Design and Implementation of Model Predictive Control for Electrical Motor Drives

Saverio Bolognani, *Student Member, IEEE,* Silverio Bolognani, *Member, IEEE,* Luca Peretti, *Student Member, IEEE,* and Mauro Zigliotto, *Member, IEEE*

Abstract

The paper deals with a Model Predictive Control (MPC) algorithm applied to electrical drives. The main contribution is a comprehensive and detailed description of the controller design process that points out the most critical aspects and gives also some practical hints for implementation. As an example, the MPC is developed for a permanent magnet synchronous motor drive. Speed and current controllers are combined together, including all of the state variables of the system, instead of keeping the conventional cascade structure. This way the controller enforces both the current and the voltage limits. Both simulation and experimental results point out the validity of the design procedure and the potentials of the MPC in the electrical drives field.

Index Terms

Model Predictive Control, synchronous motor drives, speed and current control.

I. Introduction

Model predictive control (MPC) derives from a rather old approach whose first ideas have been published more than 20 years ago [1], [2], [3], [4]. Its strategies are based on an explicit and identifiable model of the controlled system, which is used to pre-calculate the behavior of the plant and to choose an optimal value of the control variables.

Because of the computational effort required by MPC, its implementation has been formerly restricted to slowly varying systems, as chemical processes, in which the time step of the discretization is long

enough to allow the complete execution of the control algorithm. As the performance of the available computing hardware has rapidly increased and new faster algorithms have been developed, it is now possible to implement MPC for fast systems with shorter time steps [5].

Electrical drives are of particular interest for the application of MPC for at least two reasons:

1) their quite accurate linear models can be obtained by both analytical means and identification techniques;
2) bounds on drive variables play a key role in the dynamics of the system. Actually, two main approaches are available to deal with systems constraints: the conventional anti–windup techniques, with their manifold variants, widely used in the PI controllers, and MPC. It is worth to note that the difficulty in establishing constraints on the states have also limited the application of conventional state space controllers.

In spite of the mentioned advantages, MPC applications to electrical drives are still largely unexplored and they involve only few research laboratories. For example, generalized predictive control (GPC) – a special case of MPC – has been applied to induction motors first for the current regulation only [6] and, later, for the speed and current control [7]. In [8], the more general MPC solution has been adopted for the design of the current controller in the same drive. In other works [9], [10] model predictive control has been used as a current or a torque/flux controller, directly driving the inverter states.

The main contribution of this paper is a comprehensive and detailed description of the design process of an MPC controller for an electrical drive, pointing out the most critical aspects and giving some practical hints for the design and the implementation, and some suggestions for future studies and developments. As an example of application, the MPC is applied to the control of a permanent magnet synchronous motor (PMSM). Speed and current controllers are combined together in a single MPC that includes all the state variables of the system, instead of keeping the conventional cascade structure. In this way it is possible to enforce all the constraints of the system – namely current and voltage limits – in the controller. Opposite to previous works, the paper also considers the motional coupling effect between the direct and

quadrature axes of the motor.

The proposed approach exploits the main advantages of MPC, i.e. its capability of systematically coping with hard constraints on inputs and states, and its suitability for directly addressing multi-variable systems.

Other issues, that have never been addressed for MPC of electrical drives, are discussed in this paper. These are the null steady-state error in the presence of unknown load and parameters mismatch of the model, and the preservation of stability when a low-pass filter is introduced in the feedback path, especially in the speed measurement.

The paper is organized as follows. Section II provides an overview of the MPC basics. Section III presents the drive model, while Section IV is dedicated to MPC design. The controller implementation is presented in Section V, along with its simulation. A stability analysis is reported in Section VI, and the experimental results are presented in Section VII.

II. Basics of Model Predictive Control

In MPC the controller selects the next input sequence on the basis of the prediction of the future system state behavior [11]. More precisely, it chooses the input signal that minimizes a given cost function of the state. The cost function can be either a $\mathcal{L}_2$ norm of the state, or a $\mathcal{L}_1$ or a $\mathcal{L}_\infty$ norm. The quadratic cost function has been preferred in this paper.

Since the controller has to predict the future system behavior, the core of MPC is the model of the system. Let's then consider a discrete-time state space model in the form

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) \tag{1}$$

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k) \tag{2}$$

where the system variables $\mathbf{x}$, $\mathbf{u}$ and $\mathbf{y}$ satisfy the constraints

$$\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n, \mathbf{y} \in \mathcal{Y} \subset \mathbb{R}^p, \mathbf{u} \in \mathcal{U} \subset \mathbb{R}^m \,.$$

The cost function in its general form is

$$\begin{aligned} J_{N_p} &= \mathbf{x}(k+N_p)^T \mathbf{P} \mathbf{x}(k+N_p) \\ &+ \sum_{j=1}^{N_p} \Big[\mathbf{x}(k+j-1)^T \mathbf{Q} \mathbf{x}(k+j-1) \\ &+ \mathbf{u}(k+j-1)^T \mathbf{R} \mathbf{u}(k+j-1) \Big] \end{aligned} \tag{3}$$

where $\mathbf{Q} = \mathbf{Q}^T \geq 0$ weighs the state vector, $\mathbf{R} = \mathbf{R}^T > 0$ penalizes the control action and $\mathbf{P} = \mathbf{P}^T \geq 0$ weighs the state value at the end of the prediction period N_p.

The problem of finding the best control input for the system then reduces to solving the minimization of (3) subject to

$$\begin{aligned} \mathbf{x}(k+j) &\in \mathcal{X} && j = 1, \dots, N_p \\ \mathbf{u}(k+j) &\in \mathcal{U} && j = 0, \dots, N_p \\ \mathbf{x}(k+j+1) &= \mathbf{A}\mathbf{x}(k+j) + \mathbf{B}\mathbf{u}(k+j) && j = 0, .., N_p - 1 \end{aligned}$$

Expressing the state at step j as the superposition of free and driven response, that is

$$\mathbf{x}(k+j) = \mathbf{A}^j \mathbf{x}(k) + \sum_{h=0}^{j-1} \mathbf{A}^h \mathbf{B} \mathbf{u}(k+j-1-h)$$

the cost function can be transformed in a function of the initial state and of the input sequence only:

$$J'_{N_p}(\mathbf{x}(k)) = \frac{1}{2} \mathbf{x}(k)^T \mathbf{Y} \mathbf{x}(k) + \frac{1}{2} \mathbf{U}^T \mathbf{H} \mathbf{U} + \mathbf{x}(k)^T \mathbf{F} \mathbf{U} \tag{4}$$

where $\mathbf{U} = \left[\mathbf{u}(k)^T, \dots, \mathbf{u}(k+N_u-1)^T \right]^T \in \mathbb{R}^{m \cdot N_u}$ is the vector that contains all the input steps from sampling instant k to sampling instant $k+N_u-1$, where $N_u \leq N_p$ is the control horizon. The input is supposed to be constant after $k+N_u-1$, that is after N_u variations of the control signal.

The constraints too can be rewritten with the only dependence on the input $\mathbf{U}$ and the initial state $\mathbf{x}(k)$,

$$\mathbf{G}\mathbf{U} \leq \mathbf{W} + \mathbf{E}\mathbf{x}(k) \tag{5}$$

The matrices $\mathbf{H} \geq 0$, $\mathbf{F}$, $\mathbf{Y}$, $\mathbf{G}$, $\mathbf{W}$ and $\mathbf{E}$ can be determined from the matrices $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{P}$ in (3) ([12], [13]).

The new cost function (4) and the constraints (5) that have been obtained fit in the class of optimization problems called *Quadratic Programming*, for which efficient iterative solvers are available in the technical literature on non-linear programming and optimization methods (see [12], [13] and references therein). Because of the constraints, the controller does not result to be an analytically determined, linear time–invariant feedback of the state [14]. On the contrary, a convex optimization problem has to be solved online. However, the computational effort of solving this optimization problem at each time step can be greatly reduced, as explained later in Section V.

