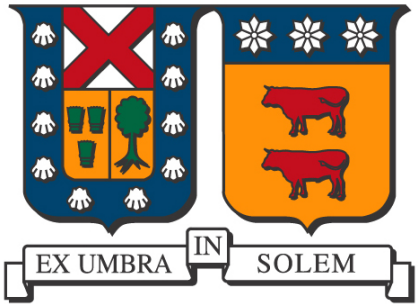


Universidad Técnica Federico Santa María

Department of Electrical Engineering

A Two–Stage Model Predictive Control Strategy for Three–level NPC Converters based on Multistep Finite Control Set and Optimal Pulse Patterns

Author:
Cristóbal González

Supervisor:
Prof. Alejandro Angulo

A thesis submitted in partial fulfillment of the requirements for the degree of

MSc Electrical Engineering

November 8, 2023

Abstract

Power transference between energy sources and the power system must meet a long and rigorous list of technical and operational requirements. To this end, LCL filter grid–connected three–level neutral point clamped converters have become a popular configuration for medium–voltage high–power applications, such as integrating distributed energy resources. The presence of the LCL filter increases the complexity of the control system when using traditional linear control schemes, so model predictive control (MPC) strategies arise as a convenient alternative as they have proved to render excellent reference tracking together with a fast dynamic response. Nevertheless, the computational complexity of MPC strategies becomes an issue as commercial control platforms must execute the real–time calculations in some tens of microseconds. Among the possible MPC strategies available, multistep–finite–control–set (MFCS) MPC has already been proven to be a viable option regarding computational burden. However, MFCS controllers do not generate periodic nor symmetric voltages between the phases of the system, resulting in continuous Fourier spectra in the system's states. This a fundamental drawback for grid–connected applications, as grid codes pose limits to the admissible amplitudes of the grid currents' harmonic components, where the analysis over interharmonics depends on the specific application, making MFCS controllers unsuitable for this type of application. On the other hand, optimal pulse patterns (OPP) guarantee a low and fixed switching frequency of the semiconductors while minimizing the distortion of the state variables. However, the state–of–the–art formulations for OPPs assume that the system can be modeled as a purely inductive load. Such a model would lead to suboptimal performance and would not allow adding constraints that guarantee meeting with grid codes regarding the grid currents.

This work proposes integrating two optimization stages dealing with different control objectives into a single and generalized control framework. This framework was then used to develop a control strategy (OPP–MFCS) that combines the benefits of OPPs and MFCS controllers. To this end, a novel formulation for OPPs considering the model of the LCL filter was proposed, allowing the direct minimization of the total demand distortion (TDD) of the grid currents while guaranteeing that the admissible limits for the harmonic components are met. On the other hand, by formulating a novel variable–step–size prediction horizon MFCS controller, extending the time–span covered by the prediction horizon with no additional computational burden was possible. This improvement led to an enhanced controller performance regarding both transient and steady–state responses of the system.

Extensive analyses of the contributions regarding OPPs, MFCS controllers, and the OPP–MFCS strategy were held. The OPP–MFCS strategy was tested through both simulation and hardware–in–the–loop experiments, showing its capability of meeting with the IEEE Std. 519–2022 technical standard while the converter operates at a low and fixed switching frequency of the semiconductors, with periodic and symmetric voltages, fulfilling all control objectives defined.

Acknowledgements

A mi familia por su incondcional apoyo y amor. A mis amigos/as y compañeros/as con quienes siempre pude contar. A los cabros de pañol, que salvaron esta tesis un montón de veces y fueron un gran apoyo en varias más. A Leyre, que cuidó de mi todo este último año. A mis profesores, que me formaron con una paciencia y dedicación inconmensurable, en especial al profesor Pablo Lezana, con quien siempre he podido contar en momentos difíciles y en el objetivo de perseguir mis metas futuras, y al profesor Alejandro Angulo, por darme tantas oportunidades de formarme como profesional y por el tremendo apañe e infinita paciencia en este largo proceso.

Contents

List of Figures

List of Tables

List of acronyms

3L–NPC: Three–level NPC.

BRAM: Block RAM.

CLUT: Combinational look–up–table.

DSP: Digital signal processors.

DSP–s: DSP slice.

FF: Flip–Flop.

FPGA: Field–programmable–gate–arrays.

HDL: Hardware description language.

HLS: High–level synthesis.

ILS: Integer least squares.

LUT: Look–up–table.

MFCS: Multistep finite–control–set.

MINLP: Mixed integer nonlinear programming.

MPC: Model predictive control.

NLP: Nonlinear programming.

NPC: Neutral point clamped.

OPP: Optimal pulse patterns.

OPRG: Optimization–based reference generator.

OPRG–OS: Optimization–based reference generator: offline stage.

OPRG–RS: Optimization–based reference generator: model–based real–time stage.

PDF: Probability density function.

PLL: Phase–locked–loop.

ROS: Real–time optimization stage.

RTL: Register transfer level.

SA: Switching angle.

SDA: Sphere decoder algorithm.

SHM: Selective harmonic mitigation.

TDD: Total demand distortion.

Chapter 1

Introduction and motivation

1.1 Context and motivation

There are no doubts about how relevant the integration of renewable energies into power systems has become in the last three decades. Most countries in the world have centered their politics and energy planning around the diversification of the energy matrix, aiming to include a wide variety of energy resources [3, 4, 5]. The enabling technology in this integration process is power electronics–based converters, as they allow incorporating sources working with different voltage levels, AC and DC voltages, and also different frequencies [6, 7, 8]. However, the operation of the power converters must not only be efficient to reduce the system's operational costs, but it must also meet a wide variety of requirements defined in technical standards [1, 9, 10, 11]. When assuming a balanced and strong grid (as in this work), those requirements are mainly limits imposed on the harmonic content of the currents at the point of connection to the power system. In that regard, there is a well–known trade–off between the harmonic distortion of the system states and the switching frequency of the power converter semiconductors. In medium–voltage high–power applications like the one analyzed in this document, operating at high switching frequencies leads to significant losses because of the large currents circulating through the semiconductors. To reduce the switching frequency and still meet grid codes, LCL filters interfacing the converter with the utility grid became popular as an effective solution [12]. The main issue that arose with LCL filters is the oscillations induced by its low–frequency resonances, causing stability issues when operating with linear controllers [13], which led to complex control schemes that consider strategies as active damping [14].

With the industry advances in the hardware capacity of commercial control platforms, specifically in field–programmable–gate–arrays (FPGA) and digital signal processors (DSP), optimization–based control techniques such as model predictive control (MPC) have gained popularity through the last decade [15, 16, 17]. MPC works by minimizing a cost function at each control step, subjected to a set of constraints applied to the control action and the predicted system states to guarantee feasible operation. Additionally, the receding horizon approach allows high–speed dynamic responses. It has also been proved that MPC strategies with long prediction horizons do not require additional control elements as active damping, as they are capable of dealing with the oscillations of the LCL filter. One of the main drawbacks of MPC is the computational burden involved in solving the optimization problem in real–time. The number of inputs and states and the prediction horizon length increase the computational capacity required to solve the optimization problem. In that line, multistep finite–control–set (MFCS) controllers have been proven to be a feasible option, as commercial control platforms are capable of solving the associated optimization problem in real–time [15, 18]. Nevertheless, MFCS controllers do not generate periodic voltage waveforms nor symmetric voltages between the three phases of the converter, generating a continuous and unpredictable Fourier spectrum in the states of the system. Therefore, in its current form, MFCS controllers are unsuitable as a control option for grid–connected applications [19].

On the other hand, optimal pulse patterns (OPP) arise as an excellent option for low and fixed switching frequency operation. OPPs are steady–state voltage patterns that minimize a cost function (related to the state variables harmonic distortion) while fixing the switching frequency of the semiconductors [20]. They are calculated offline for a set of modulation indexes and then stored in look–up–tables (LUTs). On their own, OPPs achieve a slow dynamic response but an excellent

trade–off between harmonic distortion and switching frequency. To improve the dynamic response of OPPs, the MPC strategy presented in [21] and [2] incorporated them into the control scheme, which resulted in a high–speed dynamic response and an excellent steady–state performance. The main drawback of this method is the high complexity of the optimization problem solved to define the actuation and the complexity of the control algorithm. Even though OPPs have been used for LCL filter grid–connected converters [22, 2], there is no literature documenting the incorporation process of the LCL filter model to the OPP optimization problem, even though the authors in [2] reported it was done in that work. Additionally, constraints that guarantee meeting grid codes related to the grid currents have never been included.

As can be noticed, controlling an LCL filter grid–connected medium–voltage high–power converter poses a set of control objectives. On one hand, long–term objectives arise, such as minimizing the distortion of the grid currents, operating at a low and fixed semiconductors' switching frequency, and producing periodic and symmetric voltages between the three phases. On the other hand, short–term objectives must also be met, such as a fast dynamic response, a low computational burden so the strategy can be executed in real–time, long prediction horizons to avoid additional elements such as active damping, and a low–complexity control structure. Because both long and short–term objectives are mixed in one single control problem, generating a control framework in which different entities deal with different objectives according to the time–span related to each objective arose as a natural procedure. However, those entities must work together in a single control strategy.

This work proposes a control framework that allows the development of control strategies capable of dealing with different–term objectives. Namely, for this particular control problem, the positive features of OPPs and MFCS controllers are leveraged to generate a single control strategy.

1.1.1 Hypothesis

Consider an LCL filter grid–connected three–level converter. Short– and long–term objectives must be met to guarantee proper performance. Regarding these objectives, we claim the following:

OPP and MFCS control can be integrated into one unified scheme using a reference generator block as a linking element. The OPPs can be used in real–time by the reference generator to define steady–state references to be tracked by the MFCS controller. The optimization problem defining the patterns can include the LCL filter model, resulting in an enhanced steady–state performance that guarantees meeting grid codes, whereas the MFCS control strategy will render a fast dynamic response and a proper tracking of the OPPs steady–state optimal trajectories, which will lead to fulfilling the long–term objectives of the control problem.

1.1.2 Objectives

- To design and implement a control framework with two optimization stages linked by a reference generator, consisting of the offline calculation of OPPs, and an online MFCS controller, for an LCL filter grid–connected three–level neutral point clamped (3L–NPC) converter.
- To formulate an OPPs optimization problem that incorporates the model of the LCL filter so it guarantees minimizing the harmonic distortion of the grid currents and meeting grid codes as [1] and [9].
- Develop an online reference generator that is suitable for any system in state–space representation and can be executed in real–time using a commercial control platform.
- To enhance the traditional MFCS control problem formulation by efficiently extending the prediction horizon over time to deal with the oscillations induced by the LCL filter.
- Apply the proposed strategy to a 3L–NPC converter and verify its performance regarding dynamic response and grid codes.
- Develop an efficient implementation of the proposed strategy and verify its viability to be executed in real–time by a commercial control platform.
- Implement the complete strategy in an experimental setup to test the performance of the control strategy.

Chapter 2

Background

This chapter provides the theoretical concepts and state–of–the–art of the core topics involved in this work, namely LCL filter grid–connected converters, OPP, and MFCS controllers. The main issues of controlling an LCL filter grid–connected converter are presented, together with the reasons to deem OPPs and MFCS controllers as promising alternatives for this type of system. Finally, the traditional assumptions and mathematical formulations of OPPs and MFCS controllers are detailed, and each strategy's positive features and weaknesses, in their current state, are highlighted.

2.1 LCL filter grid–connected neutral point campled converters

Since the end of the 90's decade and the beginning of the new millennium, multilevel converters have become one of the main focuses of power electronics research as an excellent solution for big–scale power conversion in medium and high–voltage systems. Topologies such as neutral point clamped (NPC) converters, modular multilevel converters, and modular multilevel matrix converters have been proposed for high–power drives [15], high voltage direct current transmission systems [23], integration of renewable energy sources to distribution and transmission systems [24, 25], and for the realization of soft open points in medium voltage distribution networks [26], to name a few applications. One of the most popular topologies, widely used for medium–voltage and high–power applications, is the three–level NPC (3L–NPC) converter [15]. The small amount of components, the simplicity of the topology and its operation, the flexibility to incorporate different sources through its split dc–link, and the fast dynamic response achievable are the main reasons why this type of converter has been a success on the market [27, 28].

