torque required to balance the external load and to impose the desired speed trajectory even in reverse operation.

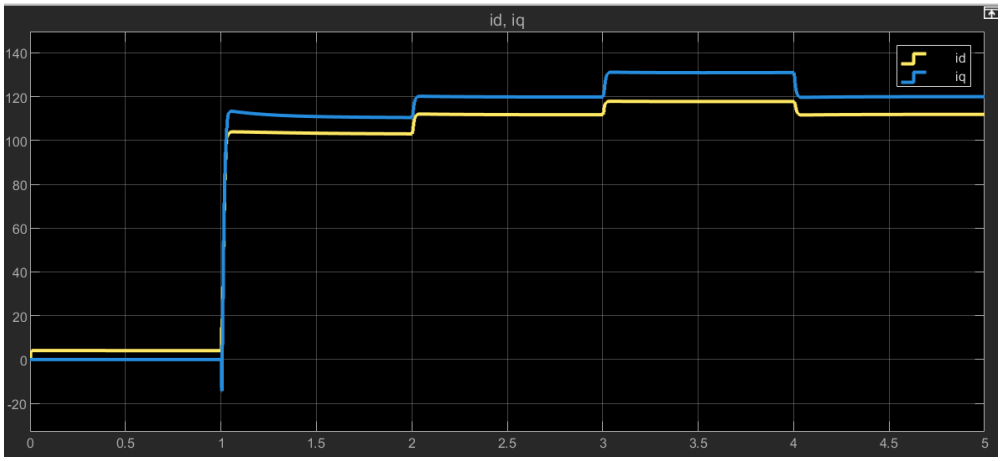


Figure 6.13: Current components i_d and i_q under $T_L = 100$ N.m and negative speed profile.

Figure 6.13 shows the corresponding current components. Compared to the no-load case, both current components increase significantly, as expected under loaded operation.

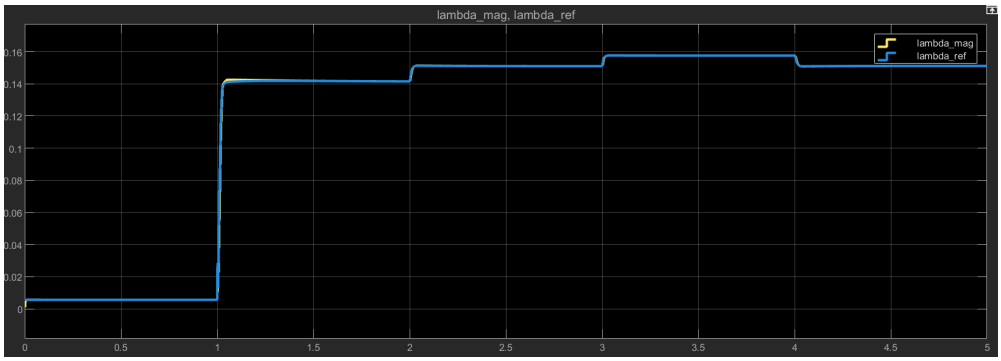


Figure 6.14: Flux magnitude and flux-magnitude reference under $T_L = 100$ N.m and negative speed profile.

Figure 6.14 shows the flux-magnitude response. The actual flux magnitude follows the reference with very small tracking error throughout the whole test.

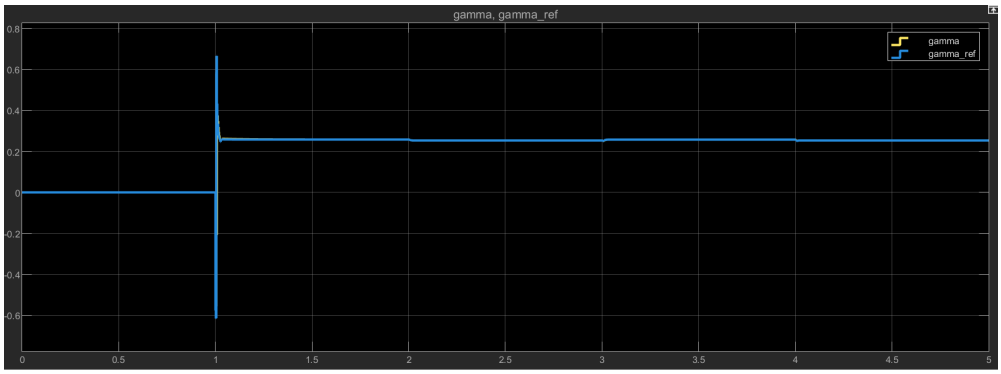


Figure 6.15: Flux angle and flux-angle reference under $T_L = 100$ N.m and negative speed profile.

