Model Predictive Control with Space-Vector Modulation for a Grid-Connected Converter with an *LCL*-Filter

Adriaan Sadie*, Toit Mouton*, Martinus Dorfling* and Tobias Geyer†

*Department of Electrical and Electronic Engineering
University of Stellenbosch, Private Bag X1, Matieland 7600, Stellenbosch, South Africa
Email: 18207987@sun.ac.za, dtmouton@sun.ac.za, mddorfling@sun.ac.za
†ABB Corporate Research 5405, Baden-Dätwill, Switzerland, Email: t.geyer@ieee.org

Keywords

≪Optimal control≫, ≪Pulse Width Modulation (PWM)≫, ≪Converter control≫, ≪Passive filter≫, ≪Three-phase system≫

Abstract

This paper presents an indirect model predictive control (MPC) strategy with space-vector modulation (SVM). The controller is used to control a high-power voltage-source inverter (VSI), connected to the grid via an *LCL*-filter. The controller evaluates a multi-variable and convex cost function over a long prediction horizon in order to determine an optimal sequence of references for a space-vector modulator. The MPC strategy is tested in a MATLAB simulation. The results of the simulation show excellent steady-state behaviour as well as a fast response during transients. The controller successfully dampens the resonant frequency of the filter and exhibits low grid-current harmonic distortion.

Introduction

The field of power electronics has seen an increase in the popularity of model predictive control (MPC) during the last couple of years. Traditionally, MPC has been limited to chemical plants because of the high computational burden when solving the underlying optimisation problem. With the rise of high-speed computational devices over the last few years such as field-programmable gate arrays (FPGA), MPC could be applied to numerous other applications.

This paper will focus on indirect MPC, where there is a modulator between the controller and the converter. The advantage of this approach is that the MPC decision variables are continuous, which normally results in a quadratic cost function [1, 2].

Grid-connected converter applications must meet the relevant standards. In this regard, *LCL*-filters are very popular, because of their high attenuation properties compared to the series inductor filters. One aspect that has to be kept in mind when designing an *LCL*-filter is the damping of the resonant frequency of the filter. The two main strategies used to address the resonant frequencies are known as passive and active damping. The former is quite easy to implement but has the drawback of adding damping resistors to the circuit which increase the ohmic losses greatly. There are a few papers on MPC methods that consider active damping. In [3], a term is included in the cost function that results in a high gain at the resonant frequency. This causes a large weight on the harmonic component when close to that frequency. Another method, examined in [4] and [5], implements a virtual damping resistance by injecting an additional current component on the converter-side.

Similar applications of MPC for two-level VSI's with *LCL*-filters are also considered, such as [1], [2] and [6]. In [1], the pulse-width modulator is approximated as a piecewise-affine (PWA) function, which results in PWA model of the overall system. The optimal control problem can be pre-solved using explicit MPC. As a result, the state-space is partitioned into a set of polyhedra, where each one is associated with a PWA control law. This approach is limited to short horizons and uses a high switching frequency

with regards to the resonant frequency of the filter. In [2], a long prediction horizon is incorporated for an application where the converter is connected to a load via an *LC*-filter. This approach also uses a continuous MPC strategy in conjunction with a pulse-width modulator. In [6], a reduced model of the *LCL*-filter is used to limit the number of sensors required for the controller, and an active damping strategy is also applied.

This paper proposes a multi-variable model predictive controller with a relatively low switching frequency and a long prediction horizon. These traits help the controller to damp the resonant frequency and ensures stability during transients [5]. The proposed controller is used to control a VSI connected to the grid via an *LCL*-filter. The topology of the converter is presented and the control problem is formulated. The optimisation stage is discussed where a projection algorithm is proposed for the gradient projection method. This is followed by the modulation technique. Results of the simulation are presented, a conclusion is drawn, and recommendations for future work are given.

Control Problem Formulation

System Model

The system topology is shown in Fig. 1 with the converter-side currents (i_a, i_b and i_c), the filter capacitor voltages (v_{ca}, v_{cb} and v_{cc}) and the grid-side currents (i_{ga}, i_{gb} and i_{gc}). The system model can be deduced by transforming these variables from *abc* to the αβ reference frame. This is done by using the Clarke transformation $\boldsymbol{\xi}_{\alpha\beta} = \boldsymbol{K}\boldsymbol{\xi}_{abc}$, where

$$\boldsymbol{K} = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}. \tag{1}$$

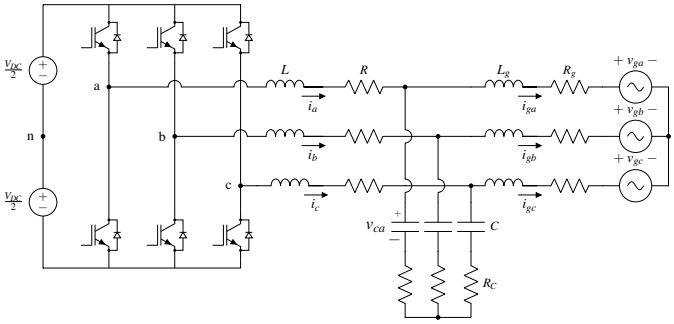

Fig. 1: Topology of the grid-connected two-level voltage-source converter.