Incorporating different high–power energy sources into the power system comes with specific requirements. First, the grid currents must have a low harmonic content, meeting grid codes as IEEE Std. 519–2022 [1] and IEEE Std. 1547–2018 [9], so a filter is usually connected between the power converter and the grid, as shown in Fig. 2.1. The second fundamental requirement is reducing the losses related to the power flow from and to the grid, as a high–efficiency performance is mandatory. To this end, operation at low switching frequencies is desired, as it reduces the switching losses of the semiconductors. However, lowering the switching frequency is usually related to an increment in the harmonic distortion of the systems' currents. For this reason, LCL filters are preferred over L filters as the attenuation of the harmonic components is higher, enabling the operation at low switching frequencies while still meeting the grid codes [29].

Even though LCL filters reduce the grid current's harmonic distortion, the large transient responses introduced by the resonances of the filter have made it necessary to incorporate different strategies to dampen the oscillations caused by the resonances. There are two main resonance frequencies [30], determined as

$$\omega_{\text{res},1} = \frac{1}{\sqrt{L_g C}} \qquad \omega_{\text{res},2} = \sqrt{\frac{L + L_g}{L L_g C}} \tag{2.1}$$

Passive damping strategies [31], based on adding passive elements as resistors, are usually discarded

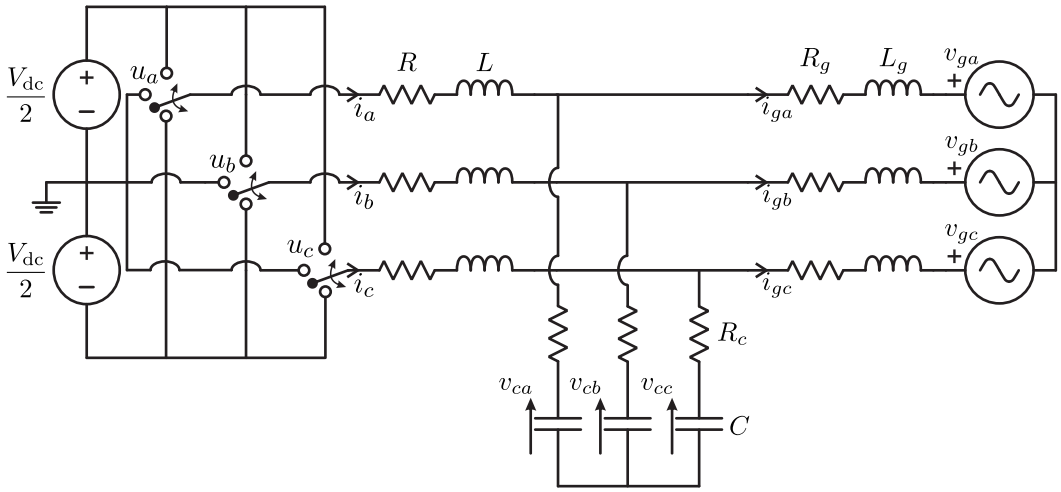


Figure 2.1: Circuit schematic of an LCL filter grid–connected three–level NPC converter.

as they increase the system's complexity and filter's cost, compromising the general efficiency. On the other hand, active damping strategies [32], which only require additional measurements and computation time, have been widely considered for both linear control methods [33] and MPC strategies [34], proving their effectiveness in reducing the impact of the filter resonances.

Even though active damping methods properly achieve the attenuation of the resonances, they increase the complexity of the control system. On the other hand, long prediction horizon MPC strategies not only enhance the performance of the system when compared with those of short horizon [19] but also remove the necessity of incorporating active damping methods as they naturally deal with the dynamics of LCL filters [35, 2]. Because of this reason, a long prediction horizon strategy as MFCS controllers is used in this work.

2.2 Multistep finite–control–set MPC

As stated in the previous section, MFCS control is a long prediction horizon MPC strategy that allows dealing with high–order systems. However, in its current form, it presents some advantages and disadvantages that will be introduced in this section.

2.2.1 Model and control problem

To develop an MFCS controller, it is assumed that the system under study can be modeled in continuous–time as

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{F}\boldsymbol{x}(t) + \boldsymbol{G}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= \boldsymbol{C}\boldsymbol{x}(t), \end{aligned} \tag{2.2}$$

where $\boldsymbol{x}$ is the state–space vector of length n_x, $\boldsymbol{u}$ is the input of the system of length n_u, and $\boldsymbol{y}$ is the output vector of length n_y. $\boldsymbol{F}$ is the continuous–time state matrix, $\boldsymbol{G}$ is the continuous–time input matrix, and $\boldsymbol{G}$ is the output matrix. Through exact discretization [36], considering a sampling period T_s, the discrete–time model of the system is expressed as

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k) \\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k), \end{aligned} \tag{2.3}$$

where $\boldsymbol{A} = e^{\boldsymbol{F}T_s}$ and $\boldsymbol{B} = -\boldsymbol{F}^{-1}\left(\boldsymbol{I} - \boldsymbol{A}\right)\boldsymbol{G}$, with $\boldsymbol{I}$ being the identity matrix of the corresponding order.

The main concept behind this type of controller is that, at each instant k, the optimal actuation $\boldsymbol{u}(k)$ is found by solving an optimization problem that minimizes a cost function. This cost function is usually composed of multiple objectives, where the principal one is to track references $\boldsymbol{y}^\star$ for the system's output. In MFCS controllers, the cost function considers a prediction horizon of $N > 1$ steps, i.e., the system's behavior has to be predicted using the model (2.3) N times.

Formally speaking, in real–time, the state variables are sampled and controlled periodically, meaning that the control problem is solved at each instant k, every T_s seconds. The problem is formulated over a prediction horizon of N steps, i.e., the optimal positions of the switches are calculated for instants k to $k+N-1$, and the system's behavior is predicted for instants $k+1$ to $k+N$. The vector comprising the predicted outputs at instant k is defined as

$$\boldsymbol{Y}(k) = \left[\boldsymbol{y}^\top(k+1) \ \dots \ \boldsymbol{y}^\top(k+N)\right]^\top,$$

whereas the vector of predicted switches positions is

$$\boldsymbol{U}(k) = \left[\boldsymbol{u}^\top(k) \ \dots \ \boldsymbol{u}^\top(k+N-1)\right]^\top.$$

The respective reference vector $\boldsymbol{Y}^\star(k)$ is defined accordingly. MFCS controllers work under the receding horizon principle, which means that, even though the whole vector $\boldsymbol{U}(k)$ is determined at each instant, only its first element, i.e., $\boldsymbol{u}(k)$, is actually applied to the system.

Traditionally, the MFCS problem is a multi–objective optimization problem, with one objective $J_y^\lambda\left(\boldsymbol{Y}(k)\right)$ related to tracking $\boldsymbol{Y}^\star(k)$ and one objective $J_u^\lambda\left(\boldsymbol{U}(k)\right)$ that directly depends on the actuation. Defining the cost function involves setting a set of weighting factors $\boldsymbol{\lambda}$, which are parameters of J_y^λ and J_u^λ and define the relative priority between the different objectives. Additionally, some operational constraints are included. This problem is usually expressed as

$$\begin{aligned}
\min_{\boldsymbol{U}(k),\boldsymbol{Y}(k)} \quad & J = J_y^\lambda\left(\boldsymbol{Y}(k)\right) + J_u^\lambda\left(\boldsymbol{U}(k)\right) \\
\text{s.t.} \quad & \\
& \boldsymbol{x}(\ell+1) = \boldsymbol{A}\boldsymbol{x}(\ell) + \boldsymbol{B}\boldsymbol{P}\boldsymbol{u}(\ell) \qquad \forall \ell \in \{k \dots k+N-1\} && \text{(2.4a)} \\
& \boldsymbol{y}(\ell) = \boldsymbol{C}\boldsymbol{x}(\ell) \qquad \forall \ell \in \{k+1 \dots k+N\} && \text{(2.4b)} \\
& \boldsymbol{x}(k) = \boldsymbol{x}_0 && \text{(2.4c)} \\
& \|\Delta\boldsymbol{u}(\ell)\|_\infty \leq 1\,, \quad \forall \ell \in \{k \dots k+N-1\} && \text{(2.4d)} \\
& \boldsymbol{U}(k) \in \mathbb{U}, && \text{(2.4e)}
\end{aligned}$$

where the model (2.3) is included as a constraint, $\boldsymbol{x}_0$ is the vector of measured states at instant k, the infinity norm is defined as

$$\|\boldsymbol{\varepsilon}\|_\infty := \max_i |\boldsymbol{\varepsilon}_i|, \tag{2.5}$$

whereas $\Delta\boldsymbol{u}(\ell)$ is defined as

$$\Delta\boldsymbol{u}(\ell) = \boldsymbol{u}(\ell) - \boldsymbol{u}(\ell-1),$$

so (2.4d) is imposed to avoid large steps in the applied voltage, and $\mathbb{U} \triangleq \boldsymbol{\mathcal{U}} \times \dots \times \boldsymbol{\mathcal{U}}$ is the N–times cartesian product of the set $\boldsymbol{\mathcal{U}}$, where $\boldsymbol{\mathcal{U}}$ is the feasible set of $\boldsymbol{u}$.

By following the same procedure done in [18], a problem that is equivalent to (2.4) can be written as

$$\begin{aligned}
\min_{\boldsymbol{U}(k)} \quad & J = \boldsymbol{U}(k)^\top \boldsymbol{Q}\boldsymbol{U}(k) + 2\boldsymbol{\Theta}^\top \boldsymbol{U}(k) \\
\text{s.t.} \quad & \\
& \|\Delta\boldsymbol{u}(\ell)\|_\infty \leq 1 \quad \forall \ell \in \{k, \dots, k+N-1\} \\
& \boldsymbol{U}(k) \in \mathbb{U},
\end{aligned} \tag{2.6}$$

where all constraints besides (2.4d) and (2.4e) are considered now in the cost function J, $\boldsymbol{Q}$ is constant and by definition a symmetric positive definite matrix when all weighting factors are greater than zero, and $\boldsymbol{\Theta}$ is updated at each control step. Both $\boldsymbol{Q}$ and $\boldsymbol{\Theta}$ depend on the specific form of the cost function and are obtained by expressing J using the model

$$\boldsymbol{Y}(k) = \boldsymbol{\Gamma}\boldsymbol{x}_0 + \boldsymbol{\Upsilon}\boldsymbol{U}(k), \tag{2.7}$$

which results from defining each element of $\boldsymbol{Y}(k)$ using the discrete–time model (2.3) along the whole prediction horizon, and the matrices $\boldsymbol{\Gamma}$ and $\boldsymbol{\Upsilon}$ are constant and obtained during the derivation.

By completing squares in J, it results in

$$J = \left(\boldsymbol{Q}^{-1}\boldsymbol{\Theta} + \boldsymbol{U}(k)\right)^\top \boldsymbol{Q}\left(\boldsymbol{Q}^{-1}\boldsymbol{\Theta} + \boldsymbol{U}(k)\right), \tag{2.8}$$

from where it can be noticed that

$$\boldsymbol{U}_{\text{unc}}(k) = -\boldsymbol{Q}^{-1}\boldsymbol{\Theta}(k) \tag{2.9}$$

is the unconstrained solution of (2.6). Given that $\mathbf{Q}$ is symmetrical and positive definite, it can be expressed as

$$\boldsymbol{H}^{\top}\boldsymbol{H} = \boldsymbol{Q}, \tag{2.10}$$

where $\boldsymbol{H}$ is a non–singular and lower triangular matrix, derived from the Cholesky decomposition of $\boldsymbol{Q}$. Subsequently, by replacing (2.10) and (2.9) in (2.8), (2.6) can be written as an integer least squares (ILS) problem as

$$\begin{aligned} \min_{\boldsymbol{U}(k)} \; & J = \left(\bar{\boldsymbol{U}}(k) - \bar{\boldsymbol{U}}_{\text{unc}}(k)\right)^{\top}\left(\bar{\boldsymbol{U}}(k) - \bar{\boldsymbol{U}}_{\text{unc}}(k)\right) \\ \text{s.t.} \quad & \\ & \bar{\boldsymbol{U}}(k) = \boldsymbol{H}\boldsymbol{U}(k) \\ & \|\Delta\boldsymbol{u}(\ell)\|_{\infty} \leq 1 \quad \forall \ell \in \{k, \ldots, k+N-1\} \\ & \boldsymbol{U}(k) \in \mathbb{U}, \end{aligned} \tag{2.11}$$

where $\boldsymbol{H}$ is understood as a transformation matrix, called lattice generator. With this transformation, J becomes the squared distance between $\bar{\boldsymbol{U}}(k)$ and $\bar{\boldsymbol{U}}_{\text{unc}}(k)$, i.e.,

$$J = \|\boldsymbol{H}\boldsymbol{U}(k) - \bar{\boldsymbol{U}}_{\text{unc}}(k)\|_2^2,$$

where the domain of $\bar{\boldsymbol{U}}(k)$ is the truncated lattice generated by the transformation of $\mathbb{U}$. This problem, as explained next, can be solved in real–time using the SDA.