Figure 6.15 shows the evolution of the flux angle. Also in this case, the actual flux angle closely follows the reference after a short initial transient and remains stable over the whole simulation interval.

Overall, these results confirm that the nonlinear map-based drive preserves correct dynamic behavior under nonzero load torque and reverse speed operation.

6.3.3 Operation Above Nominal Speed and Flux-Weakening Behavior

A further nonlinear simulation was carried out under a load torque of $T_L = 300$ N.m using a speed profile that exceeds the nominal operating speed. The purpose of this test is to evaluate the behavior of the implemented drive when the machine enters the flux-weakening region, where the available stator voltage becomes the dominant limitation.

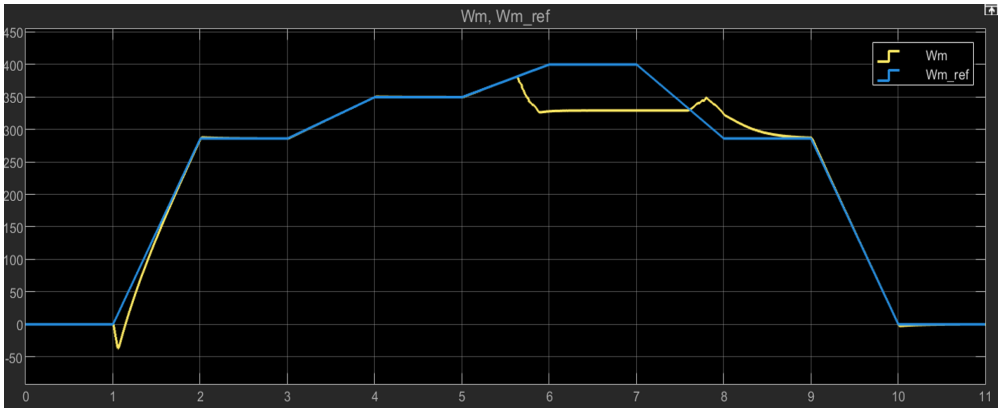


Figure 6.16: Mechanical speed response under $T_L = 300$ N.m and speed profile exceeding nominal speed.

Figure 6.16 shows the mechanical speed response. Up to nominal speed, the drive follows the imposed reference with satisfactory accuracy. Once the reference enters the higher-speed region, however, the tracking quality degrades and the actual speed no longer follows the reference perfectly. This behavior is expected, since above nominal speed the available voltage becomes a limiting factor.

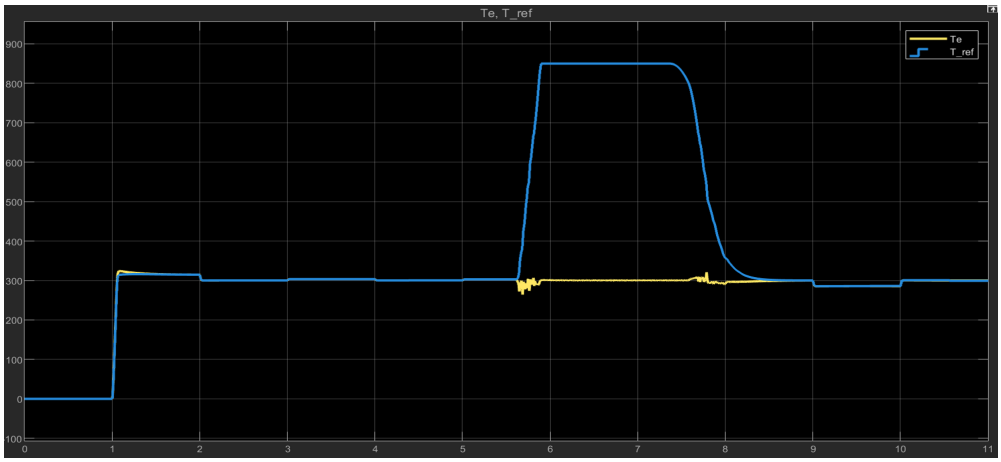


Figure 6.17: Electromagnetic torque and torque reference under $T_L = 300$ N.m in flux-weakening conditions.

Figure 6.17 shows the electromagnetic torque response. Below nominal speed, the torque tracks the reference well and remains close to the requested operating point. When the speed reference is increased further, the torque request rises sharply, but the actual electromagnetic torque cannot follow the imposed reference and remains limited around the machine capability.

