Fast Solver for Implicit Continuous Set Model Predictive Control of Electric Drives

Riccardo Torchio \textsuperscript{1,1,1}, Andrea Favato \textsuperscript{2}, Paolo Gherardo Carlet \textsuperscript{2}, Francesco Toso \textsuperscript{2}, Ruggero Carli \textsuperscript{2}, Mattia Bruschetta \textsuperscript{2}, Silverio Bolognani \textsuperscript{2}, Jose Rodriguez \textsuperscript{2}, and Ludovico Ortombina \textsuperscript{2}

\textsuperscript{1}University of Padova
\textsuperscript{2}Affiliation not available

November 8, 2023

Abstract

This paper proposes a fast and accurate solver for implicit Continuous Set Model Predictive Control for the current control loop of synchronous motor drives with input constraints, allowing for reaching the maximum voltage feasible set. The related control problem requires an iterative solver to find the optimal solution. The real-time certification of the algorithm is of paramount importance to move the technology toward industrial-scale applications.

A relevant feature of the proposed solver is that the total number of operations can be computed in the worst-case scenario. Thus, the maximum computational time is known a priori. The solver is deeply illustrated, showing its feasibility for real-time applications in the microseconds range by means of experimental tests.

The proposed method outperforms general-purpose algorithms in terms of computation time, while keeping the same accuracy.
Efficient QP solver for electric motors

Andrea Favato, Paolo Gherardo Carlet, Francesco Toso, Riccardo Torchio, Ludovico Ortombina
Mattia Bruschetta, Ruggero Carli, Silverio Bolognani, IEEE Fellow, and Jose Rodriguez, IEEE Fellow

Abstract—This paper proposes a fast and accurate solver for implicit Continuous Set Model Predictive Control for the current control loop of synchronous motor drives with input constraints, allowing for reaching the maximum voltage feasible set. The related control problem requires an iterative solver to find the optimal solution. The real-time certification of the algorithm is of paramount importance to move the technology toward industrial-scale applications. A relevant feature of the proposed solver is that the total number of operations can be computed in the worst-case scenario. Thus, the maximum computational time is known a priori. The solver is deeply illustrated, showing its feasibility for real-time applications in the microseconds range by means of experimental tests. The proposed method outperforms general-purpose algorithms in terms of computation time, while keeping the same accuracy.

Index Terms—Electric Drives, Model Predictive Control (MPC), Permanent Magnet Synchronous Motor (PMSM), Synchronous Reluctance Motor (SyRM) Quadratic Programming (QP).

I. INTRODUCTION

Model Predictive Control (MPC) is an advanced optimization-based control strategy that is gaining more and more popularity in the power electronics framework [1], [2]. Increasing availability of computational power [3] and enhancement of optimization strategies have made MPC suitable for fast dynamic systems, as electric drives.

In this framework, MPC has been mostly implemented in its Finite Set (FS) form where, in each control interval, the controller applies one of the basic inverter voltage vector [4]–[8]. A second type of MPC refers to the Continuous Set (CS) technique, which requires the modulator for synthesizing the optimal voltage reference. It provides a fixed switching frequency of the converter and it works efficiently with longer sampling intervals [9] with a significant increase in computational load.

Neglecting unconstrained solutions, which nullify the advantage in using the MPC strategy, it is important to mention the attempts to use explicit MPC in the electric drives field [12], [15], [21]. However, they have shown not to allow a substantial reduction in computational cost as the parameter space increases. Instead, we are now observing a growing interest around implementations of implicit CS-MPCs due to the recent development of efficient Quadratic Programming QP solvers, leading to the establishment of this technique also in industrial contexts. Gradient methods [17], [22] and active set methods [14] are the two most widespread kind of solvers for the considered application. In Table I, a set of existing QP solvers and some of their principal features are reported. Software packages containing solvers for fast embedded optimization are available too, e.g., acados [23]. Moreover, solver-libraries designed for MPC problems have been presented, e.g., MATMPC [24].