2.2.2 Solution of the control problem: sphere decoder algorithm

For single–step prediction horizons and a small number of voltage levels at the converter's output, the exhaustive search method has been widely used to solve (2.4) as it is a small combinatorial problem. This method consists in enumerating all possible combinations to determine the corresponding cost function value and selecting the one that renders the smallest value. Therefore, it is not well suited for the long–horizon MFCS problem as the number of possible combinations increases exponentially with the number of steps in the prediction horizon, resulting in an extremely high computational burden.

Reducing the number of candidates evaluated before finding the solution is mandatory for real–time control applications, as control periods are short and the computational resources in commercial control platforms are limited. In this line, branch–and–bound techniques arise as a promising type of method as they allow discarding many of the candidates when evaluating the related cost function at each node in their exploration tree. Specifically, during the last decade, the sphere decoder algorithm (SDA) has been the most studied method to solve the long–horizon MFCS problem.

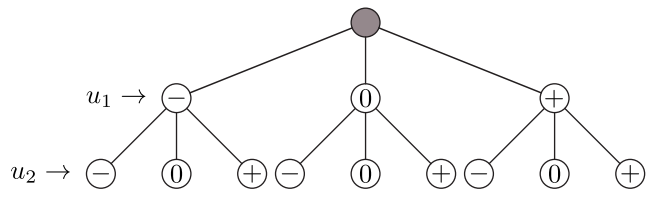


Figure 2.2: Tree structure representing the possible combinations for $\boldsymbol{U}(k)$. The node types ⊖ , ⓪ , and ⊕ mean $u_j = -1$, $u_j = 0$, and $u_j = 1$ respectively.

The SDA is used to solve the ILS problem (2.11) as it efficiently deals with its combinatorial nature by taking advantage of its structure. First, let η be a parameter defining the number of levels at the converter's output as $(2\eta + 1)$, i.e., $\eta = 1$ for a 3L–NPC converter. Then, as an example, consider a prediction horizon of $N = 5$ and a 3L–NPC converter. There are $n_u = 3$ inputs, thus, $\boldsymbol{U}(k)$ has $n_u \cdot N = 15$ entries, where each entry can take $|\mathcal{U}| = 2\eta + 1 = 3$ different values. The full set of possible combinations for $\boldsymbol{U}(k)$ has a cardinality of $(2\eta+1)^{n_u \cdot N}$, and can be represented through a search tree of depth $n_u \cdot N$, as in the example shown in Fig. 2.2, which is a case where $n_u = 1$, $N = 2$, and $\mathcal{U} = \{-1, 0, 1\}$. Each entry is denoted as u_j, so $\boldsymbol{U}(k) = [u_1, \ldots, u_{n_u \cdot N}]$. A node in the tree at level j represents u_j being equal to one of the possible values in $\mathcal{U}$. All nodes are connected to other nodes at adjacent levels

through branches. The root node (colored in gray) is not an entry of $\boldsymbol{U}(k)$ and is represented to show the connection of the first–level nodes. Each path of $n_u \cdot N$ nodes, going from the first to the last level, so all nodes in the path are exclusively at different levels, represents an option of $\boldsymbol{U}(k)$. Hence, to find a candidate, the SDA must traverse the full depth of the search tree.

The SDA is initialized with a feasible solution $\boldsymbol{U}_{\text{ini}}$ (initial incumbent). The traditional approach usually considers an educated guess as the initial solution [18]. This educated guess is generated by shifting the previous solution in one step and repeating the last control action. This can be expressed as

$$\boldsymbol{U}_{\text{eg}}(k) = \begin{bmatrix} \mathbf{0} & \boldsymbol{I_3} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{I_3} & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \dots & \dots & \mathbf{0} & \boldsymbol{I_3} \\ \mathbf{0} & \dots & \dots & \mathbf{0} & \boldsymbol{I_3} \end{bmatrix} \boldsymbol{U}_{\text{opt}}(k-1). \tag{2.12}$$

The SDA only considers candidates that are located inside a sphere of radius ρ_{inc} that is centered around $\bar{\boldsymbol{U}}_{\text{unc}}$, i.e., a new candidate $\boldsymbol{U}(k)$ must comply with

$$\|\boldsymbol{H}\boldsymbol{U}(k) - \bar{\boldsymbol{U}}_{\text{unc}}(k)\|_2^2 < \rho_{\text{inc}}^2. \tag{2.13}$$

The radius depends on the incumbent $\boldsymbol{U}_{\text{inc}}$ according to

$$\rho_{\text{inc}} = \|\boldsymbol{H}\boldsymbol{U}_{\text{inc}} - \bar{\boldsymbol{U}}_{\text{unc}}(k)\|_2, \tag{2.14}$$

where the initial radius is ρ_{ini}, which results from evaluating $\boldsymbol{U}_{\text{ini}}$ in (2.14). To look for candidates, the SDA starts searching from the first level ($j = 1$), and advances to higher levels until it reaches the top if all the nodes in the path are feasible and inside the sphere (depth–first strategy). To discard combinations located outside the sphere, when a path gets to a node at level j, a partial radius ρ_j is defined as

$$\rho_j^2 = \rho_{j-1}^2 + \left(\boldsymbol{H}_{[j,1:j]}\boldsymbol{U}_j - \bar{u}_{\text{unc},j}(k)\right)^2, \tag{2.15}$$

where ρ_{j-1} is the partial radius of that path at level $j-1$ with $\rho_0 = 0$, and $\boldsymbol{\Xi}_{[a:b,c:d]}$ is the sub–matrix of $\boldsymbol{\Xi}$ composed by the intersection of rows a to b with columns c to d. $\boldsymbol{U}_j$ is the vector generated by the path until level j, and $\bar{u}_{\text{unc},j}$ is the j^{th} element of $\bar{\boldsymbol{U}}_{\text{unc}}(k)$. Recall that $\boldsymbol{H}$ is a lower triangular matrix, so $\boldsymbol{H}_{[j,1:j]}$ are the only non–zero elements in the j^{th} row of $\boldsymbol{H}$. If $\rho_j < \rho_{\text{inc}}$ and (2.4d) is satisfied, the SDA advances to level $j+1$; if not, the branch is pruned, the paths springing from that node are discarded, and the SDA backtracks to level $j-1$. Then, it evaluates the next node in j that is connected to the node at $j-1$. If no more options are available at j, it backtracks again and repeats the process. If the algorithm reaches the last level, meaning that $\rho_{n_u \cdot N} < \rho_{\text{inc}}$, the radius of the sphere is reduced by updating it as $\rho_{\text{inc}} = \rho_{n_u \cdot N}$, and the related candidate $\boldsymbol{U}_{n_u \cdot N}$ becomes the new incumbent (i.e., $\boldsymbol{U}_{\text{inc}} = \boldsymbol{U}_{n_u \cdot N}$) until a better candidate is found or the algorithm finishes, which happens when no more options are available or the explored nodes' limit is reached. This limit is defined according to the computation time available. Once the algorithm is finished, the current incumbent becomes the solution (i.e., $\boldsymbol{U}_{\text{opt}} = \boldsymbol{U}_{\text{inc}}$). The SDA is shown in Algorithm 1, where vectors $\boldsymbol{\eta}$ and $\boldsymbol{\rho}^2$, of respective lengths $n_u \cdot N$ and $n_u \cdot N + 1$, are defined to keep track of the paths explored and the partial radius of the path, respectively, $\boldsymbol{u}_{k-1}$ is used to evaluate (4.15d), and `Max_Nodes` is the explored nodes' limit.

2.2.3 Multistep finite–control–set for LCL filter grid–connected converters

Formulating the MFCS problem as in (2.11) allows using the SDA in the way it is described through Algorithm 1 to find the optimal solution. The SDA has turned long–horizon MFCS controllers into a viable option for real–time applications. As described in [19], for high–order systems, such as LCL filter grid–connected converters, long horizons render a considerable enhancement in performance in comparison with short horizons, with the second benefit that active damping strategies are not required. In this line, extending the prediction horizon is essential to improve the performance of this type of controller for high–order systems. Additionally, control frequencies must be high to get a fine granularity to modify the actuation, rendering faster dynamic responses. In general, the prediction horizon used for three–level converters operating at a sampling period of $25\mu s$ is

Algorithm 1 Sphere decoder algorithm

function $\boldsymbol{U}_{\text{opt}} = \text{SDA}(\bar{\boldsymbol{U}}_{\text{unc}}, \boldsymbol{U}_{\text{ini}}, \boldsymbol{u}(k-1))$
  $j = 1$
  nodes_left $= 2\eta + 1$
  set each element in $\boldsymbol{\eta}$ to $-\eta$
  set each element in $\boldsymbol{\rho}^2$ to 0
  $\rho^2_{\text{inc}} \leftarrow \|\boldsymbol{H}\boldsymbol{U}_{\text{ini}} - \bar{\boldsymbol{U}}_{\text{unc}}(k)\|_2^2$ ▷ see (2.14)
  $\boldsymbol{U}_{\text{inc}} \leftarrow \boldsymbol{U}_{\text{ini}}$
  for nodes = 1 to `Max_Nodes` **do**
    nodes_left −= 1
    $u_j \leftarrow \boldsymbol{\eta}_j$
    $\boldsymbol{\rho}^2_{j+1} \leftarrow \boldsymbol{\rho}^2_j + \left(\boldsymbol{H}_{[j,1:j]}\boldsymbol{U}_j - \bar{u}_{\text{unc},j}\right)^2$ ▷ see (2.15)
    if $\boldsymbol{\rho}^2_{j+1} < \rho^2_{\text{inc}}$ and $\boldsymbol{U}_j$ meets (4.15d) **then**
      if $j = n_u \cdot N$ **then**
        $\boldsymbol{U}_{\text{inc}} \leftarrow \boldsymbol{U}_j$
        $\rho^2_{\text{inc}} \leftarrow \boldsymbol{\rho}^2_{j+1}$
        if nodes_left $= 0$ and $\boldsymbol{\eta}_j = \eta$ **then**
          break
        end if
      else
        j += 1
        $\boldsymbol{\eta}_j = -1$
        nodes_left += $2\eta + 1$
      end if
    else
      if nodes_left $= 0$ **then**
        break
      end if
      if $\boldsymbol{\eta}_j < \eta$ **then**
        η_j += 1
      else
        while true **do**
          j −= 1
          if $\boldsymbol{\eta}_j < \eta$ **then**
            $\boldsymbol{\eta}_j$ += 1
            break
          end if
        end while
      end if
    end if
  end for
  $\boldsymbol{U}_{\text{opt}} \leftarrow \boldsymbol{U}_{\text{inc}}$
end function

no larger than $N = 5$ [37] because of limited computational resources. Therefore, reducing the SDA computation time is highly desired if additional strategies that improve the steady–state performance need to be included, or the prediction horizon has to be extended.

On the other hand, [19] also highlights that MFCS controllers, in their current form, are not suitable for grid–connected converters as they do not impose a periodic pattern, resulting in asymmetrical voltages between the three phases and currents with continuous and non–deterministic harmonic spectra. This is a severe issue for grid–connected applications, as grid currents must meet grid codes related to the amplitude of individual harmonics. Additionally, a fixed switching frequency is desired, given that a variable switching frequency usually leads to conservative decisions related to the filter's design, the converter's dimensioning, and the semiconductors used. On the other hand, losses related to high switching frequencies of the semiconductors considerably hinder the power transmission efficiency, increasing the operational cost of the system. In this regard, low switching frequencies are mandatory in medium–voltage high–power applications as high

switching frequencies together with high currents would lead not only to a low efficiency but may also damage the converter unless it is designed conservatively, which would considerably increase its cost.