To simplify the notation, the subscript from $\boldsymbol{\xi}_{\alpha\beta}$ is dropped. The state variables are defined as

$$\boldsymbol{i} = \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix}, \boldsymbol{i_g} = \begin{bmatrix} i_{g\alpha} \\ i_{g\beta} \end{bmatrix} \text{ and } \boldsymbol{v_c} = \begin{bmatrix} v_{c\alpha} \\ v_{c\beta} \end{bmatrix}, \text{ and the resulting state vector[1] as } \boldsymbol{x} = [\boldsymbol{i}^T\ \boldsymbol{i}_g^T\ \boldsymbol{v}_c^T]^T.$$

The output voltage of one converter leg (with regard to the centre point *n*) is given by $v_x = \frac{V_{DC}}{2}p_x$, where $x \in \{a,b,c\}$ and $p_x \in \{-1,1\}$. The continuous-time state-space equation that describes the system is defined as

$$\dot{\boldsymbol{x}} = \boldsymbol{F}\boldsymbol{x} + \boldsymbol{G}\boldsymbol{p} + \boldsymbol{P}\boldsymbol{v}_{g,abc}\ , \text{with } \boldsymbol{v}_{g,abc} = [v_{ga}\ v_{gb}\ v_{gc}]^T \text{ and } \boldsymbol{p}_{abc} = [p_a\ p_b\ p_c]^T,$$

where $\boldsymbol{v}_{g,abc}$ and $\boldsymbol{p}_{abc}$ denote the three-phase grid voltage and switch position, respectively. The matrices $\boldsymbol{F}$, $\boldsymbol{G}$ and $\boldsymbol{P}$ are defined in the appendix. Matrix $\boldsymbol{P}$ and the grid voltage $\boldsymbol{v}_{g,abc}$ represent the effect of the grid on the system. The discrete-time state-space model of the system is found by using exact discretisation [7]. More specifically, the continuous-time state-space matrices are discretised by means of [2]

$$\boldsymbol{A} = \boldsymbol{e}^{\boldsymbol{F}\frac{T_c}{2}},\ \boldsymbol{B} = -\boldsymbol{F}^{-1}(\boldsymbol{I}-\boldsymbol{A})\boldsymbol{G}, \text{ and } \boldsymbol{T} = -\boldsymbol{F}^{-1}(\boldsymbol{I}-\boldsymbol{A})\boldsymbol{P},$$

[1] T indicates the transpose of a vector or matrix.

[2] Note that $\boldsymbol{e}$ denotes the matrix exponential, and $\boldsymbol{I}$ denotes the identity matrix with appropriate dimensions.

which results in

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{p}(k) + \boldsymbol{T}\boldsymbol{v}_{g,abc}(k). \tag{2}$$

The variable T_c indicates the carrier period. The discrete-time state variables are updated at every half-carrier cycle, since asymmetric regular sampling is used.

Because of switching, the equivalent discrete-time model is non-linear. This problem is overcome by approximating the output voltage of the pulse-width modulator as a zero-order-hold (ZOH) equivalent (term $\boldsymbol{p}(k)$ in (2)).

Control Problem

The control problem consists of formulating a desirable cost function that addresses specific control objectives. The goal is to generate modulating signals (references) for a space-vector modulator. The reference signal vector $\boldsymbol{u}(k) = [u_\alpha(k)\ u_\beta(k)]^T$ is introduced, where $\boldsymbol{u}(k) \in [-1,1]^2$. Fig. 2 shows a block diagram of the proposed controller.
The tracking error between reference signals and state variables as well as a weight $\lambda_u \geq 0$ on the change in reference signal $\boldsymbol{u}(k)$ are included as terms in the cost function [8]. A non-zero weight λ_u sets the trade-off between tracking accuracy and control effort. The cost function over the prediction horizon is defined as

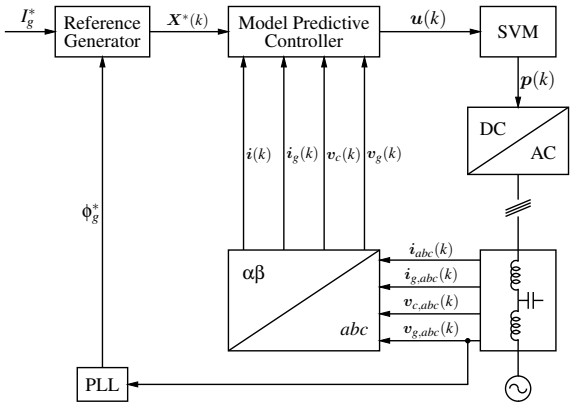

Fig. 2: Block diagram of the proposed control strategy.