The constrained CS-MPC for electric motor drives has gained interest in electric drives applications, too. Table II resumes some of the most relevant papers related to this topic. In [12], an active set solver has been successfully implemented on a Texas Instrument Digital Signal Processor for the CS-MPC torque control of a PMSM, proving that the technology is mature for industrial applications. The work was based on the results of [25], where the same authors provided the exact complexity certification of the Golfa-PbIdnani algorithm [26]. In these papers, an upper bound of the computation time in the worst-case scenario has been demonstrated. Moreover, a nonlinear CS-MPC was presented for the Synchronous Reluctance Motor (SyRM) in [20]. The work proved the flexibility of the MPC framework in tackling the nonlinear flux-current characteristics of such motors. The adopted QP solver for the specific application was a general purpose one, which is effective at the price of an increased complexity. Nevertheless, the computational burden is of paramount importance for embedded applications, and the emerging need from recent research advances is to design purpose-built algorithms.

In this work, we propose an accurate and fast method for solving the specific QP problem arising from MPC implementations for applications where a power converter is adopted. In particular, the algorithm is presented for the current control of synchronous motors, where limited computational hardware is usually available. The MPC implementation adopts a linear time-invariant (LTI) model of the plant and considers linear input voltages constraints. The motor model is formulated in the dq rotating reference frame, making the feasible voltage set rotating synchronously with the rotor position. The proposed QP solver takes advantage of the specific shape of the feasible set and the cost function to achieve a computationally efficient formulation. It allows for exploiting the maximum voltage deliverable by the converter, i.e., a hexagonal region in the stationary reference frame,
centered in the origin. The proposed algorithm adopts the choice of $N = 3$ prediction steps and $N_u = 1$ control horizon length, which has been assessed as a good trade-off between accuracy and computational effort [12].

A complete description of the QP solver is presented, allowing an easier replication of the algorithm. Moreover, the computational burden in the worst-case scenario are assessed both in terms of number of operations and in experimental conditions. The high accuracy of the algorithm is then assessed comparing with qpOASES [14].

The paper is organized as follows: Section II presents the adopted model of the motor, which has been proved to be effective for this application [18], [27], [28]. In Section III, the MPC problem with input constraints is stated. The novel proposed algorithm is described in Section IV. Experiments and simulations are provided in Section V and Section VI, respectively, proving the real-time feasibility and the effectiveness of the solver on a SyRM.

II. MATHEMATICAL MODEL

The voltage equations of a synchronous motor in the rotating dq-reference frame are described as follows (for simplicity we omit the time-dependence) [27]:

$$\frac{d}{dt}i_{dq} = A_e i_{dq} + B_c (u_{dq} + w_{dq}),$$

where

$$A_e = \begin{bmatrix} \frac{R_d}{L_d} & \omega_e \frac{L_m}{L_d} \\ -\omega_e \frac{L_m}{L_q} & -\frac{R_q}{L_q} \end{bmatrix}, \quad B_c = \begin{bmatrix} \frac{1}{L_d} & 0 \\ 0 & \frac{1}{L_q} \end{bmatrix},$$

and $i_{dq} = [i_d i_q]^T$, $u_{dq} = [u_d u_q]^T$, $w_{dq} = [w_d w_q]^T$. The $u_d$, $u_q$, $i_d$ and $i_q$ are the dq-axis voltages and currents, respectively, $R_q$ is the windings resistance, $\omega_e$ the electrical speed, whereas $L_d$ and $L_q$ are the dq-axis stator inductances. Assuming perfect knowledge of the parameters and neglecting model inaccuracies, the vector $w_{dq}$ includes only the Back-Electromotive Force (B-EMF), i.e., $w_{dq} = [0 - \omega_e \lambda_{pm}]^T$, being $\lambda_{pm}$ the permanent magnet flux linkage. In practice, significant unmodeled disturbances and nonlinear dynamics are present [29], due to, e.g., iron saturation, parasitic effects and harmonic modes introduced by the non-ideal rotor geometry. Moreover, the matrix $A_e$ was obtained by linearizing it around the nominal speed, and its speed dependency was lost. This choice permits to build the MPC problem offline, thus simplifying the calculations. If desired, the speed dependence could be included in the motor modeling, at price of more online computations. To account for all the disturbances, in real experiments an observer (e.g. a Kalman filter [30]) is implemented to obtain an estimate of both $i_{dq}$ and $w_{dq}$, thus improving the tracking performance of the controller and guaranteeing offset free tracking.