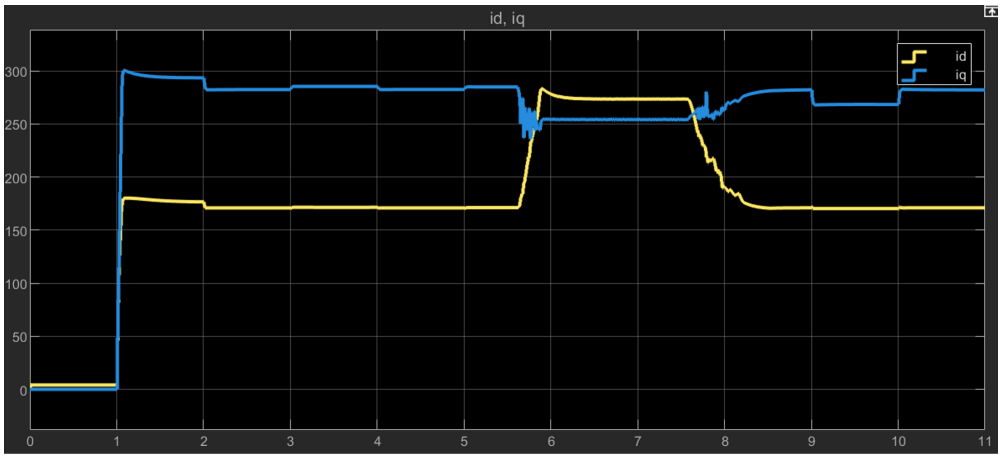


Figure 6.18: Current components i_d and i_q under $T_L = 300$ N.m in flux-weakening conditions.

Figure 6.18 shows the evolution of the current components. Below nominal speed, the current trajectories remain relatively stable and consistent with the MTPA operating region. When the speed increases beyond the base-speed region, a clear redistribution of the current components is observed.

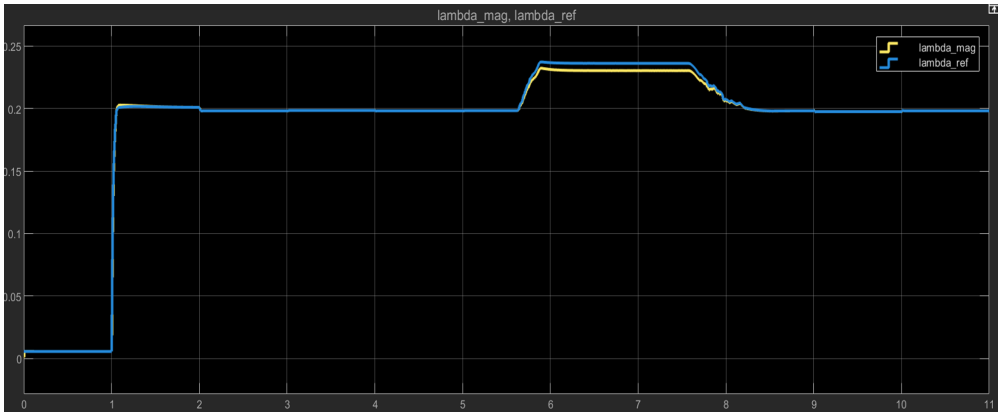


Figure 6.19: Flux magnitude and flux-magnitude reference under $T_L = 300$ N.m in flux-weakening conditions.

Figure 6.19 shows the flux-magnitude response. In the constant-flux region, the tracking remains satisfactory and the actual flux closely follows the reference. Once the machine enters the flux-weakening region, the flux magnitude is reduced as expected in order to satisfy the voltage limitation at high speed. Although the flux loop remains qualitatively consistent, the tracking becomes less accurate during the most demanding interval.

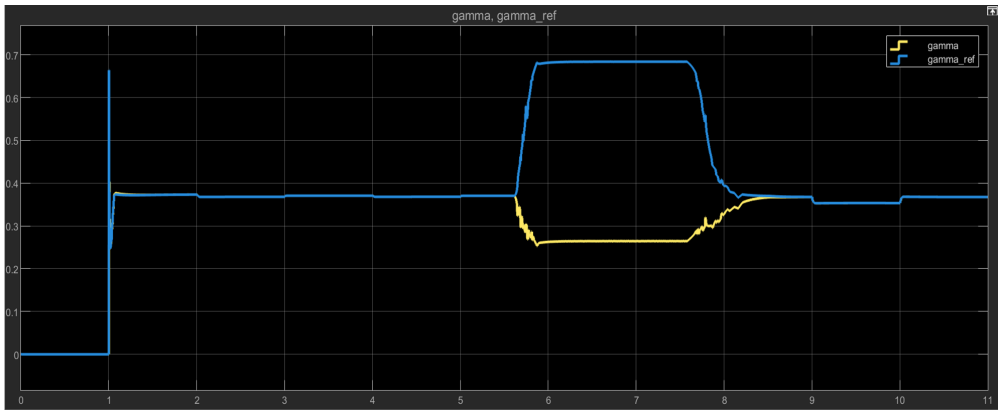


Figure 6.20: Flux angle and flux-angle reference under $T_L = 300$ N.m in flux-weakening conditions.