$$J = \sum_{\ell=k}^{k+N_p-1} ||\boldsymbol{x}^*(\ell+1) - \boldsymbol{x}(\ell+1)||_2^2 + \lambda_u ||\boldsymbol{\Delta u}(\ell)||_2^2. \tag{3}$$

The variable $\boldsymbol{x}^*$ indicates the reference vector, and the difference of two consecutive PWM reference vectors is defined as $\boldsymbol{\Delta u}(\ell) = \boldsymbol{u}(\ell) - \boldsymbol{u}(\ell-1)$. A single reference vector is represented by $\boldsymbol{u}(k)$. The sequence of reference vectors with size $2N_p \times 1$ is defined over the prediction horizon N_p as

$$\boldsymbol{U}(k) = [\boldsymbol{u}^T(k)\ \boldsymbol{u}^T(k+1) \cdots \boldsymbol{u}^T(k+N_p-1)]^T. \tag{4}$$

The optimal sequence of reference vectors is determined for each sample of the state variables and only the first reference vector is applied as input to the pulse-width modulator at time step k. The process is repeated at time step $k+1$ and then for each subsequent time step [8]. This receding horizon policy provides feedback and improves robustness.

Optimisation Approach

Optimisation Problem in Vector Format

The optimal reference sequence is calculated by minimising the cost function in equation (3). The optimisation problem is stated as

$$\boldsymbol{U}_{opt}(k) = \arg \underset{\boldsymbol{U}(k)}{\text{minimise}}\ J, \text{ s.t. } \boldsymbol{U}(k) \in \mathbb{U}, \text{ with } \mathbb{U} \in [-1,1]^{2N_p}.$$

This optimisation problem can be rewritten in vector format by applying equation (2) successively. The result of this process is derived in [9]. The state trajectory is defined as $\boldsymbol{X}(k) = [\boldsymbol{x}^T(k+1)\ \boldsymbol{x}^T(k+2) \cdots \boldsymbol{x}^T(k+N_p)]^T$ and the corresponding reference trajectory is denoted as $\boldsymbol{X}^*(k)$. The state trajectory $\boldsymbol{X}(k)$ can be written in vector format as $\boldsymbol{X}(k) = \boldsymbol{\Gamma}\boldsymbol{x}(k) + \boldsymbol{\Upsilon}\boldsymbol{U}(k) + \boldsymbol{\Psi}\boldsymbol{V}_g(k)$.

Matrices $\boldsymbol{\Gamma}$, $\boldsymbol{\Upsilon}$ and $\boldsymbol{\Psi}$ can be found in the appendix. The cost function is then reformulated in vector format in much the same manner as in [9], except that a penalty matrix $\boldsymbol{Q}$ is included to adjust the

weight of each state variable as required. The penalty matrix is written as $\boldsymbol{Q} = \text{diag}(k_1, k_2, \ldots, k_6)$ and $\boldsymbol{Q}_c = \text{diag}(\boldsymbol{Q}, \boldsymbol{Q}, \ldots \boldsymbol{Q})$.

The cost function can then be rewritten as [7]

$$J = ||\boldsymbol{\Gamma x}(k) + \boldsymbol{\Upsilon U}(k) + \boldsymbol{\Psi V}_g(k) - \boldsymbol{X}^*(k)||^2_{\boldsymbol{Q}_c} + \lambda_u ||\boldsymbol{SU}(k) - \boldsymbol{Eu}(k-1)||^2_2 \ , \tag{5}$$

where $||\zeta||^2_{\boldsymbol{Q}_c} = \zeta^T \boldsymbol{Q}_c \zeta$. The matrices $\boldsymbol{S}$ and $\boldsymbol{E}$ represent auxiliary matrices with appropriate dimensions. After some algebraic manipulation, the cost function can be expressed in the compact form

$$J = \frac{1}{2}||\boldsymbol{U}(k)||^2_{\boldsymbol{H}} + \boldsymbol{\Theta}^T(k)\boldsymbol{U}(k) + \theta(k) \tag{6}$$

where

$$\boldsymbol{\Theta}^T(k) = 2\left\{ [\boldsymbol{\Gamma x}(k) - \boldsymbol{X}^*(k)]^T \boldsymbol{Q}_c \boldsymbol{\Upsilon} + \boldsymbol{V}_g^T(k) \boldsymbol{\Psi}^T \boldsymbol{Q}_c \boldsymbol{\Upsilon} + \lambda_u [\boldsymbol{Eu}(k-1)]^T \boldsymbol{S} \right\} \tag{7}$$

and

$$\boldsymbol{H} = 2[\boldsymbol{\Upsilon}^T \boldsymbol{Q}_c \boldsymbol{\Upsilon} + \lambda_u \boldsymbol{S}^T \boldsymbol{S}] \ . \tag{8}$$

The term $\theta(k)$ in (6) can be neglected from the cost function as it is merely a constant offset. The matrix $\boldsymbol{H}$ is known as the Hessian; it is a symmetrical matrix of second-order partial derivatives of the scalar-valued cost function J. See [7] for more information regarding the Hessian.