Since MPC is a discrete-time controller, the continuous-time synchronous motor model (1) is discretized by the Euler integration with the sampling period $T_s$, it yields to the following discrete-time model:

$$x(k + 1) = Ax(k) + Bu(k) + Bw(k),$$

where $B = T_s B_c$, $A = I + T_s A_e$, being $I$ the identity matrix, and $x = i_{dq}$ denotes the system state and the subscripts on $u(k)$ and $w(k)$ are omitted to ease the mathematical notation.
III. MODEL PREDICTIVE CONTROL OF PMSM

The LTI model (3) is used as the prediction model in the MPC problem, whose ultimate goal is to track the desired currents profiles $\mathbf{x}^{ref} = [i_q^{ref}, i_d^{ref}]^T$. The MPC optimization problem is solved at each control step $k$ with respect to the $dq$ voltage increment $\Delta u(k) = u(k) - u(k-1)$. The following quadratic functional cost is adopted:

$$\begin{align*}
J &= \sum_{j=0}^{N-1} \left( ||\mathbf{x}^{ref}(k+j+1) - \mathbf{x}(k+j+1)||_Q^2 \right) + \\
&+ \sum_{i=0}^{N_u-1} \left( ||\Delta \mathbf{u}(k+i)||_R^2 \right) + ||\mathbf{x}^{ref}(k+N) - \mathbf{x}(k+N)||_S^2,
\end{align*}$$

where $Q$, $S$ and $R$ are weighting matrices. At each time step $k$, the optimal control move is obtained by solving the following optimal control problem:

$$\begin{align*}
\min_{\Delta \mathbf{u}(x)} & \quad J \\
\text{s.t.} & \quad \mathbf{x}(k+1) = \mathbf{A} \mathbf{x}(k) + \mathbf{B} \mathbf{u}(k) + \mathbf{B} \mathbf{w}, \\
& \quad \Delta \mathbf{u}(k) = \mathbf{u}(k) - \mathbf{u}(k-1), \\
& \quad \mathbf{u}_{\alpha\beta}(k+j) = T(\theta_e) \mathbf{u}(k+j) \in \mathbb{U}_{\alpha\beta}, \quad j = 0, 1, \ldots, N_u - 1,
\end{align*}$$

where $T(\theta_e)$ is the Park anti-transformation matrix, being $\theta_e$ the electric angle, and the term $\mathbf{w}$ is assumed to be constant in the prediction horizon and equal to the estimate $\hat{\mathbf{w}}(k-1)$ provided by the Kalman filter at the previous step. $\mathbb{U}_{\alpha\beta}$ represents the feasible voltage region for $\mathbf{u}_{\alpha\beta} = [u_\alpha \ u_\beta]^T$, namely, it is the maximum voltage deliverable by the converter, i.e., a hexagonal region in the stationary reference frame, centered in the origin.

Hereafter, it is assumed the prediction horizon $N = 3$ and the control horizon $N_u = 1$. The value of $N_u$ is of great importance because it sets the size of the optimization problem (5). A small control horizon allows a very efficient implementation of the proposed solver. By introducing the solution vector $\Delta \mathbf{u} = [\Delta \mathbf{u}(k), \Delta \mathbf{u}(k+1), \ldots, \Delta \mathbf{u}(k+N_u-1)]$, the problem (5) can be reformulated as:

$$\begin{align*}
\min_{\Delta \mathbf{u}} & \quad \frac{1}{2} \Delta \mathbf{u}^T \mathbf{H} \Delta \mathbf{u} + \mathbf{c}^T \Delta \mathbf{a} + \text{const}, \\
\text{s.t.} & \quad \mathbf{f} \Delta \mathbf{u} \leq \mathbf{f},
\end{align*}$$

where the size of the matrices are $\mathbf{H} \in \mathbb{R}^{2N_u \times 2N_u}$ and $\mathbf{c} \in \mathbb{R}^{2N_u \times 1}$. By choosing $N_u = 1$, i.e., $\Delta \mathbf{u} = \Delta \mathbf{u}(k)$, the voltage constraints matrices in (6) are (see Appendix A):

$$\begin{align*}
\mathbf{F} &= \left[ \begin{array}{cccccccc}
1 & 1 & 1 & -1 & -1 & -1 \\
\sqrt{3} & 0 & -\sqrt{3} & -\sqrt{3} & 0 & \sqrt{3} \\
\end{array} \right] T(\theta_e), \\
\mathbf{f} &= \left[ \begin{array}{cccccccc}
2N_{DC} & 0.5 & 1 & 1 & 0.5 & 1 \\
\sqrt{3} & \sqrt{3} & \sqrt{3} & \sqrt{3} & \sqrt{3} & \sqrt{3} \\
\end{array} \right] - \mathbf{F}\mathbf{u}(k-1).
\end{align*}$$