Figure 6.20 shows the behavior of the flux angle. Below nominal speed, the tracking is satisfactory and remains close to the imposed reference. When the machine enters flux weakening, however, a more pronounced mismatch appears between the actual and reference flux angle.

Overall, these results confirm the expected transition from the MTPA region to the flux-weakening region in the nonlinear map-based drive. The simulation shows that the implemented control structure preserves physically coherent behavior above nominal speed, including flux reduction and current redistribution, while also highlighting the degradation in tracking performance caused by voltage limitation and reduced torque capability in this operating region.

6.4 Comparative Discussion

The linear and nonlinear implementations presented in this work provide two complementary validation levels for the adopted Flux Polar Control strategy. The linear IPMSM model was useful for verifying the control structure under simplified and analytically tractable machine assumptions, where the machine behavior is described through constant electrical parameters.

The nonlinear map-based implementation provided a more realistic assessment of the same control architecture. By representing the machine through magnetic flux and torque maps, the model was able to capture saturation effects, cross-coupling, and operating-point-dependent behavior that cannot be represented in the linear case.

Overall, the comparison confirms that the adopted Flux Polar Control structure remains applicable in both cases. The linear model served as a useful control validation platform, whereas the nonlinear model provided a more realistic evaluation of the drive performance under physically meaningful operating conditions.

Chapter 7

Conclusions and Future Work

This dissertation presented the implementation of a Flux Polar Control strategy for synchronous motor drives, first on a linear IPMSM model and subsequently on a nonlinear map-based synchronous reluctance motor model.

The linear implementation provided a useful framework for validating the control structure under simplified machine assumptions and for verifying the regulation of the flux magnitude and flux angle. The nonlinear implementation then allowed the same control philosophy to be assessed under more realistic conditions, where the machine behavior is represented directly through flux, torque, and differential magnetic maps.

The obtained results show that the adopted control architecture is capable of preserving stable and physically coherent behavior across different operating conditions. In particular, the nonlinear simulations confirmed satisfactory operation in the nominal region and reproduced the expected transition toward flux-weakening operation at higher speed. At the same time, the results highlighted the practical limitations introduced by voltage constraints and nonlinear machine behavior, especially under high-load and high-speed conditions.

Overall, this work confirms that Flux Polar Control constitutes a viable and structured control approach for synchronous machine drives, while also highlighting the importance of nonlinear machine modeling for a more realistic assessment of control performance.

Possible future developments include a more systematic tuning of the control loops, an improved design of the flux-weakening reference-generation strategy, and the experimental validation of the implemented control architecture on a real machine drive setup.

Bibliography

[1] W. Leonhard, *Control of Electrical Drives*, 3rd ed. Springer, 2001.

[2] R. Krishnan, *Permanent Magnet Synchronous and Brushless DC Motor Drives*. CRC Press, 2010.

[3] Z. Q. Zhu and D. Howe, "Review of flux-weakening control techniques for permanent-magnet ac motor drives," *IET Electric Power Applications*, vol. 4, no. 1, pp. 17–25, 2010.

[4] P. C. Krause, O. Wasynczuk, S. D. Sudhoff, and S. D. Pekarek, *Analysis of Electric Machinery and Drive Systems*, 3rd ed. Wiley-IEEE Press, 2013.

[5] I. Boldea and L. N. Tutelea, *Reluctance Synchronous Machines and Drives*. Oxford University Press, 2014.

[6] F. Blaschke, "The principle of field orientation as applied to the new transvector closed-loop control system for rotating-field machines," *Siemens Review*, vol. 39, no. 5, pp. 217–220, 1972.

[7] I. Takahashi and T. Noguchi, "A new quick-response and high-efficiency control strategy of an induction motor," *IEEE Transactions on Industry Applications*, vol. IA-22, no. 5, pp. 820–827, 1986.

[8] A. Formentini, S. Rubino, and P. Zanchetta, "Direct flux vector control of synchronous motor drives," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 7, pp. 4661–4671, 2016.