Reference Signal Generation

In order to generate the reference vector $\boldsymbol{X}^*(k)$ in equations (5) and (7), the phase of the grid voltage is required. The phase is measured by means of a phase-locked-loop (PLL) and synchronised with the controller. The desired power factor is set by an outer control loop. With this information, the phase of the reference grid current ϕ^*_g can be calculated and the amplitude of the grid current I^*_g is chosen. The other reference signals can simply be calculated by means of phasor analysis.

Conditioning of Hessian

The goal is to implement the controller on an FPGA. The optimisation technique must, therefore, be efficient and relies on the conditioning of $\boldsymbol{H}$. The conditioning of a symmetrical matrix is defined as the ratio between its smallest and largest eigenvalue. A badly conditioned $\boldsymbol{H}$ has a conditioning ratio close to zero. This means that the matrix is close to being singular and becomes difficult to invert. The contours of the level sets of an ill-conditioned $\boldsymbol{H}$ are almost parallel. For optimisation techniques such as the gradient method, this leads to a slow convergence rate.

The coordinate system in which the optimal reference vector $\boldsymbol{U}_{opt}$ is formulated affects the conditioning of the Hessian $\boldsymbol{H}$. In the *abc* coordinate system, there are multiple vectors that produce the zero vector (when $u_a = u_b = u_c$). The optimal solution is, therefore, not unique and $\boldsymbol{H}$ becomes positive semidefinite (for $\lambda_u = 0$). For this reason, the optimal reference vector is formulated in the $\alpha\beta$ reference frame to ensure uniqueness, thus improving the conditioning of $\boldsymbol{H}$ and decreasing the problem dimension by one third (from $3N_p$ to $2N_p$).

Optimisation Algorithm

Three optimisation methods were considered to solve the optimisation problem, namely multi-parametric programming, active set methods, and interior point methods. According to [10], these methods either require large amounts of memory to store the controller (in case of multi-parametric programming), or high computational power when solved on-line. These methods were also reviewed in [11] and found to be either constrained to small-scale problems, or computationally taxing. A fourth solution method, the gradient projection method, is simple, yet robust and can be used for problems with a high number of dimensions without incurring a high computational burden. For this reason, the gradient projection

method is chosen as optimisation technique in this paper. The gradient of J can easily be calculated as

$$\nabla J(\boldsymbol{U}(k)) = \boldsymbol{H}\boldsymbol{U}(k) + \boldsymbol{\Theta}(k).$$

The gradient projection method uses the negative gradient $-\nabla J$ in order to point in the direction of the steepest descent. From any starting point, a step is taken in the steepest descent direction to provide the next iterate. When the next iterate falls within the infeasible region, it is projected back towards the feasible region. The step size requires the value of L, which is known as the Lipschitz continuous gradient of J, or the L-smoothness of J. Since the cost function J is twice continuously differentiable, the value of L is set equal to the largest eigenvalue of $\boldsymbol{H}$ [12].

The solution space of the optimisation problem forms a hexagon in the αβ-plane, which requires a complex projection method when compared to the simple box-constraints of the *abc* framework. The algorithm would have to perform a hyperplane search as well as a projection operator for every $\boldsymbol{u}$ in $\boldsymbol{U}$. Therefore, an alternative method is chosen, where each $\boldsymbol{u}$ is transformed back to *abc*, where a simple projection can be done using the box-constraints. The variables are transformed from αβ to *abc* using $\boldsymbol{\xi}_{abc} = \boldsymbol{K}^{-1}\boldsymbol{\xi}_{\alpha\beta}$, where $\boldsymbol{K}^{-1} = 1.5\boldsymbol{K}^T$ (see (1)). In Fig. 3, these constraints are visualised in a 2-D space, depicting only the *a*- and *b*-phases.

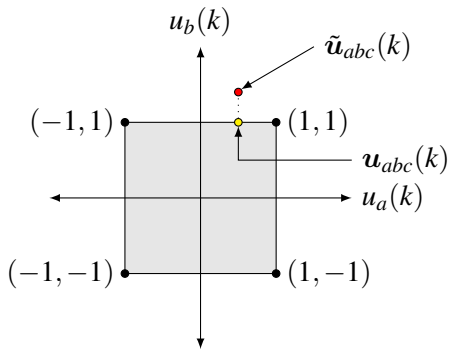

Fig. 3: Projection principle.

In the figure, $\tilde{\boldsymbol{u}}_{abc}(k)$ denotes the non-projected reference vector and $\boldsymbol{u}_{abc}(k)$ denotes the orthogonal projection thereof onto the "box". Since the transformation assumes that the zero-sequence is zero, a common-mode term is added to establish an appropriate zero-sequence.