It is worth noting that the above mentioned constraints describe an hexagon, which is the rotated and translated version of $\mathbb{U}_{\alpha\beta}$, according to the transformation $T(\theta_e)$ and $u(k-1)$, respectively. Clearly, the region $\mathbf{F} \Delta \mathbf{u}(k) \leq \mathbf{f}$ is composed of six segments lying on six lines that, hereafter, we denote as $\ell_i$ for $i = 1, \ldots, 6$, being $\ell_i$ associated to the constraint $\mathbf{F}(i,:) \Delta \mathbf{u}(k) \leq \mathbf{f}(i)$.

Now, let $\mathbf{u}^*$ be the optimal solution of constrained MPC problem (6), the applied control input at time $k$ is obtained as:

$$\mathbf{u}^*(k) = \mathbf{u}^*(k-1) + \Delta \mathbf{u}^*(k).$$

In the $\alpha\beta$-plane the optimal control input is then $\mathbf{u}_{\alpha\beta}(k) = T(\theta_e)\mathbf{u}^*(k)$.

IV. ALGORITHM DESCRIPTION

The aim of the proposed method is to solve the constrained QP problem (6) with a finite succession of iterations. First, the optimal solution of problem (6) is computed in closed form, neglecting the voltage constraints:

$$\Delta \mathbf{u}^{uc} = -\mathbf{H}^{-1}\mathbf{c}. \quad (8)$$

Then, the relative position between the unconstrained optimal solution $\Delta \mathbf{u}^{uc}$ and the hexagon region identifies the number of violated constraints. Four different situations can occur:

- 0) The solution is optimal and feasible, i.e., the optimal unconstrained solution lies within the feasible voltage set, and it is applied as voltage reference to the inverter;
- 1) one constraint is not satisfied (see Fig. 1);
- 2) two constraints are not satisfied (see Fig. 2);
- 3) three constraints are not satisfied (see Fig. 3).

Notice that, in the figures, the unconstrained solution $\mathbf{u}^{uc}_{\alpha\beta}$ and the respective feasible set $\mathbb{U}_{\alpha\beta}$ are represented in the $\alpha\beta$-plane. As mentioned before, a simple roto-translation allows for mapping the $dq$-plane into the $\alpha\beta$ one, making the representations equivalent in qualitative terms. In order to better clarify the last three points, namely when at least one constraint is not respected, we propose a deepening on the shape of the function cost (4) and the relative positioning of the hexagonal constraint.

A. Regions of Violated Constraints

In this section, we express the functional cost $J$ in (4) in terms of $\mathbf{u}_{\alpha\beta}$. It turns out that the level set curves are centered in $\mathbf{u}_{\alpha\beta}^{uc}$, where the eccentricity depends on the ratio of the $dq$ inductances and the orientation depends on the electric angle. Fig. 4 shows two examples of cost function shapes in the $\alpha\beta$-plane, where Fig. 4(a) represents the case of an anisotropic motor, while Fig. 4(b) shows the case of an isotropic one.

![Fig. 1. One constraint violation: the solution is found by adding the penalty function represented by the red line to the cost function.](image-url)
both cases, a representative hexagonal feasible voltage set in the $\alpha \beta$-plane described by (7) is reported.

By analyzing the six inequalities defined in (7), it is possible to relate the relative position of the unconstrained optimal solution with respect to the hexagonal feasible voltage set. Thus, the number of violated constraints can be quantified. Fig. 5 shows the partition of the plane according to the number of constraints, numbered from 0 to 3. For each of these cases, in Sec. IV-C a tailored solution strategy is described.