Once the optimal input sequence has been computed, only the first sample is applied to the plant, according to the *receding horizon policy*. The starting point of the optimal control scheme is periodically updated through feedback and the prediction horizon accordingly shifted (or made to recede) in time, so that the control scheme sees a predicted behavior which is naturally updated to account for the measured evolution of the system.

More precisely, at a given sampling instant k only the first control input $\mathbf{u}^*(k)$ of the open-loop optimal control sequence $\mathbf{U}^*(k)$ is applied to the physical system, which then evolves until the successive sampling instant $k+1$. Based on the newly measured state $\mathbf{x}(k+1)$ the new optimal input $\mathbf{u}^*(k+1)$ is then obtained for the shifted horizon and applied, thus combining state feedback and the optimal open-loop input sequences to effectively close the control loop.

III. Drive Model

In order to exemplify the design of an MPC, the speed and current control of a PMSM drive is developed. To this aim, the drive has to be appropriately modelled. For the electrical subsystem of the drive, a synchronous reference frame (d, q), with the d-axis fixed to the stator PM flux linkage vector is considered, as it lets all the electrical variables be constant at steady-state. The electrical dynamics of the PMSM drive can be thus described by the following stator equations, where the time dependency of the

variables is understood, and iron saturation is assumed negligible:

$$\begin{aligned}\frac{di_d}{dt} &= \frac{1}{L_d}\left(u_d - Ri_d + \omega_{me}L_q i_q\right)\\ \frac{di_q}{dt} &= \frac{1}{L_q}\left(u_q - Ri_q - \omega_{me}L_d i_d - \omega_{me}\Lambda_{mg}\right)\end{aligned} \tag{6}$$

The mechanical dynamics is described by the following equation, derived from the torque balance:

$$\frac{d\omega_{me}}{dt} = \frac{p}{J}\left(k_t i_q - \frac{B}{p}\omega_{me} - \tau_L\right) \tag{7}$$

The motor torque constant is $k_t = 3p\Lambda_{mg}/2$ and $\omega_{me} = p\omega_m$ is the electromechanical speed. Λ_{mg} is the PM flux linkage, p is the number of pole pairs, and ω_m is the mechanical speed. In addition, τ_L is the disturbance torque, while J and B are the moment of inertia and the viscous coefficient of the load.

From (6) and (7), one can realize that the PMSM drive system is described by a nonlinear set of equations, even if electrical and mechanical parameters are constant, because of the motional coupling terms in each axis of (6), involving the speed and the current of the other axis.

In (7), it has been assumed that the motor torque is generated uniquely by the interaction between stator currents and PM flux linkage (PM torque), neglecting the reluctance torque contribution (if any). This means that the motor torque is made proportional to the quadrature current only, while direct current is controlled to zero to minimize Joule losses. This is the case of any motor featuring low or null saliency on the rotor, as in Surface-mounted PM (SPM) motors. However different $d-$ and $q-$ axis inductances are indicated in (6) to account for a low saliency that may appear in the considered motors.

Larger saliency characteristic is exhibited by Interior PM (IPM) motors, and it is generally exploited to increase the delivered torque for a given current level by impressing a negative $d-$ axis current. In such a way, additional non-linearities arise from the nonlinear motor torque equation as well as from the iron saturation, often present in IPM motors.

The main goal of this paper is to introduce the reader to the MPC and to illustrate its potentials in the field of electrical drives. Therefore, the focus will be on the essential hints for MPC design, rather than on illustrating high-complexity applications. A motor with low or null saliency will be considered, with i_d

forced to zero. Nevertheless, the design procedure discussed in the paper is quite general and it contains guidelines to arrange it to a salient PM motor too, accepting of course a more complicated controller.

As regards the design constraints, input voltages are limited by the inverter DC link voltage U_{dc}, which induces an hexagonal voltage boundary rotating in the d-q voltage plane with angular speed ω_{me}. For the sake of simplicity, the boundary can be approximated by the circle with radius $U_{dc}/\sqrt{3}$, inscribed in the hexagon and invariant under the stationary-to-synchronous coordinates transformation. When more convenient, an alternative approximate boundary may be a polygonal region fixed in the d-q reference frame.

A further constraint concerns the current level, mainly for motor and inverter thermal limits. Since the i_d current is regulated to zero, current limitation actually sets the maximum amplitude of the i_q current.

IV. MPC DESIGN OF A PMSM DRIVE

A. *Definition of the MPC state variables*

The first step in the design of a model predictive controller consists in determining a discrete–time model for the system.

MPC can be best implemented for the class of systems that accepts a representation by a linear model with constraints, because in that case most of the optimization process can be moved off–line (see Section V). The linear model has to be pursued by an appropriate choice of the state variables.

As pointed out in Section III, the most important non–linearity of the PMSM drive is given by the coupling terms involving the speed and the currents i_d and i_q. In particular, the effect of the term $\omega_{me} i_q$ on the state equation for i_d cannot be ignored, while the term $\omega_{me} i_d$ in the i_q equation has generally little or no effect, once the direct current i_d is kept null. In principle, it could be possible to decouple i_d and i_q acting on the MPC output signals u_d and u_q, but then the voltage constraints in the model would not reflect the actual ones, because of the additional decoupling terms added outside. As both ω_{me} and i_q can be available for measurement in the drive, the $\omega_{me} i_q$ term will then be considered as a measured disturbance and it is included in the model.

The state variables of the system are then the currents i_d and i_q and the angular speed ω_{me}, along with the measured disturbance $\widehat{\omega_{me} i_q}$.

$$\mathbf{x} = \begin{bmatrix} i_d & i_q & \widehat{\omega_{me} i_q} & \omega_{me} \end{bmatrix}^T$$

The input signal is the two dimensional voltage vector

$$\mathbf{u} = \begin{bmatrix} u_d & u_q \end{bmatrix}^T .$$

According to this choice, the model for the motor is then

$$\dot{\mathbf{x}} = \begin{bmatrix} -\frac{R}{L_d} & 0 & \frac{L_q}{L_d} & 0 \\ 0 & -\frac{R}{L_q} & 0 & -\frac{\Lambda_{mg}}{L_q} \\ 0 & 0 & 0 & 0 \\ 0 & p\frac{k_t}{J} & 0 & -\frac{B}{J} \end{bmatrix} \mathbf{x} + \begin{bmatrix} \frac{1}{L_d} & 0 \\ 0 & \frac{1}{L_q} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \mathbf{u}$$

The dynamics of the measured disturbance $\widehat{\omega_{me} i_q}$ has been linearised around $i_q = 0$ and $\omega_{me} = 0$, therefore having $\dot{\widehat{\omega_{me} i_q}} = 0$ (i.e. a constant disturbance in the prediction horizon, measured at the beginning of the prediction process). This is the most effective choice to obtain a single linear model for the drive in all its operating conditions. Adopting variable structure modelling to account for different linearization points does not result in significant improvement of the performances, as simulations and experimental results have confirmed.

Load torque is treated as an unpredictable disturbance. Its effect on drive behavior and the related MPC rejection capability will be illustrated later in the paper.

B. Discrete time model

As said, MPC design requires a discrete–time model. Since it has to deal with both current and speed dynamics (characterised by quite different time constants), for a correct approximation of the fastest dynamics the **time step** has to be small enough with respect to the smallest time constant, which is the electrical one. The main drawback of a very small time step is that the coefficients that describe the slow dynamics associated to the speed are extremely close to zero or to 1, with numerical problems. Besides,

very small control cycles may cause an excessive microprocessor load factor and the risk of overflow. A time step $T = 1/(12\text{kHz}) = 83.3\mu s$ has been chosen in this work.