Let $\boldsymbol{u}_{abc}(k) = [u_a(k)\ u_b(k)\ u_c(k)]^T$ denote the modulating vector at the current iteration of the optimisation step. As discussed in [13], the SVM-type common-mode component can be added to $\boldsymbol{u}(k)$, with

$$\boldsymbol{u}_{svm}(k) = \boldsymbol{u}_{abc}(k) - \frac{\max(\boldsymbol{u}_{abc}(k)) + \min(\boldsymbol{u}_{abc}(k))}{2} \cdot [1\ 1\ 1]^T.$$

The addition of the common-mode term is indicated as the SVM-operator in Algorithm 1.

Algorithm 1 Gradient Projection Algorithm

function GRADIENT PROJECTION METHOD($\boldsymbol{H}, \boldsymbol{\Theta}, \boldsymbol{u}_0, L$)
  for $i = 1 : N_f$ **do** ▷ For a predetermined number of iterations.
    $\boldsymbol{U}_i \leftarrow \boldsymbol{U}_{i-1} - \frac{1}{L}(\boldsymbol{H}\boldsymbol{U}_{i-1} + \boldsymbol{\Theta})$ ▷ Step in steepest descent.
    for $n = 1 : N_p$ **do** ▷ For each $\boldsymbol{u}$ in $\boldsymbol{U}$.
      $\tilde{\boldsymbol{u}}_{abc}(n) \leftarrow \text{SVM}(\boldsymbol{K}^{-1}\boldsymbol{u}(n))$ ▷ Transform to *abc* and add common-mode term.
      for $k = 1 : 3$ **do** ▷ Project $\tilde{\boldsymbol{u}}_{abc}$ onto box.
        $\boldsymbol{u}_{abc}(k) \leftarrow \tilde{\boldsymbol{u}}_{abc}(k)$
        if $\boldsymbol{u}_{abc}(k) \geq 1$ **then**
          $\boldsymbol{u}_{abc}(k) \leftarrow 1$
        else if $\boldsymbol{u}_{abc}(k) \leq -1$ **then**
          $\boldsymbol{u}_{abc}(k) \leftarrow -1$
        end if
      end for
      $\boldsymbol{u}(n) \leftarrow \boldsymbol{K}\boldsymbol{u}_{abc}(n)$ ▷ Transform back to αβ reference frame.
    end for
  end for
end function

Implementation Feasibility

The Xilinx Zedboard XC7Z020 development board is considered for implementation of the controller, since it is relatively cheap and includes multiple on-board peripherals and expansion capabilities. The Zynq-7020 SoC architecture includes both a dual-core ARM Cortex-A9 processing system (PS) and a 28 nm Xilinx programmable logic (PL). Only the latter will be used for the controller. The PL contains 220 DSP slices, which each include an 18×25 signed multiplier and a 48-bit adder/accumulator.

The controller executes at twice the switching frequency $f_s = 1640\,\text{Hz}$ of the converter, which means that the total allowed calculation time is 303 µs. The calculation of $\boldsymbol{\Theta}(k)$ and the gradient projection algorithm are the largest contributors to resource/time usage, since both require a high number of multiplications. There exists a trade-off between the amount of required FPGA resources and the amount of time it takes those resources to execute operations, e.g. for 100 multiplications, 10 multipliers might require 10 clock cycles, whereas 100 multipliers would require a single clock cycle.

After rearranging equation (7) and taking into account that certain matrices can be calculated off-line, the amount of on-line multiplications needed for $\boldsymbol{\Theta}(k)$ is $16N_p + 18N_p^2$. The gradient projection method requires $N_f(4N_p^2 + 12N_p)$ multiplications. This means a total number of $N_p^2(4N_f + 18) + N_p(12N_f + 16)$ multiplications per sampling interval. The optimisation algorithm is limited to $N_f = 50$ iterations. It is assumed that each multiplication is executed at every consecutive clock cycle (at 10 MHz). Table I shows estimates of time required for the controller to execute for different values of N_p.

Table I: Control algorithm execution times

	Multipliers				
N_p	1	10	50	100	220
1	83.4 µs	8.34 µs	1.67 µs	0.83 µs	379 ns
4	595 µs	59.5 µs	11.9 µs	5.95 µs	2.7 µs
8	1.9 ms	188.8 µs	37.8 µs	18.9 µs	8.58 µs
12	3.9 ms	387.8 µs	77.6 µs	38.8 µs	17.6 µs
16	6.6 ms	656.6 µs	131.3 µs	65.7 µs	29.8 µs