### B. The Fundamental Algorithm Step

Given an unfeasible unconstrained solution, the proposed method solves the problem (6) by considering only one violated constraint at a time, named $\ell_i$, regardless of the region in which the unconstrained solution is located. The fundamental routine solves the modified optimal control problem:

$$
\min_{\Delta u(k)} J(\Delta u(k)) \quad \text{s.t.} \quad F(i^*, :)\Delta u(k) \leq f(i^*), \quad (9)
$$

where only the $\ell_i$ voltage constraint is considered. The solution is computed by relaxing the constraint in (9), solving the almost equivalent unconstrained problem:

$$
\min_{\Delta u(k)} J(\Delta u(k)) + \lambda_i (F(i^*, :)\Delta u - f(i^*))^2, \quad (10)
$$

for $\lambda_i$, sufficiently high to guarantee a negligible distance from the constraint. Basically, the functional cost $J$ has been augmented with a penalty function in (10), forcing the solution to stay on the active constraint. It is worth noting that the problem (10) is unconstrained with respect to (9), therefore it admits an analytic solution.

### C. Algorithm Steps

The proposed algorithm finds the optimal constrained solution of (5) with a two-step procedure. First, it computes the unconstrained solution $\Delta u^{uc}$, then, all violated constraints, if present, are taken into account one by one by recalling the **Fundamental Step** described in Sec. IV-B. In the following, all possible cases are described.

1) **One Constraint Violation:** If $\Delta u^{uc}$ lies within the triangle adjacent to the segment of the hexagon lying on the line $\ell_i$ (see Fig. 5), the constrained optimal solution certainly lies on the segment itself. A representative example is shown in Fig. 1, where the solutions are plotted, as before, in the $\alpha \beta$-plane. The optimal solution is then computed according to the following steps:

1) The solution of the problem (10) is computed considering as constraint $\ell_i^1$, i.e., with $i^* = i$, namely $\Delta u^{oa}$;

$\lambda_i$.

---

Fig. 2. Two constraint violations. Fig. 2(a) depicts the cases where the optimal solution lies on an hexagon side, whereas in Fig. 2(b) the optimal solution lies on an external hexagon vertex.

Fig. 3. Three constraint violations. Fig. 3(a) and Fig. 3(b) depict the case where the optimal solution lies on the hexagon side and it is found in one or two procedure steps, respectively. In Fig. 3(c) and Fig. 3(d), the optimal solution lies on a hexagon vertex.
The optimal solution is then computed according to the cost function shape, three different situations can occur: depending on the location of the unconstrained solution and consecutive case, the solution lies on one of two boundaries of the feasible set is illustrated in Fig. 2. In this bounded solution lies in a region that crosses two consecutive segments that lies either on one of the hexagon segments, i.e., those that are originated by the intersection of \( \ell_i \) and \( \ell_{i+1} \), and \( \ell_{i+2} \), the optimal solution lies on one of the extreme vertexes of one of the hexagon segments, i.e., those that are not originated by the intersection of \( \ell_i \) and \( \ell_{i+1} \) (see Fig. 2(b)).

The optimal solution is then computed according to the following steps:

1) The solution to the problem (10) is computed considering as constraint \( \ell_{i+1} \), i.e., with \( i^* = i + 1 \) (the central violated constraint), namely \( \Delta u^{ao} \);

2) A feasibility check is then operated, computing \( h_j \) as in (11) for \( j = i + 2, i \);

   a) if \( \Delta u^{ao} \) does not violate any constraint, then \( \Delta u^* = \Delta u^{ao} \);

   b) if \( h_{i+2} > 0 \) or \( h_i > 0 \), \( \Delta u^* \) certainly lies on \( \ell_{i+2} \) (or \( \ell_i \) respectively) and the one constraint violation procedure can be applied; Fig. 3(b), Fig. 3(c) and Fig. 3(d) summarized the possible situations, respectively.

V. Computational Analysis

In the previous section we described an algorithm to solve the constrained QP (5) that takes into account the voltage constraints and the cost function shape to provide a solution in a fixed number of operations. The procedure results to be accurate and efficient in terms of computational burden. The computational performance of the algorithm is hereafter compared with those of the open-source tool qpOASES.

A. Comparison with qpOASES

qpOASES is an open-source software, it is very robust and suitable for medium-small scale problems, and it can be considered as a very good benchmark in terms of solution accuracy. The MPC problem is built considering a SyRM motor, whose parameters are given in Table III. The averaged time required by the proposed method is compared with the one of the open source solver qpOASES in MATLAB environment. The solvers were run on a Intel(R) Core(TM) i7-8700 CPU 3.20GHz and they were tested considering all constraints and the cost function shape.
TABLE III
OVERVIEW OF THE SYRM MOTOR AND CONTROL PARAMETERS.