One modulation method that achieves fixed and low switching frequency, together with symmetric periodical voltages, is OPPs, offline calculated voltage waveforms that minimize the harmonic distortion introduced by the converter. Used as modulators, OPPs result in an outstanding steady–state performance; however, the dynamic response is slow and driven by the natural modes of the system.

This work aims to leverage the benefits of the real–time feasibility of long horizon MFCS and its fast dynamic response, with the excellent steady–state performance of OPPs.

2.3 Optimal pulse patterns

OPPs are periodic voltage waveforms designed for steady–state operation, consisting of a sequence of voltage steps at specific angular positions, called switching angles (SAs). OPPs are usually calculated to be applied as the output voltage of a power converter, where the pattern, $u(\theta)$, results from normalizing that voltage waveform, $v(\theta)$, as

$$u(\theta) = \frac{v(\theta)}{\frac{V_{\text{dc}}}{2}}, \tag{2.16}$$

where, without loss of generality, it is assumed that the converter's topology presents a DC link voltage V_{dc}. For simplicity, the traditional approach to OPPs assumes a pattern with half and quarter–wave symmetry, with only positive values in the first half–wave (unipolar pattern). Because of the half–wave symmetry and the periodicity of the pattern (2π periodicity in terms of angular position), it is held that

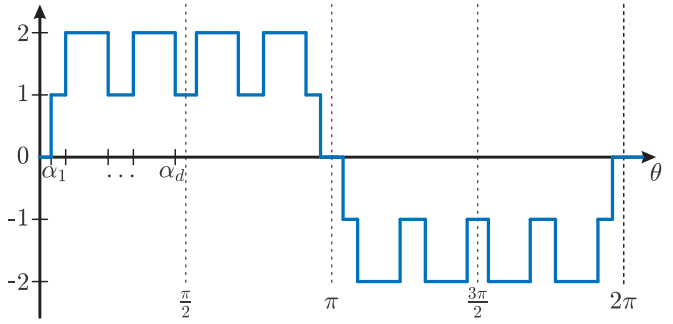


Figure 2.3: Multilevel OPP with half and quarter–wave symmetry.

$$u(\theta) = -u(\pi + \theta) \tag{2.17a}$$
$$u(0) = u(2\pi) = 0, \tag{2.17b}$$

while, because of the quarter–wave symmetry, the following holds:

$$u(\theta) = u(\pi - \theta), \tag{2.18}$$

A multilevel pattern with $2\eta + 1$ levels, $\eta = 2$, and all the traditional assumptions considered is shown in Fig. 2.3. As can be noticed from Fig. 2.3, because of (2.17) and (2.18), the pattern is fully described by its definition in the interval $\left[0, \frac{\pi}{2}\right]$. To define the pattern, let d be the number of switching transitions in a quarter–wave. In each switching transition, $u(\theta)$ changes from u_{i-1} to u_i, where $u_i \in \mathcal{U}, \forall\, i \in \{0, \ldots, d\}$, and $\mathcal{U} = \{-\eta, -\eta+1, \ldots, 0, \ldots, \eta-1, \eta\}$. Let $\mathcal{D}$ be the set defined as $\mathcal{D} := \{1, \ldots, d\}$, so each SA $\alpha_i \in [0, \frac{\pi}{2}]$ defines the angular position of the switching transition $\delta_i = u_i - u_{i-1}, \forall i \in \mathcal{D}$, and where δ_i is limited to have an amplitude of one, so

$$\delta_i \in \{-1, 1\}. \tag{2.19}$$

Notice that, as $u_0 = 0$, u_i can be expressed as

$$u_i = \sum_{j=1}^{i} \delta_j, \quad \forall i \in \mathcal{D}, \tag{2.20}$$

where, naturally, the following must hold:

$$\sum_{j=1}^{i} \delta_j \leq \eta, \quad \forall i \in \mathcal{D}. \tag{2.21}$$

The OPP $u(\theta)$ is a periodic function, meaning it can be defined through its Fourier series. Because of the half–wave symmetry, even harmonics are zero, whereas there are no cosine parts in the harmonic components because of the quarter–wave symmetry, which leads to the expression

$$u(\theta) = \sum_{h=1}^{\infty} \hat{u}_h \sin(\theta h), \tag{2.22}$$

where

$$\hat{u}_h = \begin{cases} \frac{4}{\pi h \eta} \sum_{i \in \mathcal{D}} \delta_i \cos(\alpha_i h), & h \text{ odd} \\ 0, & h \text{ even} \end{cases} \tag{2.23}$$

The three–phase patterns are generated by shifting a single–phase pattern by $\frac{2\pi}{3}$ and $\frac{4\pi}{3}$ radians. Hence, the triplen harmonics of the single–phase pattern are common–mode voltage components that do not generate currents in a balanced system with isolated neutral points. Therefore, the only components capable of producing currents in that type of system are the fundamental and the harmonics of order

$$h = 6j \pm 1, \quad j \in \mathbb{Z}_{>0}. \tag{2.24}$$

Finally, let $\mathcal{H}$ be the set of cardinality H comprising the order of all consecutive harmonics that satisfy (2.24), starting with $h = 5$, i.e., $\mathcal{H} = \{5, 7, 11, \dots\}$.

2.3.1 Traditional formulation

The main objective of OPPs is to, given a fixed value of d, determine the SAs that minimize some metric of merit for a specific modulation index in the range of $[0, \frac{4}{\pi}]$, assuming that $d \geq \eta$, and where the modulation index m is defined as

$$m = \frac{\hat{v}_1}{\frac{V_{\text{dc}}}{2}}. \tag{2.25}$$

The traditional formulation assumes that the system under study can be modeled as a purely inductive load, so it considers the total demand distortion (TDD) of a current circulating through a purely inductive load L as the metric of merit. By applying (2.23), the TDD of such a current is defined as

$$I_{\text{TDD}} = \frac{\sqrt{2} V_{\text{dc}}}{\pi \eta I_{\text{nom}} \omega_1 L} \sqrt{g_L(\boldsymbol{\delta}_d, \boldsymbol{\alpha}_d)}, \tag{2.26}$$

where $\boldsymbol{\delta}_d := \{\delta_i : i \in \mathcal{D}\}$ is the set of switching transitions, $\boldsymbol{\alpha}_d := \{\alpha_i : i \in \mathcal{D}\}$ is the set of SAs, I_{nom} is the nominal RMS value of the current, ω_1 is the fundamental angular frequency of the system, and $g_L(\boldsymbol{\alpha}_d)$ is defined as

$$g_L(\boldsymbol{\delta}_d, \boldsymbol{\alpha}_d) = \sum_{h \in \mathcal{H}} \left(\frac{1}{h^2} \sum_{i \in \mathcal{D}} \delta_i \cos(\alpha_i h) \right)^2. \tag{2.27}$$

Notice how $g_L(\boldsymbol{\alpha}_d)$ is the only term in (2.26) that is not constant and depends on $\boldsymbol{\alpha}_d$, so, for a specific modulation index, $g_L(\boldsymbol{\delta}_d, \boldsymbol{\alpha}_d)$ is chosen as the cost function in the traditional formulation.

As the OPP problem is defined for a specific modulation index, the interval $[0, \frac{4}{\pi}]$ is sampled $M-1$ times, generating a set $\mathcal{M} := \{1, \dots, M-1\}$, so $m_k = \frac{4k}{M\pi}, \forall k \in \mathcal{M}$. The optimization problem is formulated independently for each m_k, thus, for specific m and d, the SAs $\boldsymbol{\alpha}_d$ must satisfy

$$\frac{4}{\pi \eta} \sum_{i \in \mathcal{D}} \delta_i \cos(\alpha_i) = m \tag{2.28}$$

to guarantee the required m. Additionally, the SAs are bounded to be in the interval $[0, \frac{\pi}{2}]$ while preserving their sequence, which is written as the constraint

$$0 \leq \alpha_1 \leq \cdots \leq \alpha_d \leq \frac{\pi}{2}. \tag{2.29}$$

The resulting optimization problem defines a set of switching transitions $\boldsymbol{\delta}_d$ and angles $\boldsymbol{\alpha}_d$ for each $m_k, k \in \mathcal{M}$. To solve the problem, a finite number of harmonics H must be selected for the

cost function $g_L(\boldsymbol{\delta}_d, \boldsymbol{\alpha}_d)$ defined in (2.27). Once both M and H are set, the following optimization problem must be solved $M-1$ times:

$$\min_{\boldsymbol{\delta}_d, \boldsymbol{\alpha}_d} \; g_L(\boldsymbol{\delta}_d, \boldsymbol{\alpha}_d) \tag{2.30a}$$

$$\text{s.t. (2.19), (2.21), (2.28) and (2.29).} \tag{2.30b}$$

Notice (2.30) is a mixed integer nonlinear programming (MINLP) optimization problem, and its solution is out of this work's scope. For a three–level converter, i.e., $\eta = 1$, the OPPs are three–level unipolar patterns with half and quarter–wave symmetry. This means that each δ_i is now a parameter in the problem, where $\delta_i = (-1)^{i-1}$. Hence, the traditional OPP problem takes the form

$$\min_{\boldsymbol{\alpha}_d} \; g_L(\boldsymbol{\alpha}_d) \tag{2.31a}$$

$$\text{s.t. (2.28) and (2.29),} \tag{2.31b}$$

which is a nonlinear programming (NLP) optimization problem.

2.3.2 OPPs for LCL filters grid–connected converters

OPPs are calculated to minimize the TDD of a current while keeping a fixed and low switching frequency. The single pattern resulting from (2.31) is shifted by $\frac{2\pi}{3}$ and $\frac{4\pi}{3}$ radians to generate the voltages on the other two phases. Therefore, the voltages between the three phases are symmetric by definition. However, when defining I_{TDD} in (2.26) for deriving the cost function, the assumption that the system can be modeled as a purely inductive load is false in the case of an LCL filter grid–connected converter. Additionally, with the traditional formulation, it is impossible to guarantee that the grid currents resulting from operating with OPPs in steady–state will meet the grid codes required, as the harmonic components of those currents cannot be estimated using a purely inductive model. Including the LCL filter's model in the OPPs problem is mandatory so the actual TDD of the grid currents can be minimized and constraints can be added to guarantee meeting grid codes. This procedure was done in [2]; however, the authors of that work do not specify how the model was determined nor what is the resulting optimization problem. To properly incorporate OPPs in grid–connected applications, a more sophisticated model of the particular system is necessary to guarantee safe operation. In this case, the LCL filter's model is understood as a mathematical model that accurately predicts the amplitudes of the grid current's harmonic components instead, i.e., the transfer function between the grid currents and the converter output voltage. With that model, it is possible to include constraints in the optimization problem that forces the resulting OPPs to meet the grid codes in steady–state operation.

Chapter 3

Control problem and challenges

Controlling an LCL filter grid–connected converter entails dealing with many control objectives, where different objectives involve different timescales. In this chapter, the control objectives and their origin are explained. Once the control objectives and the multi–term challenges are presented, the proposed control framework is explained, showing its relevance in dealing with multi–term objectives simultaneously. Finally, the proposed control strategy is developed using this control framework.

3.1 Control objectives

This work is centered around the challenges that arise when controlling an LCL filter–grid connected 3L–NPC converter for a medium–voltage high–power application. This type of application is related to integrating different energy sources into power systems, so many regulations must be met, such as limits over certain grid currents harmonics and TDD and amplitude of interharmonics. Additionally, the system operation must be efficient and safe, so reducing the operational losses and guaranteeing a fast dynamic response is mandatory. These requirements lead to the following list of operational objectives:

- Fixed switching frequency of the converter's semiconductors, so the filter and the converter can be properly designed without making over–conservative decisions.
- Low switching frequency, to reduce the converter losses and render an efficient performance.
- Periodic and symmetric voltages between the three phases, given that grid codes demand specific harmonic components under certain amplitude limits.
- Reduced grid currents harmonic distortion, as the grid codes impose a limit over the TDD of the currents at the point of connection to the grid.
- Proper tracking of references and fast dynamic response, so the converter can properly drive the system states fast, according to the changes on the grid active and reactive power references.