The resulting discrete time model (obtained with forward Euler discretisation) is the following

$$\mathbf{x}(k+1) = \begin{bmatrix} 1 - T\frac{R}{L_d} & 0 & T\frac{L_q}{L_d} & 0 \\ 0 & 1 - T\frac{R}{L_q} & 0 & -T\frac{\Lambda_{mg}}{L_q} \\ 0 & 0 & 1 & 0 \\ 0 & T\frac{pk_t}{J} & 0 & 1 - T\frac{B}{J} \end{bmatrix} \mathbf{x}(k) + \begin{bmatrix} \frac{T}{L_d} & 0 \\ 0 & \frac{T}{L_q} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \mathbf{u}(k) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) \tag{8}$$

C. State augmentation for tracking

As the objective of the design is **tracking** and not driving the system to zero, some modifications are needed to the state variables. First of all, to be able to weigh the speed error in the cost function, the reference ω_{me}^{ref} has to appear as a state variable of the system. As information about the future reference is not generally available, its state equation is

$$\omega_{me}^{ref}(k+1) = \omega_{me}^{ref}(k) \tag{9}$$

Since the cost function penalizes the control signal too, we would not be able to achieve offset–free tracking, because the controller would apply a control signal smaller than the one that results in zero error.

There are two main ways one can deal with this issue. The first is to weigh the error $u - u^{ref}$ instead of u in the cost function, where u^{ref} is the input signal that produces the given output reference. This requires the a priori knowledge of u^{ref}, that is highly sensitive to the model parameters. The other approach consists

in weighing the input variation at each time step instead of its value. This way the input is allowed to reach the required level that produces the desired output, without worsening the cost function.

This latter solution is implemented by transforming the input description in its **differential form**:

$$u_d(k) = u_d(k-1) + \Delta u_d(k)$$

$$u_q(k) = u_q(k-1) + \Delta u_q(k)$$

Two additional state variables, the components $u_d^{k-1} = u_d(k-1)$ and $u_q^{k-1} = u_q(k-1)$, are thus introduced. The new augmented state vector is

$$\mathbf{x}_{TR}(k) = \begin{bmatrix} i_d & i_q & \widehat{\omega_{me} i_q} & \omega_{me} & \omega_{me}^{ref} & u_d^{k-1} & u_q^{k-1} \end{bmatrix}^T$$

and the input vector becomes

$$\boldsymbol{\Delta}\mathbf{u}(k) = \begin{bmatrix} \Delta u_d & \Delta u_q \end{bmatrix}^T$$

The new system has the following state update equation

$$\mathbf{x}_{TR}(k+1) = \begin{bmatrix} \mathbf{A} & 0 & \mathbf{B} \\ 0 & 1 & 0 \\ 0 & 0 & \mathbf{I} \end{bmatrix} \mathbf{x}_{TR}(k) + \begin{bmatrix} \mathbf{B} \\ 0 \\ \mathbf{I} \end{bmatrix} \boldsymbol{\Delta}\mathbf{u}(k)$$

where $\mathbf{A}$ and $\mathbf{B}$ are the matrices that appear in (8).

A slight modification leads to a more accurate system model. As the control input $\boldsymbol{\Delta}\mathbf{u}$ will not drive the inverter before the next time step, a **1–step delay** should be included in the model. This usually requires the introduction of a new state variable for each input that has to be delayed. Conversely, in the augmented form that has been obtained for tracking, it is enough to remove the dependency of $\mathbf{x}_{TR}(k+1)$ on the input $\boldsymbol{\Delta}\mathbf{u}(k)$, getting

$$\mathbf{x}_{TR}(k+1) = \begin{bmatrix} \mathbf{A} & 0 & \mathbf{B} \\ 0 & 1 & 0 \\ 0 & 0 & \mathbf{I} \end{bmatrix} \mathbf{x}_{TR}(k) + \begin{bmatrix} 0 \\ 0 \\ \mathbf{I} \end{bmatrix} \boldsymbol{\Delta}\mathbf{u}(k)$$

D. Cost function and control law parameters

Once that a discrete–time model for the system is available, the next important step is the definition of the **cost function** that shall be minimized by an appropriate choice of the control inputs at each time step. As pointed out in Section II, the cost function is a quadratic expression of the state in the form (3). In the case analysed in this work, the main quantity to be weighted is the speed error $\omega_{me}^{ref} - \omega_{me}$; as i_d has to be zero, its value will be included in the cost function as well. Moreover, as the cost matrix for the system input signals has to be positive definite, a non-zero weight has to be chosen for the inputs Δu_d and Δu_q. Another choice concerns the weight for i_q. There is no real reason to penalize a large value of i_q, especially when the goal is to achieve reference tracking in presence of possible load disturbances. However, choosing a small, non-zero value for the weight of i_q plays a key role in the stability of the whole closed-loop system. This will be confirmed in the stability analysis in Section VI.

The matrices $\mathbf{Q}$ and $\mathbf{R}$ (according to the notation used in (3)) are then

$$\mathbf{Q} = \begin{bmatrix} \gamma_{i_d} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \gamma_{i_q} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \gamma_{w_{me}} & -\gamma_{w_{me}} & 0 & 0 \\ 0 & 0 & 0 & -\gamma_{w_{me}} & \gamma_{w_{me}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{10}$$

$$\mathbf{R} = \begin{bmatrix} \gamma_u & 0 \\ 0 & \gamma_u \end{bmatrix} \tag{11}$$

where the γs are the cost coefficients. Matrix $\mathbf{P}$ is a constant. Its tuning would represent a further degree of freedom that has not been exploited in this work, choosing $\mathbf{P} = 0$.

The parameters have been set to the following values by virtual prototyping and simulations:

$$\gamma_{i_d} = 100, \quad \gamma_{i_q} = 1, \quad \gamma_{\omega_{me}} = 30, \quad \gamma_u = 0.8$$

Another important parameter of the cost function is the length N_p of the **prediction horizon**. Ideally, it has to be chosen long enough to include the most relevant part of the system dynamic. Unfortunately, due to computational reasons, it is not possible to include the whole speed dynamics in the prediction horizon, as it would result in hundreds or thousands of time steps (while the currents dynamics is fully described in few time steps). For this reason a compromise has to be found, keeping in mind that a lower limit exists for the length of the prediction horizon. Indeed, it has to be noted that after discretisation the whole state space is not reachable in one time step. In other words, the effects of a change in the input signals $\mathbf{\Delta u}$ at the time step k will have effect on the voltage $\mathbf{u}$ at the time step $k+1$, on the current $\mathbf{i}$ at time $k+2$, and on the speed ω_{me} at time $k+3$. This sets a minimum value $N_p = 4$ for the prediction horizon. In the implementation presented in this paper, it is $N_p = 5$.

There is still another parameter, not related to the cost function, but rather to the control law. It is the **control horizon** N_u, that is the number of steps after which the input signal is considered steady when predicting the future response of the system. The only reason for the parameter N_u is the simplification of the optimization process, for the sake of offline computational time. On the other hand, it results in achieving a sub–optimal solution instead of the optimal one (even if simulation and experimental results showed that this sub–optimality is not critical at all).

In the implementation of this paper it has been choosen $N_u = 1$. It means to consider a constant control input when predicting the system behavior. This may seem limiting; note however that for the same reasons that have been pointed out above, any input signal after $k+1$, where k is the sampling instant when the algorithm is executed, would have no effect on the main state variable we are interested in (the speed ω_{me}) before $k+5$, that is inside the prediction horizon $N_p = 5$.

E. State variable bounds

One of the most important features of MPC is its ability in dealing with bounds on the state variables. To exploit this capability an accurate set of bounds have to be included in the model of the system. The two most important bounds in a PMSM drive are the **current** and the **voltage limits**. They are two upper

limits on the magnitude of $\mathbf{i}$ and $\mathbf{u}$, that correspond to a circular limit in the d-q plane. However, as linear limits are highly preferable for implementation purposes, two polygonal approximations of those limits have been adopted.

As regards the current limit, a rectangular region has been enforced, corresponding to independent limits on i_d and i_q.

$$i_d \in [-\epsilon I_N \quad \epsilon I_N]$$

$$i_q \in [-I_N \quad I_N]$$

where I_N is the nominal stator current of the drive and ϵ is a coefficient smaller than 1 (for example $\epsilon = 0.2$). As i_d is kept close to zero, this approximation is somehow acceptable.