<table>
<thead>
<tr>
<th>Motor Data</th>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pole pairs</td>
<td>( p )</td>
<td>2</td>
</tr>
<tr>
<td>Phase resistance</td>
<td>( R )</td>
<td>1 ( \Omega )</td>
</tr>
<tr>
<td>d-axis inductance</td>
<td>( L_d )</td>
<td>0.2 H</td>
</tr>
<tr>
<td>q-axis inductance</td>
<td>( L_q )</td>
<td>0.06 H</td>
</tr>
<tr>
<td>Nominal current</td>
<td>( I_N )</td>
<td>6 A</td>
</tr>
<tr>
<td>Nominal ( d ) current</td>
<td>( I_{N,d} )</td>
<td>3 A</td>
</tr>
<tr>
<td>Nominal ( q ) current</td>
<td>( I_{N,q} )</td>
<td>5.2 A</td>
</tr>
<tr>
<td>Nominal speed</td>
<td>( \Omega_N )</td>
<td>700 rpm</td>
</tr>
<tr>
<td>Prediction Horizon</td>
<td>( N )</td>
<td>3</td>
</tr>
<tr>
<td>Control Horizon</td>
<td>( N_u )</td>
<td>1</td>
</tr>
<tr>
<td>Input Weight d-axis</td>
<td>( r_d )</td>
<td>0.0001</td>
</tr>
<tr>
<td>Input Weight q-axis</td>
<td>( r_q )</td>
<td>0.0002</td>
</tr>
<tr>
<td>Prediction Error Weight</td>
<td>( q,s )</td>
<td>1</td>
</tr>
</tbody>
</table>

TABLE IV
AVERAGED TIME COMPARISON BETWEEN THE PROPOSED METHOD AND qpOASES FOR DIFFERENT NUMBER OF STEPS.

<table>
<thead>
<tr>
<th>Case</th>
<th>Proposed Method</th>
<th>qpOASES</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.6</td>
<td>28.2</td>
</tr>
<tr>
<td>1</td>
<td>1.65</td>
<td>30.4</td>
</tr>
<tr>
<td>1.5</td>
<td>1.67</td>
<td>30.4</td>
</tr>
<tr>
<td>2</td>
<td>1.7</td>
<td>30.0</td>
</tr>
<tr>
<td>2.5</td>
<td>1.72</td>
<td>30.0</td>
</tr>
</tbody>
</table>

As can be noticed, the proposed solver finds the MPC solution with a lower computational burden with respect to qpOASES. It is worth highlighting that in the worst case scenario, namely, when the solution violates all three constraints and the optimal one is an hexagon vertex, the average computation time remains very limited. The execution time of this case is considered the upper bound for the control algorithm cost.

B. Worst-Case Computational Cost

The real-time feasibility certification for algorithms that aspire to large-scale industrial applications plays a crucial role. To this aim, we propose a worst-case performance analysis. Since the computational performances strongly rely on the specific implementation, the total number of algebraic operations performed to find the solution in the worst-case scenario is quantified for the proposed method. The sequence of operations is listed below:

1) compute the unconstrained solution (8);
2) feasibility check (11) \( \Rightarrow \) solution unfeasible;
3) find new solution (10);
4) feasibility check (11) \( \Rightarrow \) solution unfeasible;
5) find new solution (10);
6) feasibility check (11) \( \Rightarrow \) solution unfeasible;
7) find the solution among the vertices.

The corresponding number of operations is reported in Table V. It is worth noting that step 7) does not require any additional algebraic operations. In conclusion, the peculiarities of the proposed method are twofold: the maximum number of steps are fixed, hence, the total number of operations can be evaluated and the maximum run time can be easily determined depending on the specific hardware. The aforementioned characteristics make the algorithm well suitable for implementation on industrial hardware.

VI. EXPERIMENTAL RESULTS

The test bench adopted for the experiments is shown in Fig. 6. It consists of a master motor directly connected to the SyRM motor under test, whose parameters are listed in Table III. The controllers have been implemented in a dSPACE MicroLabBox, a compact development system for
laboratory purposes, which has dual-core real-time processor at 2 GHz and dedicated electric motor control features. It provides the gate signals for the inverter and performs current and position acquisitions.