Traditional control strategies usually consider only some of these control objectives simultaneously, as it is challenging to include all of them in a single control strategy. The standard approach is dealing with grid codes using offline calculated optimal patterns, whereas real–time controllers deal with proper tracking of ideal references and fast dynamic responses.

Selective harmonic mitigation (SHM) methods have been the main optimal modulation strategies that dealt with grid codes in grid–connected applications [22, 38]. This type of strategy bounds the amplitudes of the pattern harmonic components according to the grid code by adding a constraint to the problem for each specific harmonic. SHM has proved to work fine when combined with a properly designed LCL filter, even though the constraints are usually related to the voltage harmonics at the point of connection to the grid. Incorporating the dynamics of the LCL filter in the optimization problem to directly limit specific grid current harmonics has never been reported. Doing so would guarantee meeting grid codes by operating with OPPs in steady–state. In general,

when offline calculated patterns are used, their application in real–time control is based on incorporating them as modulators for other control strategies. This results in slow dynamic responses, even if the steady–state performance can comply with the rest of the control objectives.

On the other hand, real–time controllers based on MPC, as MFCS controllers, are designed to track sinusoidal references while rendering fast dynamic responses [39]. The weighting factors in the cost function are tuned to deal with the trade–off between switching frequency and harmonic distortion, trying to reduce both metrics as much as possible (Pareto optimallity). Aside from the proper tracking and fast dynamic responses, the main benefit of this type of strategy is its viability in real–time, as their implementation (and algorithms related) are well studied and, in general, are relatively easy to implement.

Control objectives are usually grouped together according to their nature. Offline calculated patterns deal with metrics such as switching frequency, TDD, and, in general, grid codes. On the other hand, real–time control strategies can adequately deal with reference tracking and fast dynamic responses. This separation is not a coincidence and is justified mainly because of the timescales involved in each objective. Metrics such as switching frequency, TDD, or harmonics amplitudes are defined by the steady–state operation of the system, meaning that at least a complete fundamental cycle has to be analyzed. Covering a whole fundamental period would require hundreds of steps using sampling periods of tens or even some hundreds of microseconds (for fundamental frequencies of 50 or 60 Hz). Using an MPC controller to that end would imply solving an optimization problem of that size in real–time, requiring a computational capacity that is out of the current technology capacities. On the other hand, offline calculated patterns can meet those requirements, as offline computation does not need to be fast and counts with larger computational resources to implement more sophisticated strategies. However, offline calculated solutions cannot deal with the tracking of references in real–time nor with the requirement of fast dynamic responses. Real–time controllers can execute those tasks, as control periods are short and can only incorporate a small number of steps into the prediction horizon, which is enough to guarantee fast dynamics responses and good reference tracking, even though it is impossible to guarantee meeting long–term objectives.

Long and short–term objectives are part of the same control problem, making it tough to develop a real–time control strategy that can address all of them simultaneously. In that line, to meet long–term objectives via real–time control, the strategy must be fed with information about the expected steady–state operation. To this end, this work formalizes a generalized framework to develop a control strategy that meets all objectives mentioned above. As a real–time control strategy is not enough to guarantee that those objectives are fulfilled, a framework incorporating an offline and a real–time optimization stage is necessary, as each stage deals with the different timescales involved in the problem. Those two entities, represented by optimal modulation methods and MPC, respectively, must be linked, even though they have worked separately in many strategies, so incorporating them in one single control framework is a natural step forward.

3.2 Solution approach: a generalized control framework

As OPPs and MFCS controllers cannot achieve the control objectives independently, it is necessary to incorporate them into a single control strategy. To this end, the control framework in Fig. 3.1 is presented, where $\boldsymbol{x}$ are the states of the system, $\boldsymbol{u}$ are the inputs, $\boldsymbol{w}$ are eventual disturbances, and a generalized system is represented through generic functions $f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w})$ and $g(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w})$, and the controller comprises two main entities: the optimization-based reference generator (OPRG), that deals with the long–term objectives, and the real–time optimization stage (ROS), that operates in real–time. The OBRG is separated in two entities: the offline stage (OPRG–OS) and the model–based real–time stage (OPRG–RS).

The full control framework operates in the following way:

- The OBRG–OS determines and stores a set of optimal policies calculated offline. These policies are the solution to a high–complexity problem that cannot be solved in real–time because of the hardware requirements, so the output $\tilde{\boldsymbol{u}}$ of this block is a control action that renders good operational indicators in the long–term. Depending on the applied strategy and its implementation, it may or may not require feedback from the system. The solutions provided by the OBRG–OS are targeted to achieve the control objectives in the long term, so, in the context of power electronics–based systems, its timescale goes from units to hundreds or even thousands

of milliseconds. Given that the control actions are determined for a large timescale, by directly applying $\tilde{\boldsymbol{u}}$, the control loop's dynamic response will generally be slow.

- The OBRG–MRS uses the optimal actuation $\tilde{\boldsymbol{u}}$ and the model of the system to generate the optimal trajectories $\tilde{\boldsymbol{x}}$ that would result, in the long–term, from applying $\tilde{\boldsymbol{u}}$ to the system. This block communicates and coordinates the optimal actions calculated offline by the OBRG–OS with the ROS, as it takes information regarding the long–term operation while running in real–time in coordination with the ROS. The optimal trajectories are generated to be used as tracking references by the ROS according to its own real–time timescale, so $\tilde{\boldsymbol{x}}$ or $\tilde{\boldsymbol{y}}$ may be passed to the ROS, where $\tilde{\boldsymbol{y}} = \boldsymbol{C}\tilde{\boldsymbol{x}}$.

- The ROS takes the optimal trajectories $\tilde{\boldsymbol{x}}$ or the tracking references $\tilde{\boldsymbol{y}}$ and includes them in the real–time optimization problem. This problem defines the control action, so it must be solvable in real–time and may consider operational indicators to be optimized together with the tracking of $\tilde{\boldsymbol{y}}$. This entity must properly track the references and guarantee a fast dynamic response. Given that the ROS solves the real–time problem, which may have a prediction horizon of a few control steps, it considers a timescale that goes from tens to hundreds of milliseconds in the power electronics context.

A detailed description of how this framework was used to develop the proposed control strategy is given in the next chapter.

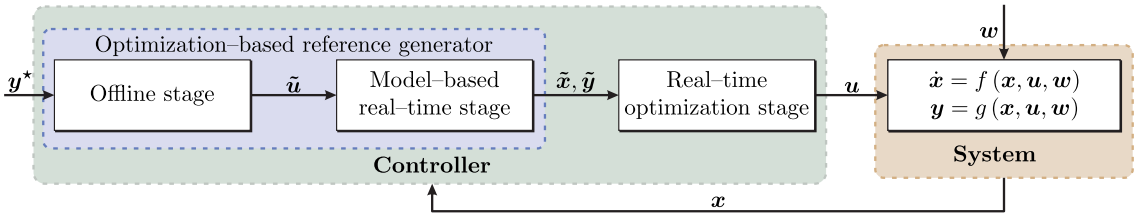


Figure 3.1: Generalized control framework proposed to develop the control strategy.

Chapter 4

Proposed control framework

The proposed control framework comprises three fundamental entities. This chapter presents the proposed execution of each entity and the link between them developed for this application. It also details the improvements done in OPPs and MFCs control to work together to meet the control objectives. Additionally, details are given regarding the efficient implementation of both OBRG–MRS and ROS so they can be viable to run in real–time using commercial control platforms.

4.1 Control strategy structure

The proposed control framework is the foundation for the proposed control strategy. The system under study is the LCL filter grid-connected 3L–NPC converter of Fig. 2.1, and the main goal is to control the active and reactive power flow at the point of connection to the grid while achieving the objectives detailed in Chapter 3. Dealing with all the objectives simultaneously made it necessary to develop a control framework capable of separating the control objectives through different entities, where all entities must operate together. The joint operation of the entities comprising the controller allows to deal with all control objectives simultaneously, which is a crucial difference from traditional approaches, where the main focus is only on the real–time controllers. The specific approach through which the proposed framework was applied to the studied system is illustrated in Fig. 4.1. A detailed mathematical and theoretical development of each entity is given in further sections, but, in general, each element is described as follows:

- OBRG–OS: The offline optimization stage, which is part of the OBRG is based on OPPs. Solving the OPPs NLP optimization problem results in a set of SAs and harmonic content that can be stored in LUTs to be used for real–time control. The traditional formulation for calculating OPPs deals with the objective of a fixed and low semiconductors switching frequency; however, minimizing the TDD of the grid currents and meeting the grid codes is not possible because the system is modeled as a purely inductive load. To this end, it is necessary to incorporate the model of the LCL filter in the optimization problem to directly minimize the TDD of the grid currents and, based on that model, add constraints over their harmonic components so the grid codes are met.

  The resulting SAs and the harmonic content of the patterns are stored in LUTs indexed by the modulation index. A strong grid is assumed to calculate the required modulation index, and a phase–locked–loop (PLL) is used to determine the angular position of the grid voltages. Notice that the ideal references $\boldsymbol{y}^\star$ are sinusoidal signals under the strong grid assumption. Naturally, the ideal references $\boldsymbol{y}^\star$ can be directly mapped from active and reactive power references $P^\star$ and $Q^\star$, respectively, using phasors theory. The SAs ($\boldsymbol{\alpha}_d$) and the harmonic content of the pattern ($\tilde{\boldsymbol{u}}$) corresponding to that modulation index are passed to the OBRG–MRS, as shown in Fig. 4.1.

- OBRG–MRS: The main goal of the OBRG–MRS is to generate the tracking references $\tilde{\boldsymbol{Y}}$ and $\tilde{\boldsymbol{U}}$ to be followed by the ROS. To this end, it uses the SAs and the harmonic components of the pattern received from the OBRG–OS. The tracking references $\tilde{\boldsymbol{Y}}$ are the system output's trajectories that would be obtained if the OPPs were directly applied by the converter in steady–state. They are generated from the harmonic components passed by the OBRG–OS. On the

other hand, $\tilde{\boldsymbol{U}}$ is generated from the SAs, where $\tilde{\boldsymbol{U}}$ is the vector resulting from sampling the corresponding OPP along the prediction horizon.

- ROS: The real–time optimization stage solves an MFCS control problem, where both $\tilde{\boldsymbol{Y}}$ and $\tilde{\boldsymbol{U}}$ are tracked, so the ideal sinusoidal references are not part of the ROS. Notice that the elements in $\tilde{\boldsymbol{Y}}$ are generated using the OPPs from the OBRG–OS, which are pulsed waveforms and not sinusoidal. Hence, the references tracked have harmonic content, even though it was minimized by the OBRG–OS. Keep in mind that power converters cannot generate a purely sinusoidal voltage as the possible states of the output voltage are discrete. OPPs consider this discrete nature, meaning that references $\tilde{\boldsymbol{Y}}$ can be perfectly tracked if the respective OPP can be perfectly generated in real–time. Tracking feasible trajectories is a fundamental element in the proposed controller, in contrast with traditional approaches, which attempt to track sinusoidal references, even though it is impossible. In any case, recall that the fundamental component of the elements in $\tilde{\boldsymbol{Y}}$ are the sinusoidal references that compose $\boldsymbol{Y}^{\star}$.

  Notice how the information from the OBRG–OS regarding the steady–state operation is passed to the ROS, where $\tilde{\boldsymbol{Y}}$ would be the output of the system if $\tilde{\boldsymbol{U}}$ is directly applied. This means that perfectly tracking both $\tilde{\boldsymbol{Y}}$ and $\tilde{\boldsymbol{U}}$ would result in the same steady–state performance of OPPs, i.e., the system would operate at a low and fixed switching frequency while meeting the grid codes.

  Recall that, as stated in Chapter 2, the ROS's prediction horizon must be as long as possible to obtain a good steady–state performance, meaning that any implementation improvements will positively impact its performance.