As regards the voltage limit, a finer approximation has been used instead, because both u_d and u_q can be far from zero. The circular region $|\mathbf{u}| \leq U_N$ has been converted in an octagon $\mathcal{P}_8$, therefore enforcing eight linear constraints. The octagon is inscribed into the circular limit region.

$$\mathbf{u} \in \mathcal{P}_8 \subset \{\mathbf{u} : |\mathbf{u}| \leq U_N\}$$

Particular attention has to be paid in enforcing these constraints on the present and future steps inside the prediction horizon. While the voltage corresponds to the control input and therefore it is chosen by the controller itself, the current value comes from a measurement on the real currents. Due to overshoots and measurement noise, the current at sampling time k can lie outside the given region. However, as the control input applied at sampling time k has no effect on the currents until the instant $k + 2$, **current limits** on the two steps k and $k + 1$ must be **removed**.

F. Integral action for offset–free tracking

A typical problem of MPC is the way it deals with **unmeasured disturbances and parameter mismatch** in the system model. In fact, as the control input applied to the system depends on the current state only, and not on the past history of the system, there is no guarantee of steady–state zero offset in case of either disturbances or inexact system modelling. This issue is addressed in the MPC literature

using the information coming from the comparison between the predicted and the actual system output, together with some models of the disturbances acting on the plant [15], [16], or adding some sort of robustness to model predictive control [17], [18].

In the present case, a simpler solution has been adopted, as prediction errors are not available because the optimization is perfomed offline (see Section V). The proposed solution consists in adding an integrator of the angular speed error $\omega_{me}^{ref} - \omega_{me}$ outside of the controller, the output of which is used to move the reference ω_{me}^{ref}. In other words, instead of feeding the controller with the speed reference provided by the user, it is fed by

$$\bar{\omega}_{me}^{ref} = \omega_{me}^{ref} + K_{INT} \int \left(\omega_{me}^{ref} - \omega_{me}\right) \text{ dt}$$

Avoiding the problem of **integrator wind-up** (that has to be faced with this approach) is particularly easy as the MPC controller algorithm (see Section V) automatically returns a flag if any of the constraints is active. If one or more of the constraints are active, the integration of the error is not executed for that single time step.

This solution can also be interpreted as modelling all the disturbances in one additional signal on the angular speed ω_{me} and estimating this disturbance by integrating the signal ω_{me} and its reference. The choice of K_{INT} is not particularly challenging. In this paper, a value of $K_{INT} = 20$ let the steady-state speed error be eliminated, with little influence on the system dynamic.

G. Presence of filters in the feedback

In the practical implementation of a speed controller, a **low-pass filter in the speed feedback** may appear. This can be introduced by the designer, either to filter out high–frequency noise from the signal coming from the resolver/encoder, or to account for some unavoidable effects like time delays in the measurement. The presence of an additional pole in the gain loop has one main consequence: it worsens the dynamic behavior of the closed-loop systems, even compromising the closed-loop stability.

It is quite easy to deal with this problem in the MPC, by introducing an additional state in the controller,

namely the filtered speed ω_{me}^f governed by

$$\dot{\omega}_{me}^f = -\frac{1}{\tau_f}\omega_{me}^f + \frac{1}{\tau_f}\omega_{me}$$

where τ_f is the time constant of the low-pass filter. In the cost function, weighting the error $\omega_{me}^{ref} - \omega_{me}^f$ instead of $\omega_{me}^{ref} - \omega_{me}$ allows the designer to choose large time constants in the feedback filter without compromising the stability of the closed-loop system. Experimental tests (not reported in this paper) confirmed the effectiveness of this approach.

The only drawback is the augmented state space, which implies a more complex algorithm and longer computational time. Nevertheless, it is a solution one should keep in mind when feedback filtering is somehow unavoidable.

V. Implementation and algorithm

So far, MPC has been mainly used in slow process control, due to its large computational demand, needed to seek the future control actions that minimize a given cost function. Previous approaches tried to solve this optimization problem analytically. These solutions (GPC, for example) have the unacceptable drawback of making the inclusion of constraints difficult, when not impossible at all [14]. Consequently, the development and implementation of the control algorithm is crucial to the application of MPC to fast dynamic systems.

In [8], most of the optimization problem is solved off-line, thus keeping the complexity of the on-line part of the algorithm rather low. This is possible by computing an explicit solution of the control law [12]. Indeed, in the case of a linear system with constraints and a quadratic cost function of the state, the **optimization problem** of finding the control values that minimize the given cost function is a *multi–parametric quadratic program*. Moreover, the control law is a piecewise-linear and continuous state feedback, defined on a finite number of contiguous regions. More precisely, in each region of the state space the control law takes the form

$$\mathbf{u}_i(k) = \mathbf{F}_i\,\mathbf{x}(k) + \mathbf{G}_i \tag{12}$$

where i is the index of the *active region*, that is the convex region in which the state is at the sampling time k, and the matrices $\mathbf{F}_i$ and $\mathbf{G}_i$ are the result of the optimization algorithm described in [12].

Fig. 1 shows an example of regions partition of the controller. From the figure one can see the main hyperplane that divides positive from negative speed errors, and the regions that appear as the state get closer to the current limit (6 A in this example). These regions automatically enforce the current limit of the controller.

Applying (12) to the system means to search the region in which the system state is located and then evaluating the affine feedback law in the form (12), to obtain the input to be applied to the system. The online part of the algorithm consists of a region search and a matrix multiplication and sum, as one can see in Fig. 2. The step of determining in which region the system lies may be computationally intensive, in the presence of numerous regions. The complexity can be reduced implementing a dichotomic search among the regions. Instead of verifying whether the state is within a region, a **binary search tree** is built [19]. As the regions aren't ordered in any way, the search tree consists in determining a series of hyperplanes that severs the contiguous regions, and then creating a tree of comparison of the current state with each of these hyperplanes. The result of each comparison marks out the current state as "above" or "below" the hyperplane. Hence, each region (which is a leaf of the tree) can be identified by the result of the series of comparison from the root of the tree to the leaf itself, passing through the intermediate nodes forming the branches of the binary tree.

Actually, the aforementioned dichotomic method moves offline the most computationally expensive part of the search of the active region, leaving online only a short series of tests to locate the current state among the hyperplanes and the computation of the control input based on the affine feedback law returned by the active region. This makes it possible to execute the MPC algorithm fast enough to fit in the typical time steps of the electrical drives.

Moreover, as the problem fits in the general frame of multi parametric programming, efficient tools are available to solve it explicitly. In particular, the *Multi–Parametric Toolbox* (MPT) [13] for Matlab

provides both a solver for the optimization problem and some routines to generate the region partition and the binary search tree for its simulation and implementation.

The developed MPC algorithm has been widely simulated in different operating conditions, before its actual implementation. MPT allows both the simulation in Matlab and the implementation in C code. One of the simulation results is in Fig. 3, which shows the response of the drive to a pulse speed reference from 500 rpm to 1000 rpm. The parameters of the simulated drive are those of the test-bed that has been later used for experimental validation, and they are reported in Section VII. The MPC has been designed to enforce a current limit of 6 A and a voltage limit of 173 V (DC bus at 300 V).

One can note the good speed response with limited overshoot and without steady-state error. Acceleration and deceleration are limited by the current limit that affects i_q, while i_d is kept close to zero. Voltage limits are not active in this operating conditions. Several feedback laws are used during the transients, as one can see from the indexes of the active regions in the different moments of the response.

Simulations like that of Fig. 3 have been used to **tune the MPC** and to evaluate its complexity, because analytical means are not easily available.

VI. Stability analysis

Generally, research studies about MPC closed-loop stability assume that the prediction horizon goes to ∞. Unfortunately, this hypothesis does not hold in the present application, since the prediction horizon is far from including most of the future dynamics of the motor.