A. Solution Accuracy

To verify the effectiveness of the proposed method, a tailored test was designed. A reduced value of the DC bus was set for the controllers in order to achieve an unfeasible working point, namely, the unconstrained solution is outside the voltage hexagon. In this condition, the solver is pushed to find a solution along the hexagon edges, where voltage constraints are active. During the test, qpOASES was used as solver and it was considered as the benchmark. The recorded data were used to run the MPC problem offline and the voltage solutions were computed with the proposed method for each time-step. The results are shown in Fig. 7. The voltages are plotted in per unit with respect to the average value of the DC bus. The solutions, i.e. the voltage references, found by qpOASES and the presented solver were overlapped, as confirmed by the voltage reference found on the voltage hexagon during this transient. The solutions of both algorithms are identical, proving the correctness of the proposed solver.

B. Solver Performance

To evaluate the solver performance in terms of computational cost, we propose a current step transient (see Fig. 8). The master motor kept the motor under test at its nominal speed, and the nominal current value was set as reference. This condition represents one of the most stressing test for evaluating current controller performance. Thus, it is expected this represents the worst-case scenario for the computational cost of the algorithms. Fig. 8(a) shows the $dq$ currents transients, normalized with respect to the nominal peak current value. The most interesting part of this test can be observed in Fig. 8(b), where the turnaround time of both algorithms is measured by dSPACE. The time required by measurements acquisitions and elaborations is about 8 $\mu$s. It can be observed that there is a strong agreement between the computational performance in Table V and the experimental-based ones. The proposed solver shows a peak at the first instant with 14 $\mu$s, while the average time stabilizes at 12 $\mu$s. Fig. 8(c) shows a flag value which indicates at which step the proposed solver finds the optimal solution. Thus, in the worst-case situation, the proposed solver needs 14 $\mu$s at maximum to find the optimal voltages. The voltage solutions are reported in Fig. 8(d), which confirms that solutions are found on the voltage hexagon during this transient.

We highlight that the time required by qpOASES is inevitably higher, because the generality of the considered solver increases the required time to solve the problem.

VII. CONCLUSIONS

In this work, a fast and effective method for solving in real-time the quadratic programming problem related to the MPC has been proposed. The solver was designed for the implicit current model predictive control of a synchronous motor drive. The number of steps in the worst-case scenario is fixed, thus the number of operations can be assessed in advance. One of the main features of the proposed method is that the real-time certification can be achieved. Experimental results have been obtained by using the open source solver qpOASES as benchmark for comparing the optimal solution. The proposed solver has shown an excellent accuracy and, considering the worst-case scenario, the optimal voltage is found in few microseconds, making it promising for its implementation in large-scale real-time industrial applications.

APPENDIX

The system of inequalities in (7) describes an hexagon centered in the stator reference frame origin. However, the
hexagon is described in the rotating reference frame with respect to the optimization variable $\Delta u(k)$, which is the solution of the MPC problem. The expression of the $\ell_i$ voltage constraint, namely an hexagon edge, is $u_{\ell_i} = m_i u_a + q_i$, where $m_i, q_i$ are real numbers. In matrix notation the expression becomes:

$$[-m_i \, 1] \begin{bmatrix} u_a \\ u_{\ell_i} \end{bmatrix} = q_i.$$  

Then, (12) can be expressed with respect to the dq voltage $u$ as:

$$[-m_i \, 1] \begin{bmatrix} \cos(\theta_e) - \sin(\theta_e) \\ \sin(\theta_e) \cos(\theta_e) \end{bmatrix} \begin{bmatrix} u_d \\ u_q \end{bmatrix} = q_i.$$  

Introducing the optimization variable $\Delta u(k)$, it results:

$$[-m_i \, 1] \begin{bmatrix} \cos(\theta_e) - \sin(\theta_e) \\ \sin(\theta_e) \cos(\theta_e) \end{bmatrix} \begin{bmatrix} \Delta u_d(k) \\ \Delta u_q(k) \end{bmatrix} = \begin{bmatrix} u_d(k-1) \\ u_q(k-1) \end{bmatrix}.$$  

Each row of the constraint matrices in (7) contains the coefficients contained in matrix $F$ and $f$ in (7) are the ones of the six lines of the adopted hexagon. For instance, the first border has $m = -\sqrt{3}$ and $q = \frac{2\sqrt{3}}{3} u_{DC}$. The expression in (14) is used with the inequality relation for all the lines, which requires that the solution lies within the feasible region identified by the hexagon, including its boundaries.

References