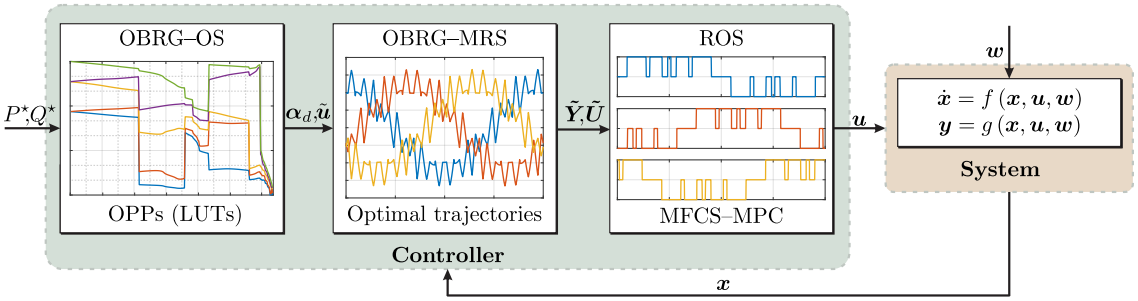


Figure 4.1: Control strategy developed based on the proposed generalized framework.

4.2 OBRG–OS: optimal pulse patterns

As previously stated, the OBRG–OS is an offline calculation stage determining OPPs that minimize the grid currents TDD and guarantee meeting grid codes by modifying and extending the traditional formulation presented in Chapter 2. The same assumptions of half and quarter–wave symmetry were considered. Even though dropping those assumptions may lead to better results, they are kept for simplicity and without loss of generality. However, generalized approaches, as in [20], can be easily incorporated.

4.2.1 Incorporation of the LCL filter model

As detailed in Section 2.3.1, the traditional approach assumes that the system under study can be approximately modeled as a purely inductive load. This assumption can be a good approximation to the real system's dynamics in cases such as electrical drives. However, when the system differs considerably from that specific model, it will lead to suboptimal solutions, which is the case of LCL grid–connected converters, as in Fig. 2.1. Given that an LCL filter is a third–order system, a first–order model as a purely inductive line would not be a good approximation, neglecting the effect of the filter's capacitor.

In the context of grid–connected converters, the main interest regarding the harmonics introduced by the converter is to reduce the grid current distortion, as it affects the active and reactive power at the point of connection to the grid. In grid codes [1, 9], these distortions are measured

through the TDD of those currents. Hence, the objective in this case is to minimize the TDD of the grid currents. The TDD of a single current is defined as

$$I_{g,\mathrm{TDD}} = \frac{1}{\sqrt{2} I_{\mathrm{nom}}} \sqrt{\sum_{h \neq 1} \hat{i}_{g,h}^2}, \tag{4.1}$$

where $\hat{i}_{g,h}$ is the amplitude of the h^{th} grid current harmonic. To incorporate this metric into the OPP optimization problem, the $I_{g,\mathrm{TDD}}$ has to be expressed in terms of the harmonic components of the resulting pattern. To this end, the transfer function $T_{gv}(s)$, that relates the grid currents with the converter voltage is required. When evaluating $T_{gv}(s)$ at $s = j\omega_1 h$, the amplitude $\hat{i}_{g,h}$ of the h current component can be expressed as

$$\hat{i}_{g,h} = \frac{\hat{v}_h}{|T_{gv}(h)|}, \tag{4.2}$$

where $T_{gv}(h)$ takes the form

$$T_{gv}(h) = \frac{\nu + j\chi}{1 + j\varsigma} \tag{4.3}$$

and

$$\begin{aligned} \nu &= R + R_g - \left[(L + L_g) C R_c + (R_g L + R L_g) C \right] \omega_1^2 h^2 \\ \chi &= \left[(R R_c + R_g R_c + R R_g) C + L + L_g \right] \omega_1 h - L L_g C \omega_1^3 h^3 \\ \varsigma &= C R_c \omega_1 h. \end{aligned} \tag{4.4}$$

By replacing (2.23) and (4.2) in (4.1), the expression for $I_{g,\mathrm{TDD}}$ takes the form

$$I_{g,\mathrm{TDD}} = \frac{4 V_{\mathrm{dc}}}{\sqrt{2} I_{\mathrm{nom}} \pi} \sqrt{g\left(\boldsymbol{\alpha}_d\right)}, \tag{4.5}$$

where

$$g\left(\boldsymbol{\alpha}_d\right) = \sum_{h \in \mathcal{H}} \left(\frac{1}{|T_{gv}(h)| h} \sum_{i \in \mathcal{D}} \delta_i \cos\left(\alpha_i h\right) \right)^2. \tag{4.6}$$

By substituting $g_L\left(\boldsymbol{\alpha}_d\right)$ by $g\left(\boldsymbol{\alpha}_d\right)$ as the cost function in (2.31), i.e.,

$$\min_{\boldsymbol{\alpha}_d} \; g\left(\boldsymbol{\alpha}_d\right) \tag{4.7a}$$

$$\text{s.t. (2.28) and (2.29)}, \tag{4.7b}$$

OPPs that actually minimize the TDD of the grid currents are obtained. In this case, the cost function depends on the nominal frequency of the system ω_1, which is a key difference from the traditional approach. By simplifying the model of the system to an equivalent inductance, the solution obtained is independent of the frequency at which the patterns will be used and on the value of the equivalent inductance; however, the resulting patterns will be suboptimal. In this case, the patterns depend on both the frequency and parameters of the system, but the resulting patterns are optimal.

4.2.2 Grid code constraints

Many technical standards define maximum allowable amplitudes for specific harmonic components of the currents at the point of connection to the power grid. Given that $T_{gv}(h)$ was derived, it is possible to include constraints in the optimization problem so the resulting OPPs guarantee meeting the grid codes regarding the amplitude of those harmonics. To this end, let $\mathcal{H}_c$ be the set of harmonic orders h whose amplitude is limited to a maximum value $i_h^{\max}$ by the grid codes. Given that v_h has only sine components because of the symmetry imposed to the pattern, according to (2.23), its amplitude $\hat{v}_h$ can be positive or negative, the constraints that guarantee meeting the grid codes can be written as

$$-i_h^{\max} \leq \frac{2 V_{\mathrm{dc}}}{|T_{gv}(h)| \pi h} \sum_{i \in \mathcal{D}} \delta_i \cos\left(\alpha_i h\right) \leq i_h^{\max}, \; \forall h \in \mathcal{H}_c. \tag{4.8}$$

By incorporating (4.6) and (4.8) into a unique optimization problem, the resulting OPPs will minimize the TDD of the grid currents at the same time that grid codes are being met. That optimization problem is formulated as

$$\min_{\boldsymbol{\alpha}_d} \; g\left(\boldsymbol{\alpha}_d\right) \tag{4.9a}$$

$$\text{s.t. (2.28), (2.29), and (4.8).} \tag{4.9b}$$

By solving (4.9), the OBRG–OS takes charge of minimizing $I_{g,\mathrm{TDD}}$ and meeting grid codes at steady–state operation. The SAs and the harmonic components of the patterns resulting from (4.9) are stored in LUTs for each $m_k, k \in \mathcal{M}$.

4.2.3 Connection with the OBRG–MRS

The SAs are stored in LUTs indexed by the modulation index. To calculate the reference modulation index $m^\star$, the procedure below is followed. For this procedure consider all variables and parameters in per unit, where the base values of the system are

$$\begin{aligned} S_B = S_{\mathrm{nom}} \quad & V_B = \sqrt{\frac{2}{3}} V_g \quad I_B = \sqrt{2} I_{\mathrm{nom}} \quad \omega_B = \omega_g \\ S_B = \frac{3}{2} V_B I_B \quad & Z_B = \frac{V_B}{I_B} \quad L_B = \frac{Z_B}{\omega_B} \quad C_B = \frac{1}{\omega_B Z_B}, \end{aligned} \tag{4.10}$$

where ω_g is the grid angular frequency, S_{nom} is its rated apparent power, V_g is its rated RMS line–to–line voltage, and I_{nom} is the rated RMS value of the grid phase current.

The reference for the apparent power $\mathbf{S}^\star$ is calculated as

$$\mathbf{S}^\star = p^\star + jq^\star$$

so the phasor of the grid current reference $\mathbf{I}_g^\star$ can be determined form the grid voltage's phasor $\mathbf{V}_g = 1$ as

$$\mathbf{I}_g^\star = \mathrm{conj}\left(\frac{\mathbf{S}^\star}{\mathbf{V}_g}\right),$$

where the operator conj() is the complex conjugate operator. Following the same procedure, the references for the capacitor voltage $\mathbf{V}_c^\star$, the converter current $\mathbf{I}^\star$ and the converter voltage $\mathbf{V}^\star$ are equal to

$$\begin{aligned} \mathbf{V}_c^\star &= \frac{1}{1 + jR_cC}\left(\left(R_g + jL_g\right)\mathbf{I}_g^\star + \mathbf{V}_g\right) \\ \mathbf{I}^\star &= jC\mathbf{V}_c^\star + \mathbf{I}_g^\star \\ \mathbf{V}^\star &= \left(R + jL\right)\mathbf{I}^\star + \mathbf{V}_c^\star, \end{aligned}$$

so $m^\star$ is results in

$$m^\star = \frac{|\mathbf{V}^\star|}{\frac{V_{\mathrm{dc}}}{2}}. \tag{4.11}$$

Then $m^\star$ is used as input of the LUTs to get the associated SAs and harmonic components that are passed to the OBRG–MRS.

4.3 ROS: multistep finite control set

Before explaining how the OBRG–MRS works, it is relevant to understand the operation of the ROS, as the OBRG–MRS generates the references for the ROS optimization problem. The proposed ROS is based on MFCS control; however, some modifications and enhancements were done from the traditional formulation so it can incorporate the information of the steady–state operation of the OPPs and find the optimal real–time solution more efficiently. Additionally, a variation over how the prediction horizon is sampled in time is proposed to extend the prediction horizon in time at no additional real–time computational burden.

4.3.1 Model of the system

The system under study is the one in Fig. 2.1, modeled in continuous–time as

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{F}\boldsymbol{x}(t) + \boldsymbol{G}\boldsymbol{u}_{\alpha\beta}(t) \\ \boldsymbol{y}(t) &= \boldsymbol{C}\boldsymbol{x}(t), \end{aligned} \tag{4.12}$$

where $\boldsymbol{\xi}_{abc} = [\xi_a\, \xi_b\, \xi_c]^\top$, $\boldsymbol{\xi}_{\alpha\beta} = [\xi_\alpha\, \xi_\beta]^\top$, $\boldsymbol{\xi}_{\alpha\beta} = \boldsymbol{P}\boldsymbol{\xi}_{abc}$ and $\boldsymbol{P}$ is the well known Clarke's transformation matrix. The state–space vector is defined as $\boldsymbol{x} = [i_\alpha\, i_\beta\, i_{g\alpha}\, i_{g\beta}\, v_{c\alpha}\, v_{c\beta}\, v_{g\alpha}\, v_{g\beta}]^\top$, the input vector is the vector of the converter's equivalent switches positions of the phases, so $\boldsymbol{u}_{abc} \in \boldsymbol{\mathcal{U}}$, where $\boldsymbol{\mathcal{U}} \triangleq \mathcal{U} \times \mathcal{U} \times \mathcal{U}$, and $\mathcal{U} := \{-1, 0, 1\}$. The output is $\boldsymbol{y} = [i_\alpha\, i_\beta\, i_{g\alpha}\, i_{g\beta}\, v_{c\alpha}\, v_{c\beta}]^\top$, and all variables are expressed in per unit. The matrices $\boldsymbol{F}$, $\boldsymbol{G}$ and $\boldsymbol{C}$ are defined as:

$$\begin{aligned} \boldsymbol{F} &= \omega_B \begin{bmatrix} -\frac{R+R_c}{L}\boldsymbol{I}_2 & \frac{R_c}{L}\boldsymbol{I}_2 & -\frac{1}{L}\boldsymbol{I}_2 & \boldsymbol{0}_2 \\ \frac{R_c}{L_g}\boldsymbol{I}_2 & -\frac{R_g+R_c}{L_g}\boldsymbol{I}_2 & \frac{1}{L_g}\boldsymbol{I}_2 & -\frac{1}{L_g}\boldsymbol{I}_2 \\ \frac{1}{C}\boldsymbol{I}_2 & -\frac{1}{C}\boldsymbol{I}_2 & \boldsymbol{0}_2 & \boldsymbol{0}_2 \\ \boldsymbol{0}_2 & \boldsymbol{0}_2 & \boldsymbol{0}_2 & \boldsymbol{M}_g \end{bmatrix} \\ \boldsymbol{G} &= \frac{V_{\mathrm{dc}}\omega_B}{2V_B}\begin{bmatrix} \frac{1}{L}\boldsymbol{I}_2 & \boldsymbol{0}_{6\times 2} \end{bmatrix}^\top \\ \boldsymbol{C} &= [\boldsymbol{I}_6 \;\; \boldsymbol{0}_{6\times 2}] \\ \boldsymbol{M}_g &= \begin{bmatrix} 0 & -\omega_g \\ \omega_g & 0 \end{bmatrix}, \end{aligned} \tag{4.13}$$

where ω_g is the grid frequency, $\boldsymbol{I}_n$ is the identity matrix of order n, $\boldsymbol{0}_{n\times m}$ is a zero matrix with n rows and m columns, and $\boldsymbol{0}_n$ is a squared zero matrix of size n.