Alternatively, two main directions can be investigated: **Lyapunov analysis** (only briefly sketched) and **linearization** (hereafter used).

A. Lyapunov analysis

Global stability of the closed-loop system can be investigated by the use of an appropriate Lyapunov function. Indeed, the control law is the piecewise affine feedback (12). That means that the closed-loop

system has a similar piecewise structure in the form

$$\mathbf{x}(k+1) = \alpha_i \, \mathbf{x}(k) + \beta_i \, ,$$

where α_i and β_i result from algebraic manipulations on $\mathbf{F}_i$, $\mathbf{G}_i$ and the system matrices $\mathbf{A}$ and $\mathbf{B}$. As it has been proved in [20], for such an autonomous system it is possible to seek an appropriate Lyapunov function

$$V = \mathbf{x}^T \mathbf{L}_i \mathbf{x}$$

that satisfies the requirements of being positive and decreasing along the trajectories of the system. The matrices $\mathbf{L}_i$ can be identically equal to a matrix $\mathbf{L}$, then resulting in a quadratic Lyapunov function, or they can be different constant matrices for each i–th region. In the latter case, V would result in a piecewise quadratic function, and its existance is a sufficient condition to prove exponential stability of the system [20]. A complete Lyapunov analysis has not been completed in this paper, but the software toolbox MPT [13] offers some routines to search a suitable Lyapunov function of the desired form.

B. Linearization

The first approach consists in linearising the system around an equilibrium point, for small variations of the state. This approach is simplified by the fact that the controller results to be piecewise affine in a finite number of regions in the state space. In particular, there is a main region (the one in which none of the constraints is active) in which the control law is a linear feedback of the state.

Supposing that the system is laying in that region, the stability can be investigated by the classical methods that apply for linear systems. For example, one can be interested in checking the position of the closed-loop poles when varying the weights in the cost function, which are the main tuning knobs for MPC. As the systems has dimension $n = 7$, 7 poles appear in the map. It can be verified that two of them are independent on the weights and located in $z = 1$ (therefore representing constant modes). One can intuitively suppose that these modes refer to the measured disturbance $\widehat{\omega_{me} i_q}$ (see (8)) and to the speed reference ω_{me}^{ref} (see (9)). Another pole is located close to 1, slightly inside the stable unit circle.

The remaining four poles are two complex conjugate couples. These, together with the (just mentioned) real one, model the electro–mechanical dynamic of the system.

In Fig. 4 the position of the poles is plotted when the speed error weight in the cost function is varied around the nominal one. The main effect occurs on the real pole close to 1, that moves towards the origin as the weight $\gamma_{\omega_{me}}$ increases. This means that a higher weight corresponds to faster dynamics of the system. However, one should keep in mind that this analysis does not include the effects of state variable bounds and it is valid only for small variations of the states, which leaves the state within the main linear region, where the linearization has been performed. This can be easily seen by simulation.

In Fig. 5, instead, the effects of modifying the weight of the current i_q is analysed. With $\gamma_{i_q} = 0$, one of the complex conjugate pole pair is outside of the unit circle, and as the i_q weight increases, they move inside. As observed in Section IV when choosing the cost function parameters, a non–zero value of γ_{i_q} is therefore needed to obtain the closed-loop stability. What one would observe if the weight was set to zero, is that the system would not stay in the main linear region, switching instead between the upper and the lower current limits (a sort of bang–bang operation mode). On the other hand, a too high γ_{i_q} causes the real pole to move very close to 1, slowing down the speed response of the whole system.

Finally, in Fig. 6, the position of the poles is plotted for different values of the common weight of Δu_d and Δu_q. The optimization algorithm needs a non–zero value for these weights γ_u. It is worth to note that their decrease moves one of the complex pair of poles towards the origin (therefore improving stability and dynamic response), so that the smallest value that does not involve numerical problems has been chosen in the design.

VII. Experimental results

The designed MPC for a PMSM drive has been experimentally validated on a laboratory testbench, sketched in Fig. 7. The testbench uses an SPM motor supplied by a three-phase inverter, controlled by a Fast Control Prototyping (FCP) board. A vector-controlled induction motor drive is used as mechanical load. The whole digital control of the SPM motor drive is implemented in the FCP board, from the PWM

generation to the speed/current model predictive control. The PWM and sampling frequencies are both 12 kHz. Main drive data are reported in Table I.

Table I

MAIN DRIVE DATA

Nominal torque	13.8	Nm
Nominal current	8.5	A rms
Nominal speed	2160	rpm
Phase resistance	0.8	Ω
Phase inductance ($L_d = L_q$)	$6.5 \cdot 10^{-3}$	H
Pole pairs	3	
Total moment of inertia	$8.2 \cdot 10^{-3}$	kg m^2
DC bus voltage	300	V

Some experimental results are hereafter reported. At first the drive has been tested in the same conditions of the simulation reported in Fig. 3. The experimental results are reported in Fig. 8. As in the simulation, an i_q limit of 50% of the nominal current ($8.5\sqrt{2} = 12$ A peak) has been enforced to highlight its effects on the dynamic. One can note the good overall performance of the drive and the tight correspondance between the experimental and simulated results (apart from the chattering mainly due to the inverter dead–times). The reliability of simulation in predicting MPC behavior is thus confirmed.

For the sake of comparison, the PMSM motor drive has been also designed and tested using three PI controllers for the speed and current loops. $d-$ and $q-$ current controllers have been designed for a bandwidth in between the inverse of the electric time constant and the inverse of the sampling time, while a bandwidth a decade lower has been chosen for the speed controller. Experimental results are in Fig. 9. Fig. 8 and Fig. 9 show the similar behavior of the MPC and the PI controllers within the nominal speed of the motor. The PI controllers are equipped with anti-windup algorithms, which represent the most critical design issue as the output of the current controllers (the d- and q- voltages) are the components of a limited space vector and are not limited independently. Managing voltage limitations for current control loops is the greatest disadvantage of PI solutions, while MPC has an inherent capability in operating under

limited conditions. The slightly higher noise superimposed to currents and voltages in the MPC version can be reduced by a proper modification of the weights in the MPC controller, which is comparable to changing a sort of proportional contribution.

The robustness of the controller to parametric mismatch has been briefly explored in one of its most common occurences, that is a wrong assumption about the moment of inertia of the load. Fig. 10 shows the behavior of the controller when it is designed for a moment of inertia 3 times greater than the actual one. With respect to Fig. 9, a larger overshoot appears in the speed response, but the behavior is still stable and acceptable.

In Fig. 11 a speed reference higher than the nominal is imposed, to force the intervention of the voltage limit. This test points out the impressive capability of the MPC to control both current and voltage limits. In particular, the voltage limit does not allow the reaching of the maximum speed reference.

As it can be seen, a sensible chattering is present as the voltages approach their limit. Voltage chattering mirrors the chattering in the selection of the MPC active region, which occurs as the drive transits from a non-limited to a limited condition operating point. The reduction of the chattering through a smoother, or even hysteretical transition between MPC active regions is still an open point which should be further investigated in future works.

In Fig. 12 and 13 a triangular speed reference is commanded. In the former, the slope is chosen so that the required torque current (i_q=3 A) is beyond its limit. Therefore, there is room left for speed error recovery, as testified by the current overshoots that occur after every acceleration reversal. In the latter, the slope is twice larger, so that a 6 A i_q is needed. As this value coincides with the enforced current limit, therefore no overshoot is allowed and the controller achieve the best possible result while respecting that limit. The speed error cannot be completely cancelled during acceleration.

The last set of experiments deals with the drive rejection capabilities to torque disturbances. It is worth to recall that torque disturbance is not present in the state model used to design the MPC. The key role in the zero–offset rejection of this kind of disturbances is played by the external integral action described in

Section IV. In these tests the current limit has been set to its nominal value (12 A) to operate the motor up to its full torque.