[9] S. Rubino, A. Formentini, P. Zanchetta, and N. Bianchi, "Flux polar control: A unified torque control for ac motor drives," *IEEE Transactions on Industry Applications*, vol. 55, no. 6, pp. 7557–7567, 2019.

[10] S. Rubino, L. Tolosano, F. Mandrile, E. Armando, and R. Bojoi, "Flux polar control (fpc): A unified torque controller for ac motor drives," *IEEE Transactions on Industry Applications*, vol. 59, no. 4, pp. 4140–4163, July/August 2023.

[11] A. Vagati, M. Pastorelli, G. Franceschini, and V. Drogoreanu, "Flux-observer-based high-performance control of synchronous reluctance motors by including cross saturation," *IEEE Transactions on Industry Applications*, vol. 35, no. 3, pp. 597–605, May/June 1999.

[12] A. Bertozzi, N. Bianchi, and E. Fornasiero, "Accurate flux-linkage lookup tables for ipm machines via finite-element analysis," *IEEE Transactions on Industry Applications*, vol. 55, no. 1, pp. 123–131, 2019.

[13] N. Bianchi, S. Bolognani, and M. Dahidah, "Maximum torque per ampere control of ipm motors based on analytical expressions," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 1, pp. 250–257, 2010.

Appendix A

MATLAB Codes

Overview

This appendix reports the main MATLAB scripts used in the implementation of the linear and nonlinear drive models. The reported scripts include the generation of the lookup tables adopted in the reference generation stage, as well as the preprocessing of the nonlinear machine maps used in the map-based synchronous reluctance motor model.

A.1 Linear Model Codes

A.1.1 MTPA and Flux-Angle LUT Generation

```
%% MTPA and 2D flux-angle LUT generation for the linear IPMSM
clear; clc;

%% Machine and simulation parameters
Rs = 0.3;               % Stator resistance [Ohm]
Ld = 4e-3;              % d-axis inductance [H]
Lq = 28e-3;             % q-axis inductance [H]
LambdaPM = 61.4e-3;     % Permanent-magnet flux linkage [Wb]
p = 2;                  % Pole pairs
J = 0.1;                % Mechanical inertia [kg*m^2]
B = 0.01;               % Viscous friction coefficient [N*m*s]
Ts = 1e-4;              % Sampling time [s]

kT = 1.5 * p;           % Torque constant factor

%% Current sweep used to compute the MTPA trajectory
is_max = 24.75;         % Maximum current magnitude [A]
Nis = 400;
is_vec = linspace(1e-3, is_max, Nis);

id_opt = zeros(size(is_vec));
iq_opt = zeros(size(is_vec));
T_mtpa = zeros(size(is_vec));
lambda_mtpa = zeros(size(is_vec));
gamma_mtpa = zeros(size(is_vec));

%% MTPA computation: maximize torque at fixed current magnitude
for kk = 1:length(is_vec)
    is = is_vec(kk);

    iq_from_id = @(id) sqrt(max(0, is^2 - id.^2));
```

```
32
33      Te_from_id = @(id) kT * ( ...
34          LambdaPM * iq_from_id(id) + ...
35          (Ld - Lq) * id .* iq_from_id(id) );
36
37      fun = @(id) -Te_from_id(id);
38
39      id_star = fminbnd(fun, -is, 0);
40      iq_star = iq_from_id(id_star);
41
42      lambda_d = Ld * id_star + LambdaPM;
43      lambda_q = Lq * iq_star;
44
45      id_opt(kk) = id_star;
46      iq_opt(kk) = iq_star;
47      T_mtpa(kk) = Te_from_id(id_star);
48      lambda_mtpa(kk) = sqrt(lambda_d^2 + lambda_q^2);
49      gamma_mtpa(kk) = atan2(lambda_q, lambda_d);
50  end
51
52  %% Torque-indexed 1D LUTs along the MTPA trajectory
53  [T_sorted, idx] = sort(T_mtpa);
54  id_sorted  = id_opt(idx);
55  iq_sorted  = iq_opt(idx);
56  lam_sorted = lambda_mtpa(idx);
57  gam_sorted = gamma_mtpa(idx);
58
59  [T_unique, iu] = unique(T_sorted, 'stable');
60  id_unique  = id_sorted(iu);
61  iq_unique  = iq_sorted(iu);
62  lam_unique = lam_sorted(iu);
63  gam_unique = gam_sorted(iu);
64
65  NT = 300;
66  Tref = linspace(0, T_unique(end), NT);
67
68  id_LUT  = interp1(T_unique, id_unique,  Tref, 'pchip', 'extrap');
69  iq_LUT  = interp1(T_unique, iq_unique,  Tref, 'pchip', 'extrap');
70  lam_LUT = interp1(T_unique, lam_unique, Tref, 'pchip', 'extrap');
71  gam_LUT = interp1(T_unique, gam_unique, Tref, 'pchip', 'extrap');
72
73  MTPA.Tref = Tref;
74  MTPA.id = id_LUT;
75  MTPA.iq = iq_LUT;
76  MTPA.lambda_mag = lam_LUT;
77  MTPA.gamma = gam_LUT;
78
79  %% 2D LUT generation: gamma = f(T, lambda)
80  Nd = 181;
81  Nq = 181;
82  id_grid = linspace(-is_max, 0, Nd);
83  iq_grid = linspace(0, is_max, Nq);
84  [ID, IQ] = meshgrid(id_grid, iq_grid);
85
86  Is = sqrt(ID.^2 + IQ.^2);
87  mask = (Is <= is_max);
88
89  IDv = ID(mask);
```

```
IQv = IQ(mask);

lam_d = Ld .* IDv + LambdaPM;
lam_q = Lq .* IQv;
lam_mag = sqrt(lam_d.^2 + lam_q.^2);

T = kT .* (lam_d .* IQv - lam_q .* IDv);
gamma = atan2(lam_q, lam_d);

good = isfinite(T) & isfinite(lam_mag) & isfinite(gamma);
Tg = T(good);
Lg = lam_mag(good);
Gg = gamma(good);

F_gamma = scatteredInterpolant(Tg, Lg, Gg, 'linear', 'nearest');

T_bp = Tref;
lam_min = max(min(lam_LUT) * 0.5, 1e-4);
lam_max = max(lam_LUT);
NL = 200;
lam_bp = linspace(lam_min, lam_max, NL);

[TT, LL] = ndgrid(T_bp, lam_bp);
gamma_table = F_gamma(TT, LL);

FPC2D.T_bp = T_bp;
FPC2D.lam_bp = lam_bp;
FPC2D.gamma_table = gamma_table;

save('MTPA_LUT.mat', 'MTPA', 'FPC2D');
```