Results

System Parameters

The mathematical system model is realised in a MATLAB-based simulation. The system parameters are chosen as follows: prediction horizon $N_p = 14$; input change penalty $\lambda_u = 6 \times 10^4$; DC-link voltage $V_{DC} = 1.05\,\text{kV}$; grid voltage (line-to-line) $V_g = 690\,\text{V}$ (rms); fundamental grid frequency $f_1 = 50\,\text{Hz}$; switching frequency $f_s = 1.65\,\text{kHz}$; simulation sampling interval $T_s = 606.07\,\text{ns}$. The *LCL*-filter parameters are given in Table II. The nominal grid current is chosen as $I_g = 4132\,\text{A}$ (rms) and a unity power factor is chosen $\phi_g = 0^\circ$ at the point of common coupling (PCC). The other reference signals are generated by using phasor analysis. The penalty matrix is chosen as $\boldsymbol{Q} = \text{diag}(0.2\ 0.2\ 1\ 1\ 0.1\ 0.1)$ in order to prioritise the grid-side current. The resonant frequency of the transfer function from the converter output voltage to the grid current is calculated as $f_{res,1} = 690\,\text{Hz}$. The resonant frequency of the transfer function from the converter output voltage to the converter current is at $f_{res,2} = 537\,\text{Hz}$ [5].

Table II: System parameters

Parameter	Symbol	Value
Filter inductance	L	4.17 µH
Grid inductance	L_g	44.38 µH
Filter capacitance	C	1.98 mH
Filter inductor ESR	R	0.54 mΩ
Grid inductance ESR	R_g	1.76 mΩ
Filter capacitor ESR	R_C	0.67 mΩ

Simulation Results

The base values for the per unit system (pu) are $V_{base} = \sqrt{2}V_g$, $I_{base} = \sqrt{2}I_g$ and $\omega_b = f_1$. In order to show the response of the controller during transients, a step in the grid-current reference signals is

introduced at $t = 120\,\text{ms}$ from 0.5 to 1 pu.
The converter currents and filter capacitor voltages are shown in Fig. 4 and Fig. 5, respectively. The grid-side currents are shown in Fig. 6, and the modulation signals are shown in Fig. 7. Fig. 8 shows the output of the pulse-width modulator.

The grid current spectrum is shown in Fig. 9 with the magnitude of the fundamental component at 1 pu. The total demand distortion (TDD) of the grid current at steady state is 0.66% at nominal grid current. When the reference grid current is at 0.5 pu, the TDD is 1.09%. The fundamental component of the grid-current has an amplitude of 5842 A, which is close to the reference amplitude of 5844 A. The largest harmonic occurs at 350 Hz with an amplitude of 234.11 A. The controller has a settling time of roughly 2.5 ms, as the reference signals step from 0.5 pu to the nominal value for I_g.

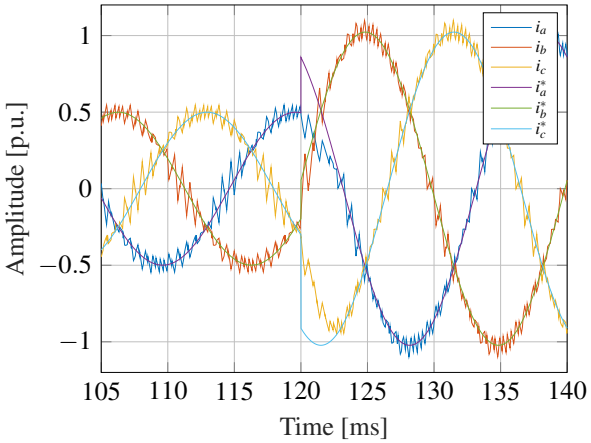

Fig. 4: Converter currents with references.

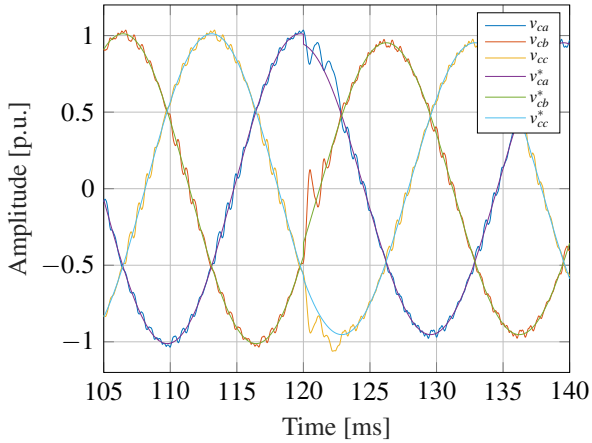

Fig. 5: Filter capacitor voltages with references.