In Fig. 14 a constant speed reference of 800 rpm is commanded, while the torque load varies from 20% to 40%, and then 20% again, of the nominal torque (13.8 Nm). The torque load is obtained by commanding the induction motor drive connected to the drive under test. The torque measured by the torquemeter shows a slightly larger value due to additional friction at this speed. The figure shows a good capability of the MPC to limit the torque disturbance effects and to achieve zero steady-state speed error. The maximum transient speed error is within 1.5% of the nominal drive speed.

In Fig. 15 the test is carried out at 100 rpm, i.e. less than 5 % of the nominal speed. At $t = 0$, the application of a rated load torque step activates the current limitation. The settling time is then longer at load torque onset, while lower at load detach. The speed error is higher than that exhibited in Fig. 14, proportionally to the torque step amplitude.

As can be seen, the superimposed noise of Fig. 14 is higher than that of Fig. 15. As a matter of fact, the speed sensor which has been used for the experimental work is an encoder with a resolution of 1000 pulses per revolution. The mechanical speed has been obtained measuring the period of the pulse waveform coming from the encoder. It is well known that this measurement mode suffers of accuracy problems as the speed increases, as in our case. An industrial practice, which involves some hardware modifications, consists in toggling between period and frequency measurements of the encoder pulses, for low and high speeds respectively.

The experimental results presented in this chapter, together with all the others carried out in the development of the work, and the simulation results as well, point out the promising features and characteristics of MPC applied to electrical motor drives. These potentials stimulate further exploration and study on this type of controllers in order to achieve the familiarity required to transfer them to practical applications.

VIII. Conclusions

By using a permanent magnet synchronous motor drive as a test bench, the paper gives an exhaustive description of the design procedure of a Model Predictive Control (MPC) applied to the combined control of the motor speed and current. After a brief introduction of the MPC fundamentals, the design is illustrated in detail giving a step-by-step discussion of the main critical points and the hints for their successful handling. Suggestions for extending the design to different drive controllers are also included. Simulations, as well as numerous experimental results, highlight the promising features and characteristics of MPC applied to electrical motor drives. As a last contribution, the MPC potentials pointed out in the paper should stimulate further exploration and study on this type of controllers, in order to achieve the familiarity required to transfer the results to practical applications.

References

[1] D. W. Clarke, C. Mohtadi, and P. S. Tuffs, "Generalized predictive control – part I. the basic algorithm," *Automatica*, vol. 23, pp. 137–148, 1987.

[2] ——, "Generalized predictive control – part II. extensions and interpretations," *Automatica*, vol. 23, pp. 149–160, 1987.

[3] D. W. Clarke and C. Mohtadi, "Properties of generalized predictive control," *Automatica*, vol. 25, pp. 859–875, 1989.

[4] C. E. Garcia, D. M. Prett, and M. Morari, "Model Predictive Control: Theory and practice – a survey," *Automatica*, vol. 25, pp. 335–348, 1989.

[5] M. Morari and J. H. Lee, "Model Predictive Control: Past, present, and future," *Computers and Chemical Engineering*, vol. 23, no. 4–5, pp. 667–682, 1999.

[6] L. Zhang, R. Norman, and W. Shepherd, "Long–Range Predictive Control of current regulated PWM for induction motor drives using the synchronous reference frame," *IEEE Trans. Control Syst. Technol.*, vol. 5, no. 1, pp. 119–126, 1997.

[7] R. Kennel, A. Linder, and M. Linke, "Generalized predictive control (GPC) – ready for use in drive applications?" in *IEEE Power Electronics Specialists Conference*, 2001, pp. 1839–1844.

[8] A. Linder and R. Kennel, "Model predictive control for electrical drives," in *IEEE Power Electronics Specialists Conference*, 2005, pp. 1793–1799.

[9] ——, "Direct model predictive control – a new direct predictive control strategy for electrical drives," in *European Conference on Power Electronics and Applications*, Sep. 2005.

[10] R. Morales and M. Pacas, "Predictive torque and flux control for the synchronous reluctance machine," *Bulletin of the Polish Academy of Sciences*, vol. 54, no. 3, pp. 271–277, 2006.

[11] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[12] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit solution of model predictive control via multiparametric quadratic programming," in *Proc. of the American Control Conference*, Chicago, IL, 2000, pp. 872–876.

[13] M. Kvasnica, P. Grieder, and M. Baotić, "Multi-Parametric Toolbox (MPT)," 2004. [Online]. Available: http://control.ee.ethz.ch/ mpt/

[14] J. M. Maciejowski, *Predictive Control with Constraints*. Prentice Hall, 2002.

[15] K. R. Muske and T. A. Badgwell, "Disturbance modeling for offset-free linear model predictive control," *J. of Process Control*, vol. 12, no. 5, pp. 617–632, 2002.

[16] A. Faanes and S. Skogestad, "Offset–free tracking with MPC with model mismatch: Experimental results," *Industrial and Engineering Chemistry Research*, vol. 44, no. 11, pp. 3966–3972, 2005.

[17] G. Pannocchia and E. C. Kerrigan, "Offset-free receding horizon control of constrained linear systems," *AIChE Journal*, vol. 51, no. 12, pp. 3134–3146, 2005.

[18] G. Pannocchia, "Robust model predictive control with guaranteed setpoint tracking," *J. of Process Control*, vol. 14, no. 8, pp. 927–937, 2004.

[19] P. Tøndel, T. A. Johansen, and A. Bemporad, "Evaluation of piecewise affine control via binary search tree," *Automatica*, vol. 39, pp. 945–950, 2003.

[20] G. Ferrari-Trecate, F. A. Cuzzola, D. Mignone, and M. Morari, "Analysis of discrete–time piecewise affine and hybrid systems," *Automatica*, vol. 38, pp. 2139–2146, 2002.

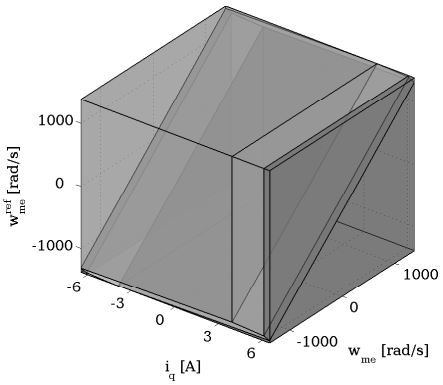

Figure 1. An example of region partition of the controller. As the dimension of the state is greater than 3, projection has been applied on the state variables other than the current i_q, the velocity ω_{me} and the reference ω_{me}^{ref}.

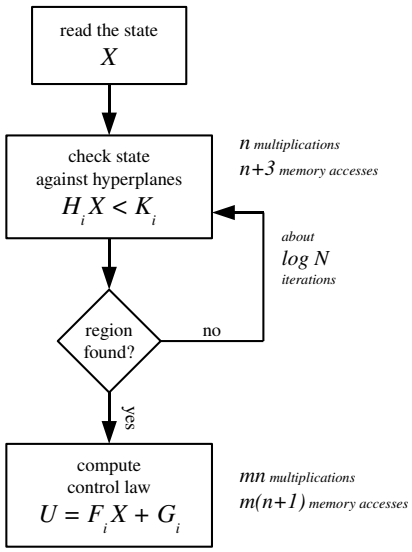

Figure 2. Flow chart of the control algorithms with a rough estimation of the computational complexity of each step. N is the number of regions, n is the dimension of the state and m is the dimension of the input vector. In the experimental setup described in this paper we have $N = 80$, $n = 7$, $m = 2$.