Listing A.1: MATLAB script used to generate the MTPA and flux-angle lookup tables for the linear IPMSM model.

A.2 Nonlinear Model Codes

A.2.1 MTPA and Flux-Weakening LUT Generation

```
clear; clc;

%% Machine and operating limits
Vmax = 157;           % Voltage limit [V]
Rs   = 0.0286;        % Stator resistance [Ohm]
p    = 2;             % Pole pairs
Ts   = 1e-4;          % Sampling time [s]
J    = 0.05;          % Mechanical inertia [kg*m^2]
B    = 0.001;         % Viscous friction coefficient [N*m*s]

% Electrical-speed and torque grids used for the FW lookup tables
we_grid = linspace(0, 900, 41);
T_grid  = linspace(-800, 800, 161);

%% Load map data
Smap = load('data_mapping.mat');
map  = Smap.mapping;

Selab = load('data_mapping_elaboration.mat');
```

```
m = Selab.mapping;

% Positive-quadrant breakpoints
id_bp_pos = map.Id_matrix(1,:);
id_bp_pos = id_bp_pos(:)';

iq_bp_pos = map.Iq_matrix(:,1);
iq_bp_pos = iq_bp_pos(:);

% Electromagnetic maps
Te_map       = map.Torque_dq;
lambda_d_map = map.fluxD;
lambda_q_map = map.fluxQ;

% Interpolants defined on the positive quadrant
F_Te = griddedInterpolant({iq_bp_pos, id_bp_pos}, Te_map, ...
                          'linear', 'nearest');

F_ld = griddedInterpolant({iq_bp_pos, id_bp_pos}, lambda_d_map, ...
                          'linear', 'nearest');

F_lq = griddedInterpolant({iq_bp_pos, id_bp_pos}, lambda_q_map, ...
                          'linear', 'nearest');

%% 1D MTPA LUT
id_mtpa = m.id_mtpa(:);
iq_mtpa = m.iq_mtpa(:);

T_mtpa = F_Te(abs(iq_mtpa), abs(id_mtpa));

[T_mtpa_u, idx] = sort(T_mtpa);
id_mtpa_u = id_mtpa(idx);
iq_mtpa_u = iq_mtpa(idx);

[T_mtpa_u, ia] = unique(T_mtpa_u, 'stable');
id_mtpa_u = id_mtpa_u(ia);
iq_mtpa_u = iq_mtpa_u(ia);

keep = [true; diff(T_mtpa_u) > 0];
T_mtpa_u  = T_mtpa_u(keep);
id_mtpa_u = id_mtpa_u(keep);
iq_mtpa_u = iq_mtpa_u(keep);

% Extend the MTPA LUT to positive and negative torque
T_mtpa_full  = [-flipud(T_mtpa_u(:)); T_mtpa_u(:)];
id_mtpa_full = [ flipud(id_mtpa_u(:)); id_mtpa_u(:)];
iq_mtpa_full = [-flipud(iq_mtpa_u(:)); iq_mtpa_u(:)];

%% 2D FW LUT
[ID, IQ] = meshgrid(id_bp_pos, iq_bp_pos);

id_candidates = ID(:);
iq_candidates = IQ(:);

Te_candidates = F_Te(abs(iq_candidates), abs(id_candidates));
ld_candidates = F_ld(abs(iq_candidates), abs(id_candidates));
lq_candidates = F_lq(abs(iq_candidates), abs(id_candidates));