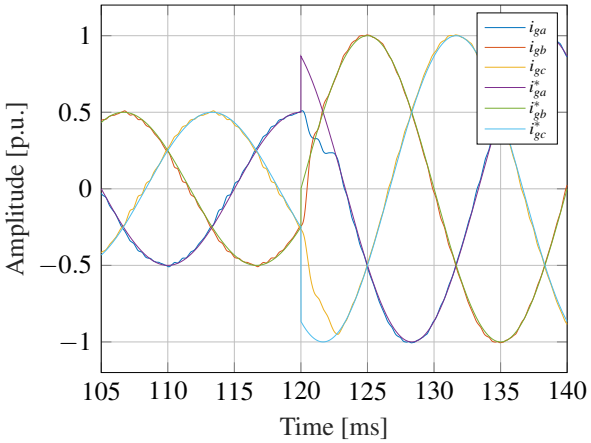

Fig. 6: Grid-side currents with references.

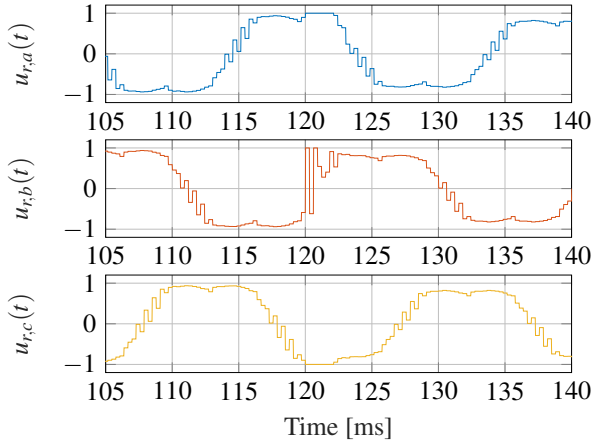

Fig. 7: Modulating signals (SVM references).

Conclusion

The controller performs well during steady-state and transient conditions, as it produces grid currents with a very low TDD. The system responds quickly to a step in the reference signals and settles within 2.5 ms, which is one eighth of the fundamental period.
The prediction horizon can be chosen without substantially increasing the computational complexity of the control algorithm. The results prove that the controller does not need an active damping strategy as the model predictive controller automatically damps the resonances of the *LCL*-filter. Classic, PI-based controllers with an active damping loop require switching frequencies of at least three times the resonance frequency [14], i.e., 2070 Hz in this example. The model predictive controller operates at

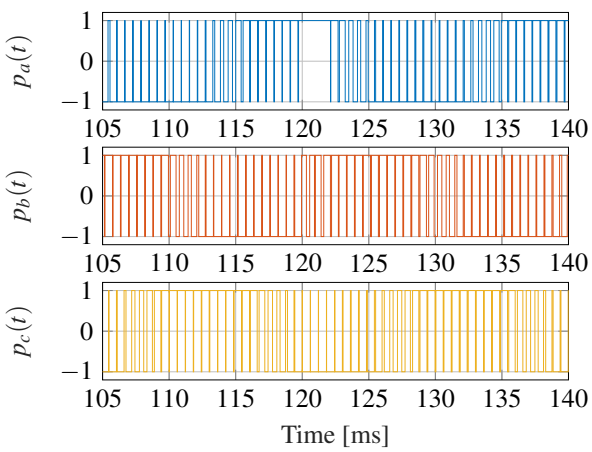

Fig. 8: Pulse-width modulated signals.

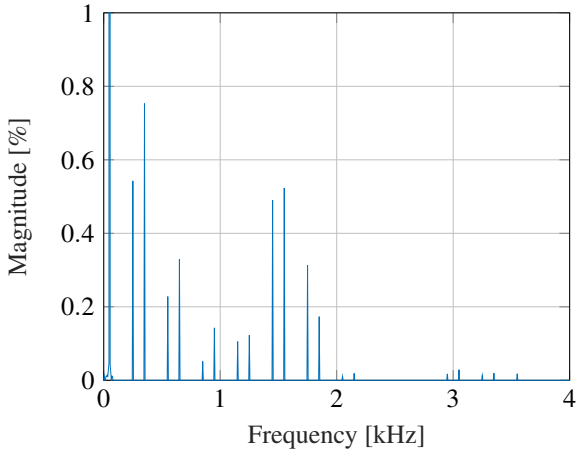

Fig. 9: Grid current spectrum at steady-state nominal grid current.

1650 Hz and, thus, at 2.4 times the resonance frequency. The lower switching frequency reduces the switching losses of the converter, thus increasing the efficiency.
When using a high-speed FPGA, natural sampling can be used to reduce the large seventh harmonic.

Recommendations

The effects of grid harmonics on the converter currents should be considered, as it creates unwanted harmonic content in the controller. One possible approach is to implement a proportional-integral (PR) controller to suppress specific harmonics. Another strategy, as mentioned earlier and used in [4], involves the implementation of a virtual resistance (VR) in series with the grid inductor in order to attenuate the grid harmonics. A follow-up paper will include practical results.

References

[1] S. Mariéthoz and M. Morari, "Explicit model-predictive control of a PWM inverter with an *LCL* filter," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 2, pp. 389–399, 2009.