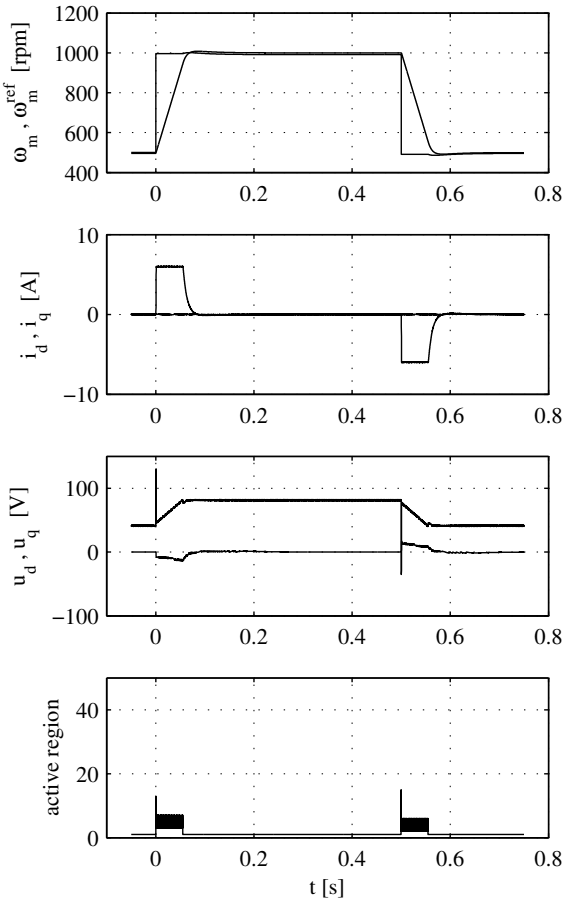

Figure 3. Simulated drive response to a 500–to–1000 rpm reference speed pulse. From top to bottom: speed reference and response, $d-$ and $q-$ currents, $d-$ and $q-$ voltages, and active region number of the MPC controller.

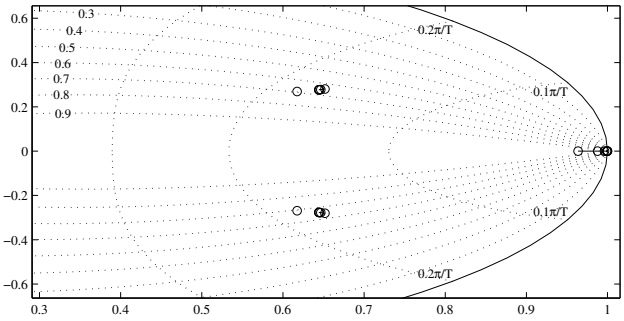

Figure 4. Pole map for speed error weight $\gamma_{\omega_{me}} = 3,\ 10,\ 30,\ 100,\ 300$.

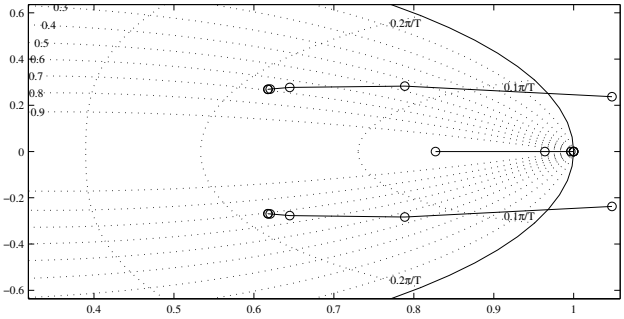

Figure 5. Pole map for quadrature current weight $\gamma_{i_q} = 0,\ 0.1,\ 1,\ 10$.

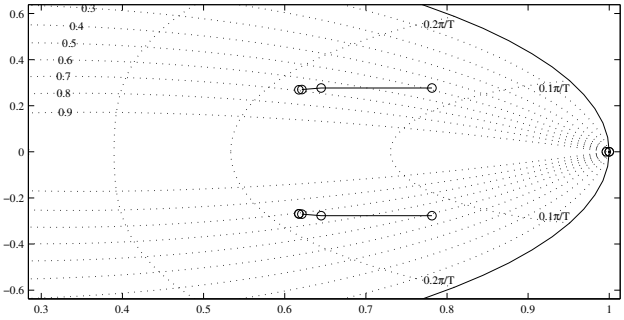

Figure 6. Pole map for differential input voltage weigth $\gamma_u = 0.08,\ 0.8,\ 8$.

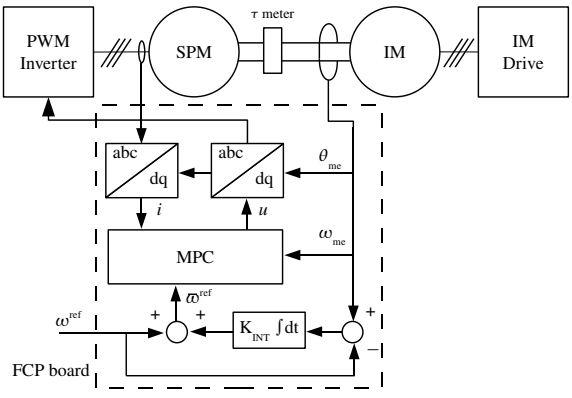

Figure 7. Schematic of the laboratory test bench.

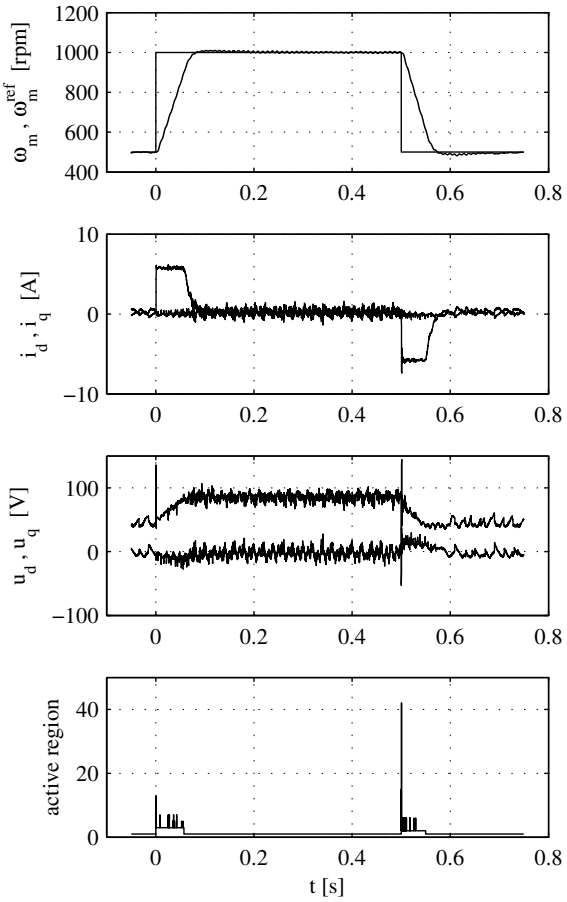

Figure 8. Experimental drive response to a 500–to–1000 rpm reference speed pulse. From top to bottom: speed reference and response, $d-$ and $q-$ currents, $d-$ and $q-$ voltages, and active region number of the MPC controller.

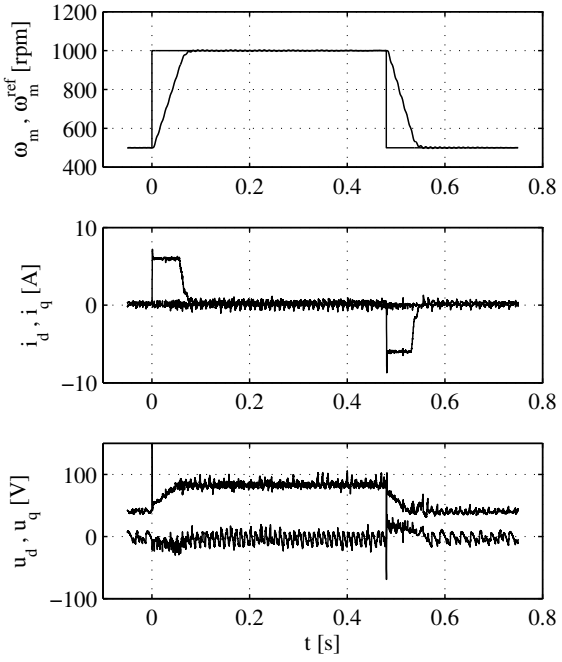

Figure 9. Experimental drive response to a 500–to–1000 rpm reference speed pulse with PI controllers. From top to bottom: speed reference and response, $d-$ and $q-$ currents, $d-$ and $q-$ voltages.