```

```
lambda_mag_candidates = sqrt(ld_candidates.^2 + lq_candidates.^2);

id_fw_lut = nan(length(we_grid), length(T_grid));
iq_fw_lut = nan(length(we_grid), length(T_grid));

for iw = 1:length(we_grid)
    we = we_grid(iw);

    % Steady-state dq voltage for each candidate operating point
    vd_cand = Rs .* id_candidates - we .* lq_candidates;
    vq_cand = Rs .* iq_candidates + we .* ld_candidates;
    Vmag_cand = sqrt(vd_cand.^2 + vq_cand.^2);

    for it = 1:length(T_grid)
        Tref = T_grid(it);

        idx_feas = find(Vmag_cand <= Vmax);

        if ~isempty(idx_feas)
            % Among feasible points, prioritize torque matching
            [~, idx_sort] = sort(abs(Te_candidates(idx_feas) - Tref));
            idx_sorted = idx_feas(idx_sort);

            % Select the lowest-flux point among the best torque matches
            idx_top = idx_sorted(1:min(20, length(idx_sorted)));

            [~, kbest_local] = min(lambda_mag_candidates(idx_top));
            kbest = idx_top(kbest_local);

            id_fw_lut(iw, it) = id_candidates(kbest);
            iq_fw_lut(iw, it) = iq_candidates(kbest);
        end
    end
end

%% Fill missing entries by interpolation along the torque axis
for iw = 1:length(we_grid)
    valid = ~isnan(id_fw_lut(iw, :));
    if nnz(valid) >= 2
        id_fw_lut(iw, :) = interp1(T_grid(valid), id_fw_lut(iw, valid),
            ...
                                   T_grid, 'linear', 'extrap');
        iq_fw_lut(iw, :) = interp1(T_grid(valid), iq_fw_lut(iw, valid),
            ...
                                   T_grid, 'linear', 'extrap');
    end
end

%% Export lookup tables and parameters to the base workspace
assignin('base', 'T_mtpa_u', T_mtpa_full);
assignin('base', 'id_mtpa_u', id_mtpa_full);
assignin('base', 'iq_mtpa_u', iq_mtpa_full);

assignin('base', 'we_fw_grid', we_grid);
assignin('base', 'T_fw_grid', T_grid);
assignin('base', 'id_fw_lut', id_fw_lut);
assignin('base', 'iq_fw_lut', iq_fw_lut);

```

```
assignin('base', 'Vmax', Vmax);
assignin('base', 'Rs', Rs);
assignin('base', 'p', p);
assignin('base', 'Ts', Ts);

disp('MTPA and FW LUTs generated successfully.');
```

Listing A.2: MATLAB script used to generate the MTPA and flux-weakening lookup tables for the nonlinear map-based implementation.