[2] S. Vazquez, C. Montero, C. Bordons, and L. G. Franquelo, "Model predictive control of a VSI with long prediction horizon," *International Symposium on Industrial Electronics (ISIE)*, pp. 1805–1810, 2011.

[3] H. Miranda, R. Teodorescu, P. Rodriguez, and L. Helle, "Model predictive current control for high-power grid-connected converters with output *LCL* filter," in *2009 35th Annual Conference of IEEE Industrial Electronics*, 2009, pp. 633–638.

[4] J. Scoltock, T. Geyer, and U. K. Madawala, "A model predictive direct current control strategy with predictive references for MV grid-connected converters with *LCL*-filters," *IEEE Transactions on Power Electronics*, vol. 30, no. 10, pp. 5926–5937, 2015.

[5] N. Panten, N. Hoffmann, and F. W. Fuchs, "Finite control set model predictive current control for grid-connected voltage-source converters with *LCL* filters: A study based on different state feedbacks," *IEEE Transactions on Power Electronics*, vol. 31, no. 7, pp. 5189–5200, 2016.

[6] D. K. Yoo, L. Wang, E. Rogers, and W. Paszke, "Model predictive control of three phase voltage source converters with an *LCL* filter," in *2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE)*. IEEE, 2014, pp. 562–567.

[7] T. Geyer, *Model predictive control of high power converters and industrial drives*. John Wiley & Sons, 2016.

[8] T. Geyer and D. E. Quevedo, "Multistep finite control set model predictive control for power electronics," *IEEE Transactions on Power Electronics*, vol. 29, no. 12, pp. 6836–6846, 2014.

[9] J. M. Geldenhuys, H. du Toit Mouton, A. Rix, and T. Geyer, "Model predictive current control of a grid connected converter with *LCL*-filter," in *2016 IEEE 17th Workshop on Control and Modeling for Power Electronics (COMPEL)*, 2016, pp. 1–6.

[10] G. Goodwin, M. M. Seron, and J. A. de Dona, *Constrained Control and Estimation: An Optimisation Approach*. Springer, Mar. 2006.

[11] S. Richter, "Computational complexity certification of gradient methods for real-time model predictive control," Doctoral Thesis, ETH Zurich, 2012.

[12] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[13] D. G. Holmes, "The significance of zero space vector placement for carrier-based PWM schemes," *IEEE Transactions on Industry Applications*, vol. 32, no. 5, pp. 1122–1129, 1996.

[14] T. Mouton and T. Geyer, "Trajectory-based LQR control of a grid-connected converter with an *LCL* filter," *Proc. of the IFAC Conference on Nonlinear Model Predictive Control, Madison, WI, USA, Aug*, 2018.

Appendix

The matrices mentioned in the paper are defined as

$$\boldsymbol{F} = \begin{bmatrix} \frac{R+R_C}{-L} & 0 & \frac{R_C}{L} & 0 & \frac{1}{-L} & 0 \\ 0 & \frac{R+R_C}{-L} & 0 & \frac{R_C}{L} & 0 & \frac{1}{-L} \\ \frac{R_C}{L_g} & 0 & \frac{R_g+R_C}{-L_g} & 0 & \frac{1}{L_g} & 0 \\ 0 & \frac{R_C}{L_g} & 0 & \frac{R_g+R_C}{-L_g} & 0 & \frac{1}{L_g} \\ \frac{1}{C} & 0 & \frac{1}{-C} & 0 & 0 & 0 \\ 0 & \frac{1}{C} & 0 & \frac{1}{-C} & 0 & 0 \end{bmatrix}, \boldsymbol{G} = \begin{bmatrix} \frac{V_{DC}}{2L} & 0 \\ 0 & \frac{V_{DC}}{2L} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \boldsymbol{P} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{-L_g} & 0 \\ 0 & \frac{1}{-L_g} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \boldsymbol{K},$$

$$\boldsymbol{\Gamma} = \begin{bmatrix} (\boldsymbol{A})^T & (\boldsymbol{A}^2)^T & \cdots & (\boldsymbol{A}^{N_p})^T \end{bmatrix}^T,$$

$$\boldsymbol{\Upsilon} = \begin{bmatrix} \boldsymbol{B} & 0 & \cdots & 0 \\ \boldsymbol{AB} & \boldsymbol{B} & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ \boldsymbol{A}^{N_p-1}\boldsymbol{B} & \boldsymbol{A}^{N_p-2}\boldsymbol{B} & \cdots & \boldsymbol{B} \end{bmatrix} \text{ and } \boldsymbol{\Psi} = \begin{bmatrix} \boldsymbol{T} & 0 & \cdots & 0 \\ \boldsymbol{AT} & \boldsymbol{T} & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ \boldsymbol{A}^{N_p-1}\boldsymbol{T} & \boldsymbol{A}^{N_p-2}\boldsymbol{T} & \cdots & \boldsymbol{T} \end{bmatrix}.$$