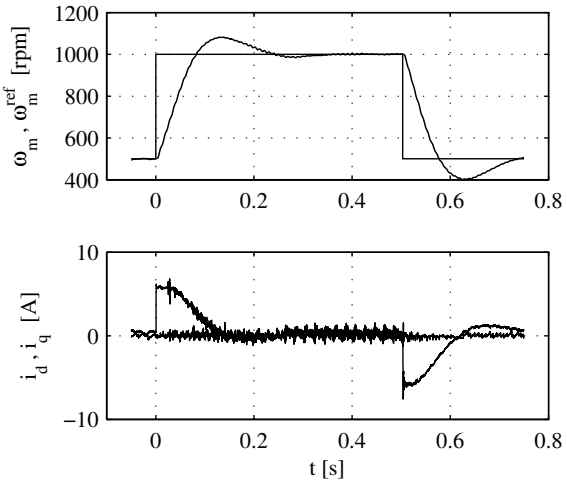

Figure 10. Experimental drive response to a 500–to–1000 rpm reference speed pulse when the actual moment of inertia is 1/3 of the one considered in the MPC model. From top to bottom: speed reference and response, and $d-$ and $q-$ currents.

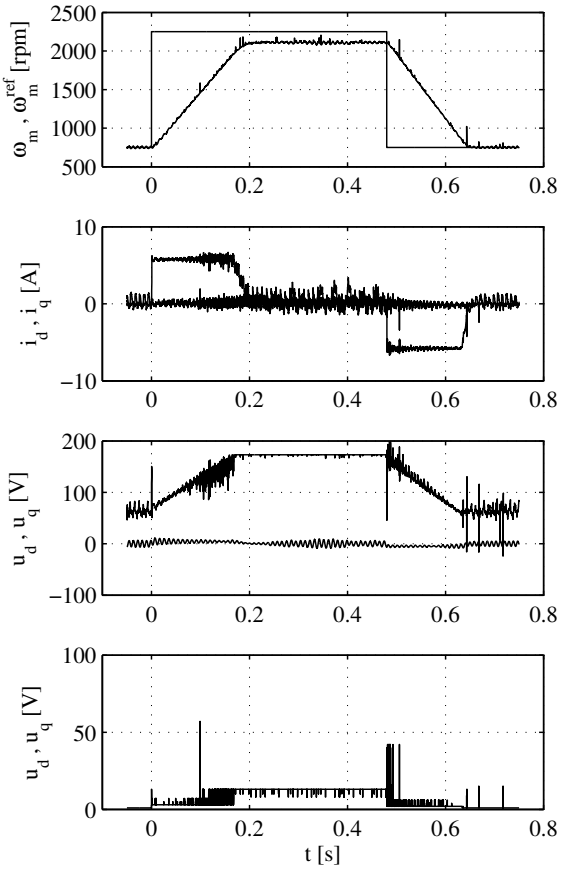

Figure 11. Experimental drive response to a 750–to–2250 rpm reference speed pulse. From top to bottom: speed reference and response, $d-$ and $q-$ currents, and $d-$ and $q-$ voltages.

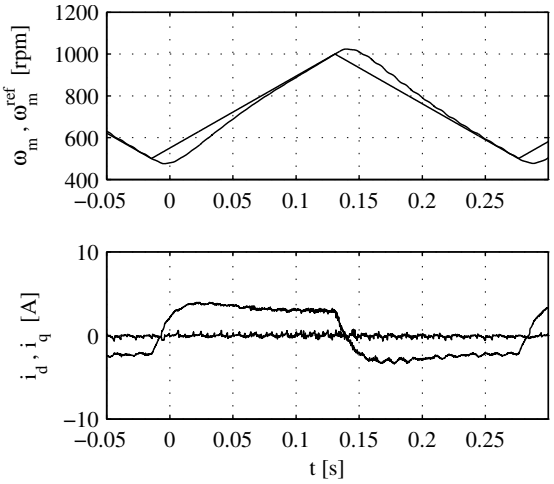

Figure 12. Experimental drive response to a lower acceleration triangular speed reference. From top to bottom: speed reference and response, and $d-$ and $q-$ current.

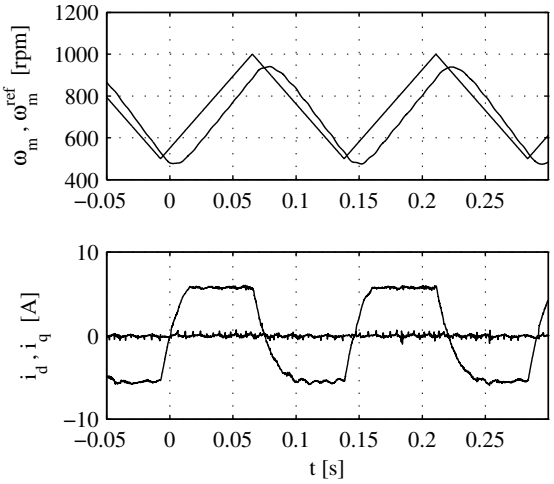

Figure 13. Experimental drive response to a higher acceleration triangular speed reference. From top to bottom: speed reference and response, and $d-$ and $q-$ current.

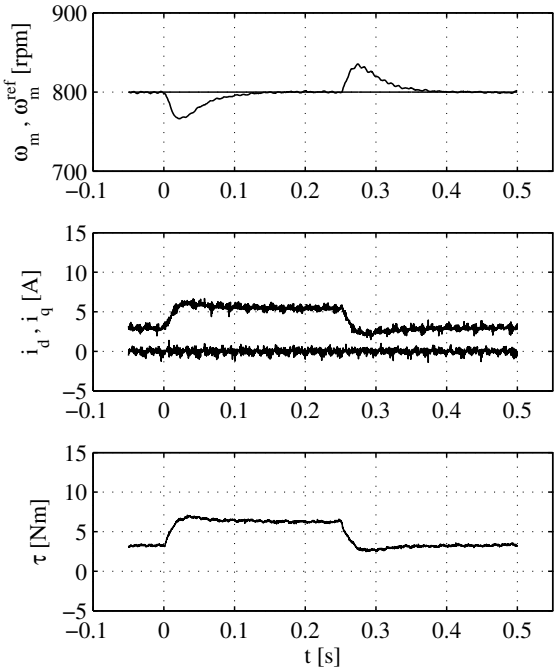

Figure 14. Experimental drive response at 800 rpm to a 20% to 40% load torque. From top to bottom: speed reference and response, $d-$ and $q-$ current, and measured torque.

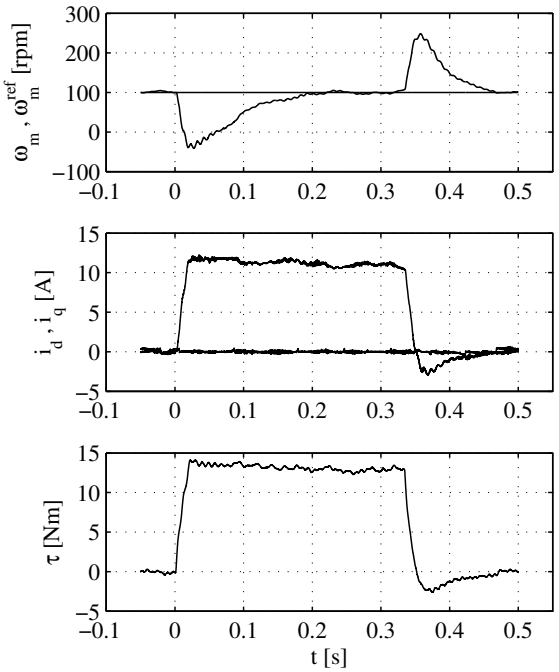

Figure 15. Experimental drive response at 100 rpm to a 0% to 100% load torque. From top to bottom: speed reference and response, $d-$ and $q-$ current, and measured torque.