A.2.2 Preparation of Symmetric Flux, Torque, and Jacobian Tables

```
S = load('data_mapping.mat');
mapping = S.mapping;

%% Original positive-quadrant breakpoints
id_bp_pos = mapping.Id_matrix(1,:);
iq_bp_pos = mapping.Iq_matrix(:,1);

id_bp_pos = id_bp_pos(:)';
iq_bp_pos = iq_bp_pos(:);

%% Symmetric breakpoint vectors
% Zero is not duplicated when mirroring the original axes
id_bp = [-fliplr(id_bp_pos(2:end)), id_bp_pos];
iq_bp = [-flipud(iq_bp_pos(2:end)); iq_bp_pos];

id_bp = id_bp(:)';
iq_bp = iq_bp(:);

%% Original electromagnetic tables on the positive quadrant
% Tables are organized as F(row = iq, col = id)
fluxD_pos = mapping.fluxD;
fluxQ_pos = mapping.fluxQ;
Te_pos    = mapping.Torque_dq;

%% Full-grid tables obtained by symmetry
[IDfull, IQfull] = meshgrid(id_bp, iq_bp);

IDpos = abs(IDfull);
IQpos = abs(IQfull);

% Interpolate the original maps over the mirrored grid
fluxD_abs = interp2(id_bp_pos, iq_bp_pos, fluxD_pos, IDpos, IQpos,
    'linear');
fluxQ_abs = interp2(id_bp_pos, iq_bp_pos, fluxQ_pos, IDpos, IQpos,
    'linear');
Te_abs    = interp2(id_bp_pos, iq_bp_pos, Te_pos,    IDpos, IQpos,
    'linear');

% Replace boundary NaNs with nearest-neighbor interpolation
nan_mask = isnan(fluxD_abs);
if any(nan_mask(:))
    fluxD_abs(nan_mask) = interp2(id_bp_pos, iq_bp_pos, fluxD_pos, ...
                                  IDpos(nan_mask), IQpos(nan_mask),
                                     'nearest');
end
```

```
42
43  nan_mask = isnan(fluxQ_abs);
44  if any(nan_mask(:))
45      fluxQ_abs(nan_mask) = interp2(id_bp_pos, iq_bp_pos, fluxQ_pos, ...
46                                    IDpos(nan_mask), IQpos(nan_mask),
                                         'nearest');
47  end
48
49  nan_mask = isnan(Te_abs);
50  if any(nan_mask(:))
51      Te_abs(nan_mask) = interp2(id_bp_pos, iq_bp_pos, Te_pos, ...
52                                 IDpos(nan_mask), IQpos(nan_mask),
                                      'nearest');
53  end
54
55  % Apply sign symmetries to reconstruct the full tables
56  sd = sign(IDfull);  sd(sd == 0) = 1;
57  sq = sign(IQfull);  sq(sq == 0) = 1;
58
59  fluxD_tab = fluxD_abs .* sd;          % lambda_d odd in i_d
60  fluxQ_tab = fluxQ_abs .* sq;          % lambda_q odd in i_q
61  Te_tab    = Te_abs    .* (sd .* sq);  % torque sign follows sign(i_d
        i_q)
62
63  %% Jacobian entries computed from the full tables
64  did = mean(diff(id_bp));
65  diq = mean(diff(iq_bp));
66
67  [dfluxD_diq, dfluxD_did] = gradient(fluxD_tab, diq, did);
68  [dfluxQ_diq, dfluxQ_did] = gradient(fluxQ_tab, diq, did);
69
70  Ldd_tab = dfluxD_did;   % d(lambda_d)/d(i_d)
71  Ldq_tab = dfluxD_diq;   % d(lambda_d)/d(i_q)
72  Lqd_tab = dfluxQ_did;   % d(lambda_q)/d(i_d)
73  Lqq_tab = dfluxQ_diq;   % d(lambda_q)/d(i_q)
74
75  %% Export full tables to the base workspace for Simulink
76  assignin('base', 'id_bp', id_bp);
77  assignin('base', 'iq_bp', iq_bp);
78
79  assignin('base', 'fluxD_tab', fluxD_tab);
80  assignin('base', 'fluxQ_tab', fluxQ_tab);
81  assignin('base', 'Te_tab', Te_tab);
82
83  assignin('base', 'Ldd_tab', Ldd_tab);
84  assignin('base', 'Ldq_tab', Ldq_tab);
85  assignin('base', 'Lqd_tab', Lqd_tab);
86  assignin('base', 'Lqq_tab', Lqq_tab);
87
88  % Low-pass filter coefficient used elsewhere in the model
89  Ts = evalin('base', 'Ts');
90  tau_dlam = 5 * Ts;
91  alpha = exp(-Ts / tau_dlam);
92  assignin('base', 'alpha', alpha);
93
94  Lmin = 1e-6;
95  assignin('base', 'Lmin', Lmin);
96
```

```
97 disp('Exported full symmetric LUTs and Jacobian tables to base
       workspace.');
```

Listing A.3: MATLAB preprocessing script used to generate the full symmetric flux, torque, and Jacobian lookup tables for the nonlinear map-based synchronous reluctance motor model.