Through exact discretization, based on a sampling period T_s, the discrete–time model of the system is expressed as

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}_{\alpha\beta}(k) \\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k), \end{aligned} \tag{4.14}$$

where $\boldsymbol{A} = e^{\boldsymbol{F}T_s}$ and $\boldsymbol{B} = -\boldsymbol{F}^{-1}\left(\boldsymbol{I}_8 - \boldsymbol{A}\right)\boldsymbol{G}$.

4.3.2 Control problem

The procedure to formulate the control problem is similar to the traditional approach presented in Chapter 2. One key difference is that the proposed MFCS controller does not track the ideal references. In this case, the objective is tracking both $\tilde{\boldsymbol{Y}}(k)$ and $\tilde{\boldsymbol{U}}(k)$ while considering the same operational constraints as in the traditional formulation. This problem is expressed as

$$\min_{\boldsymbol{U}(k),\boldsymbol{Y}(k)} \;\; J_y^\lambda\left(\boldsymbol{Y}(k)\right) + J_u^\lambda\left(\boldsymbol{U}(k)\right)$$

s.t.

$$\boldsymbol{x}(\ell+1) = \boldsymbol{A}\boldsymbol{x}(\ell) + \boldsymbol{B}\boldsymbol{P}\boldsymbol{u}_{abc}(\ell) \qquad \forall \ell \in \{k \dots k+N-1\} \tag{4.15a}$$

$$\boldsymbol{y}(\ell) = \boldsymbol{C}\boldsymbol{x}(\ell) \qquad \forall \ell \in \{k+1 \dots k+N\} \tag{4.15b}$$

$$\boldsymbol{x}(k) = \boldsymbol{x}_0 \tag{4.15c}$$

$$\|\Delta\boldsymbol{u}_{abc}(\ell)\|_\infty \leq 1 \quad \forall \ell \in \{k \dots k+N-1\} \tag{4.15d}$$

$$\boldsymbol{U}(k) \in \mathbb{U}. \tag{4.15e}$$

In this case, the multi–objective function is composed by

$$\begin{aligned} J_y^\lambda\left(\boldsymbol{Y}(k)\right) &= \sum_{\ell=k+1}^{k+N} \|\tilde{\boldsymbol{y}}(\ell) - \boldsymbol{y}(\ell)\|^2_{\boldsymbol{\lambda}_\ell^y} \\ J_u^\lambda\left(\boldsymbol{U}(k)\right) &= \sum_{\ell=k}^{k+N-1} \|\tilde{\boldsymbol{u}}_{abc}(\ell) - \boldsymbol{u}_{abc}(\ell)\|^2_{\boldsymbol{\lambda}_\ell^u}, \end{aligned} \tag{4.16}$$

where

$$\|\boldsymbol{\varepsilon}\|_{\boldsymbol{\lambda}} := \sqrt{\boldsymbol{\varepsilon}^\top \boldsymbol{\lambda}\boldsymbol{\varepsilon}}, \tag{4.17}$$

$J_u^{\lambda}\left(\boldsymbol{U}(k)\right)$ considers the tracking of the pattern in a similar way as in [40], and $\boldsymbol{\lambda}$ is a square positive definite matrix with the corresponding dimension. The weighting factor matrices $\boldsymbol{\lambda}_\ell^y$ and $\boldsymbol{\lambda}_\ell^y$, defined for each step, are

$$\boldsymbol{\lambda}_\ell^y = \operatorname{diag}\left\{\lambda_\ell^i,\, \lambda_\ell^i,\, \lambda_\ell^g,\, \lambda_\ell^g,\, \lambda_\ell^c,\, \lambda_\ell^c\right\} \tag{4.18}$$

$$\boldsymbol{\lambda}_\ell^u = \lambda_\ell^u \boldsymbol{I}_3, \tag{4.19}$$

where λ_ℓ^u is the weighting factor of the switches positions tracking error, λ_ℓ^i is the weight of the converter currents tracking error, λ_ℓ^c is the one of the capacitors voltages and λ_ℓ^g the grid currents one.

As explained in Chapter 2, problem (4.15) can be written as:

$$\begin{aligned} \min_{\boldsymbol{U}(k)} \;& J = \boldsymbol{U}(k)^\top \boldsymbol{Q}\boldsymbol{U}(k) + 2\boldsymbol{\Theta}^\top \boldsymbol{U}(k) \\ \text{s.t.} \;& \\ & \|\Delta \boldsymbol{u}_{\boldsymbol{abc}}(\ell)\|_\infty \leq 1 \quad \forall \ell \in \{k, \ldots, k+N-1\} \\ & \boldsymbol{U}(k) \in \mathbb{U}. \end{aligned} \tag{4.20}$$

The derivation process of (4.20) requires defining the following matrices:

$$\boldsymbol{Q} = \boldsymbol{\Upsilon}^\top \boldsymbol{M}_y \boldsymbol{\Upsilon} + \boldsymbol{M}_u \tag{4.21}$$

$$\boldsymbol{M}_y = \operatorname{diag}\left\{\boldsymbol{\lambda}_{k+1}^y, \ldots, \boldsymbol{\lambda}_{k+N}^y\right\} \tag{4.22}$$

$$\boldsymbol{M}_u = \operatorname{diag}\left\{\boldsymbol{\lambda}_{k}^u, \ldots, \boldsymbol{\lambda}_{k+N-1}^u\right\} \tag{4.23}$$

$$\boldsymbol{\Upsilon} = \begin{bmatrix} \boldsymbol{CB} & & & \\ \boldsymbol{CAB} & \boldsymbol{CB} & & \\ \vdots & \vdots & \ddots & \\ \boldsymbol{CA}^{N-1}\boldsymbol{B} & \boldsymbol{CA}^{N-2}\boldsymbol{B} & \cdots & \boldsymbol{CB} \end{bmatrix} \tag{4.24}$$

$$\boldsymbol{\Theta} = \boldsymbol{\Upsilon}^\top \boldsymbol{M}_y \boldsymbol{\Gamma} \boldsymbol{x}_0 - \boldsymbol{\Upsilon}^\top \boldsymbol{M}_y \tilde{\boldsymbol{Y}}(k) - \boldsymbol{M}_u \tilde{\boldsymbol{U}}(k) \tag{4.25}$$

$$\boldsymbol{\Gamma} = \begin{bmatrix} (\boldsymbol{CA})^\top & \ldots & \left(\boldsymbol{CA}^N\right)^\top \end{bmatrix}^\top, \tag{4.26}$$

When its arguments are matrices, the operator $\operatorname{diag}\{\ldots\}$ generates a matrix with blocks of elements corresponding to the respective matrix argument, and all other elements are zeros.

Once the problem is written as in (4.20), it can be noticed how the prediction horizon can be extended in time without increasing the number of steps considered, therefore, at no additional computational cost. This is achieved by modifying the sampling time along the prediction horizon, as explained in the next section.

Finally, it is important to write the proposed formulation in the standard ILS form used to solve the problem using the SDA so it is clear that the proposed MFCS optimization problem is viable in real–time. This is written as

$$\begin{aligned} \min_{\boldsymbol{U}(k)} \;& J = \|\boldsymbol{H}\boldsymbol{U}(k) - \bar{\boldsymbol{U}}_{\text{unc}}(k)\|_2^2 \\ \text{s.t.} \;& \\ & \|\Delta \boldsymbol{u}_{\boldsymbol{abc}}(\ell)\|_\infty \leq 1 \quad \forall \ell \in \{k, \ldots, k+N-1\} \\ & \boldsymbol{U}(k) \in \mathbb{U}. \end{aligned} \tag{4.27}$$

4.3.3 Variable–step–size predictive model

MFCS–MPC is based on predicting the system's behavior through a horizon of N steps, where a step is understood as each of the future instants at which the state variables are predicted. To this end, a discrete–time model as (4.14) is used to evaluate the system's performance for different inputs at each of the N steps.

The control system running in real–time is usually driven by a type of processor unit such as a micro–controller or a DSP. This processor unit is in charge of synchronizing and coordinating the measurements of the system variables, the execution of the control algorithm, and the implementation of the control decisions. This synchronization process is done by periodically interrupting this processor unit with a period T_s, frequently called sampling or control period, depending on

whether the measurements are synchronized with the control actions or not. This is the main reason why the traditional approach of MFCS discretizes the horizon with N instants equally separated by a fixed interval T_s. Recall that MFCS–MPC operates according to the receding horizon principle, which means that, even though the control actions are calculated for the whole prediction horizon, only the control actions of the first step are applied, repeating the procedure on every control period. When using MPC, namely MFCS–MPC, for control of high–order systems (third order or above), long prediction horizons render considerable benefits in terms of both dynamic and steady–state performance [19]. This allows to expect that, by extending the timespan covered by the prediction horizon, the performance can be improved to some extent. However, extending the prediction horizon by enlarging the number of steps entails exponential growth in terms of the computational burden related to solving the control problem in real–time. Another option to look further ahead in time is by increasing the sampling period T_s, so each step covers a larger timespan. This procedure allows to cover a longer timespan with the same number of steps in the prediction horizon; however, recall that MFCS control is a direct MPC strategy, meaning that the decision variables are directly applied to the semiconductors at the end of each control period. Therefore, the control actions are held during the whole period T_s, changing only at the next step. Given that each actuation is held for a whole period, T_s must be as small as possible for the control actions to have a fine granularity. A fine granularity is required to guarantee fast dynamics responses, and also allows the resulting actuation to be similar to the original OPP in steady–state [41].

This work proposes a variable–step discretization of time to extend the timespan covered by the prediction horizon. Consider the ordered set $\mathcal{S} := \{s_1, s_2, \ldots, s_N\}$, where each element is a scaling factor of the real–time sampling period T_s. By scaling the size of each step in the prediction horizon, a new set of samplings instants $\{k', k'+1, \ldots, k'+N\}$ is generated, where instant $k' = k$. This variable–step–size manner of sampling time is illustrated in Fig. 4.2, where the upper plot shows the proposed sampling strategy, whereas the bottom plot represents the traditional regular–sampling. If all elements in $\mathcal{S}$ are greater or equal to one, the timespan covered by the prediction horizon is always larger than that of the traditional approach. It is important to recall that a high granularity of the actuation is required and that the control actions are applied for a period of T_s seconds. Thus, it is recommended to define $s_1 = 1$ so the decision taken is actually applied for the same period considered when solving the control problem.

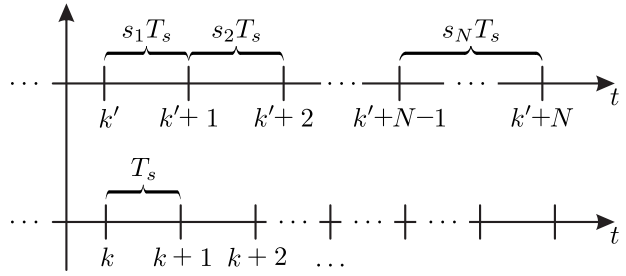


Figure 4.2: Proposed variable discretization of time.

It is important to mention that this strategy implies no additional computational burden regarding the real–time control problem. The ILS formulation in (4.27) does not change, so the problem is solvable through the same methods as in the regularly–sampled discretization case. However, the expressions for $\boldsymbol{\Upsilon}$ and $\boldsymbol{\Gamma}$ are different; therefore, the matrices $\boldsymbol{Q}$, $\boldsymbol{H}$, and $\boldsymbol{\Theta}$ also change, even though their expressions are the same. In this case, at step ℓ, the sampling period is equal to $s_\ell T_s$, which implies that, for that step, the discrete–time model is defined by the matrices $\boldsymbol{A}_\ell = e^{\boldsymbol{F} s_\ell T_s}$ and $\boldsymbol{B}_\ell = -\boldsymbol{F}^{-1}\left(\boldsymbol{I}_8 - \boldsymbol{A}_\ell\right)\boldsymbol{G}$. By considering model (2.7), the generalized expressions for $\boldsymbol{\Upsilon}$ and $\boldsymbol{\Gamma}$ are the following:

$$\boldsymbol{\Upsilon} = \begin{bmatrix} \boldsymbol{C}\boldsymbol{B}_1 & & & \\ \boldsymbol{C}\boldsymbol{A}_2\boldsymbol{B}_1 & \boldsymbol{C}\boldsymbol{B}_2 & & \\ \vdots & \vdots & \ddots & \\ \boldsymbol{C}\left(\prod_{\ell=N}^{\ell=2}\boldsymbol{A}_\ell\right)\boldsymbol{B}_1 & \boldsymbol{C}\left(\prod_{\ell=N}^{\ell=3}\boldsymbol{A}_\ell\right)\boldsymbol{B}_2 & \cdots & \boldsymbol{C}\boldsymbol{B}_N \end{bmatrix} \tag{4.28}$$

$$\boldsymbol{\Gamma} = \begin{bmatrix} (\boldsymbol{C}\boldsymbol{A}_1)^\top & (\boldsymbol{C}\boldsymbol{A}_2\boldsymbol{A}_1)^\top & \ldots & \left(\boldsymbol{C}\prod_{\ell=N}^{\ell=1}\boldsymbol{A}_\ell\right)^\top \end{bmatrix}^\top \tag{4.29}$$

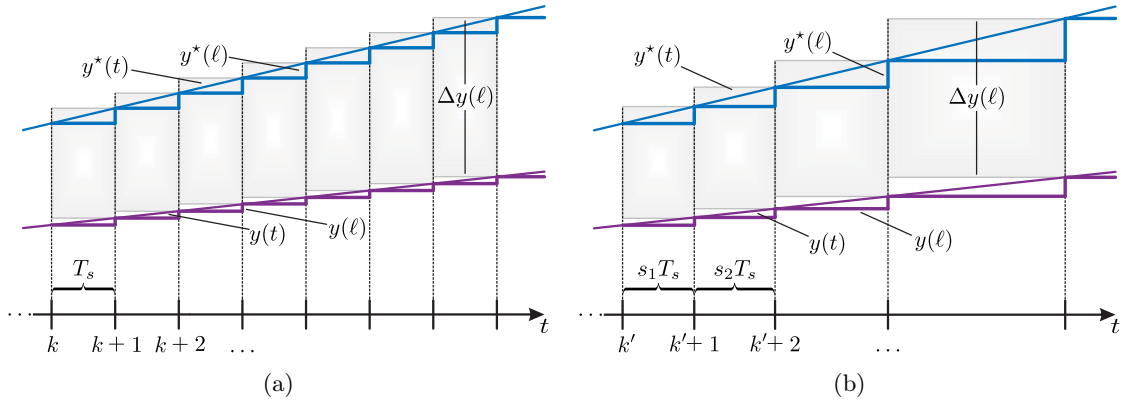


Figure 4.3: Approximation of integral for different sampling methods along a prediction horizon. (a) Traditional sampling. (b) Variable–step–size sampling.

Recall that both $\boldsymbol{\Upsilon}$ and $\boldsymbol{\Gamma}$ exclusively depend on the parameters of the system, so they are calculated offline, which means that the variable–step–size prediction horizon only affects the offline computational burden, which, in general, is not an issue. Also, it is important to notice that the matrices involved have the same size as in the original formulation and that the number of matrix multiplications does not change.

Cost function

Consider the continuous–time cost function

$$\int_{t_k}^{t_{k+N}} \Delta y(\tau)^2 d\tau, \tag{4.30}$$

where $\Delta y(t) = y^\star(t) - y(t)$, and t_k is the time at the sampling instant k. Using the integral approximation

$$\int_{t_k}^{t_k+\Delta t} f(\tau)d\tau \approx \Delta t f(t_k + \Delta t), \tag{4.31}$$

the value of this cost function can be approximated, as shown in Fig. 4.3a for a regularly–sampled time, as

$$\int_{t_k}^{t_{k+N}} \Delta y(\tau)^2 d\tau = \sum_{\iota=1}^{N} \int_{t_{k+\iota-1}}^{t_{k+\iota}} \Delta y(\tau)^2 d\tau \approx T_s \sum_{\iota=1}^{N} \Delta y(t_{k+\iota})^2, \tag{4.32}$$

where T_s can be factorized because the time is sampled regularly, meaning each rectangle in Fig. 4.3a has the same base. When defining a discrete–time cost function with the form

$$\lambda_y \sum_{\ell=k+1}^{k+N} \|\Delta y(\ell)\|^2, \tag{4.33}$$

it can be understood as the approximation process in (4.32), where the factor T_s is absorbed by the weighting factor λ_y, which is also constant. Notice that (4.32) and (4.33) are equal if $\lambda_y = T_s$. Let us define

$$\lambda_y = \lambda_y^r T_s,$$

where λ_y^r would be the "real" weighting factor of the cost function. For a constant λ_y, a larger T_s would result in a smaller λ_y^r and vice versa. Let us now consider a variable–sampled time with the scaling factors defined by the set $\mathcal{S}$, as shown in Fig. 4.3b. In this case, the continuous–time cost function takes the form

$$\int_{t_k}^{t_{k+N}} \Delta y(\tau)^2 d\tau = \sum_{\iota=1}^{N} \int_{t_{k+\iota-1}}^{t_{k+\iota}} \Delta y(\tau)^2 d\tau \approx T_s \sum_{\iota=1}^{N} s_\iota \Delta y(t_{k+\iota})^2, \tag{4.34}$$

where the base of each rectangle in the approximation shown in Fig. 4.3b is now $s_\iota T_s$. In this case, the discrete–time cost function (4.33) is, in general, not equal to (4.34), as λ_y is constant,

but the factor $s_\iota T_s$ is, in general, different at each step. However, (4.33) can be understood as an approximation of (4.34) in which λ_y absorbs the term $s_\iota T_s$ in each step, but at the same time it applies a "real" weight that changes depending on s_ι. A larger s_ι results in a smaller "real" weighting factor for that step, and a smaller s_ι would imply a larger "real" weight. If the intention is to have an exact approximation of (4.34), it is necessary to define a different cost function with the form

$$\sum_{\ell=k+1}^{k+N} \lambda_y^\ell \|\Delta y(\ell)\|^2, \tag{4.35}$$

where $\lambda_y^\ell = \lambda_y \cdot s_\iota$ with $\ell = k + \iota$ and $\iota \in \{1, \ldots, N\}$. Notice that this cost function is equal to the approximation (4.34) when $\lambda_y = T_s$. In this case, scaling λ_y by s_ι results in weighting factors λ_y^ℓ with equal "real" weights for all the steps in the prediction horizon. The traditional approach in MFCS–MPC is to weight all steps in the prediction horizon equally, which is simply done by setting a constant λ_y at each step in a regularly–sampled prediction horizon. However, to weight all the steps equally in a variable–sampled prediction horizon, it is necessary to scale the weighting factor at each step by the scaling factor s_ι, as setting a constant λ_y for each step would result in smaller weights for the steps with larger values of s_ι and larger weights for those with a smaller s_ι.

4.3.4 Generalized search tree exploration pattern

As noticed from Algorithm 1, in the traditional SDA, the exploration of all levels in the search tree starts with the node $u_j = -\eta$ (which is usually understood as exploring the tree from left to right, considering the traditional arrangement of the tree). This situation is illustrated in Fig. 4.4a, for a case with $n_u = 1$ and $N = 3$. The first explored candidate, if feasible, is marked in red, whereas the initial solution, located anywhere in the tree, is marked in green. The SDA initial candidate is always feasible, and, as will be analyzed in the next chapter, it is chosen to render a small cost function value. Hence, instead of starting the exploration of the tree in an arbitrary manner, as in Algorithm 1, it is proposed first to explore the zones of the tree that are close to the initial solution and paths composed of elements that are similar to those of the initial solution. This exploration pattern of the three can be understood as a rearrangement of the search tree's elements while keeping the original exploration pattern (i.e., from left to right), so the first path explored is the initial solution.

Understanding this proposed pattern as a rearrangement of the tree allows representing it as in Fig. 4.4b, where it can be noticed that the first explored path is the initial solution. The arrangement consists in modifying the order of the nodes at each level and letting the first one on the left be the respective entry in the initial solution, while the other ones can be sorted arbitrarily. In this case, for simplicity, all the nodes initially located at the left of the initial solution entry are positioned at the extreme right in the same order. For example, if the initial solution entry is of type $\oplus$, the level is sorted, from left to right, as $\left[\oplus, \ominus, \circledcirc\right]$.

To incorporate this exploration pattern into the traditional SDA, an array $\boldsymbol{O}_{\text{ep}}$ containing the order of each level is added as an input. This addition allows ordering each level as desired, bringing the flexibility to define the arrangement of the tree based on any chosen criteria. To illustrate this idea, e.g., the array $\boldsymbol{O}_{\text{ep}}$ related to the rearrangement of the search tree in Fig. 4.4b, would be

$$\boldsymbol{O}_{\text{ep}} = \begin{bmatrix} 1 & -1 & -0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix}.$$

The addition of $\boldsymbol{O}_{\text{ep}}$ as an input in Algorithm 1 only requires a few modifications to the original SDA, as shown in Algorithm 2, where the function SDA_{ep} is defined. Incorporating the exploration pattern does not increase the SDA computational cost. Notice that this modification to the SDA is general for any η, so it can be applied to any multilevel converter.

It is important to highlight that, because of the proposed exploration pattern, the first explored path is always the one of the initial candidate, which is perfectly known before starting the exploration. This is a big difference from the traditional approach. To start the traditional SDA, the initial radius ρ_{ini} has to be calculated, so, for the initial candidate, the partial radius at each level is determined. This is equivalent to evaluating $n_u \cdot N$ nodes (is not exactly the same given that the feasibility conditions are not evaluated). The traditional SDA does not use this initial information, as it always explores the tree from level $j = 1$ to $j = n_u \cdot N$, starting from the node $u_j = -\eta$. As

Algorithm 2 SDA with exploration pattern

function $\boldsymbol{U}_{\text{opt}} = \text{SDA}_{\text{ep}}(\bar{\boldsymbol{U}}_{\text{unc}}, \boldsymbol{U}_{\text{ini}}, \boldsymbol{u_{abc}}(k-1), \boldsymbol{O}_{\text{ep}})$
  $j = 1$
  nodes_left $= 2\eta + 1$
  set each element in $\boldsymbol{\eta}$ to 1
  set each element in $\boldsymbol{\rho}^2$ to 0
  $\rho^2_{\text{inc}} \leftarrow \|\boldsymbol{H}\boldsymbol{U}_{\text{ini}} - \bar{\boldsymbol{U}}_{\text{unc}}(k)\|_2^2$ ▷ see (2.14)
  $\boldsymbol{U}_{\text{inc}} \leftarrow \boldsymbol{U}_{\text{ini}}$
  for nodes = 1 to `Max_Nodes` **do**
    nodes_left −= 1
    $u_j \leftarrow \boldsymbol{O}_{\text{ep},\,[j,\boldsymbol{\eta}_j]}$
    $\boldsymbol{\rho}^2_{j+1} \leftarrow \boldsymbol{\rho}^2_j + \left(\boldsymbol{H}_{[j,1:j]}\boldsymbol{U}_j - \bar{u}_{\text{unc},j}\right)^2$ ▷ see (2.15)
    if $\boldsymbol{\rho}^2_{j+1} < \rho^2_{\text{inc}}$ and $\boldsymbol{U}_j$ meets (4.15d) **then**
      if $j = n_u \cdot N$ **then**
        $\boldsymbol{U}_{\text{inc}} \leftarrow \boldsymbol{U}_j$
        $\rho^2_{\text{inc}} \leftarrow \boldsymbol{\rho}^2_{j+1}$
        if nodes_left $= 0$ and $\boldsymbol{\eta}_j = 2\eta + 1$ **then**
          break
        end if
      else
        j += 1
        $\boldsymbol{\eta}_j = 1$
        nodes_left += $2\eta + 1$
      end if
    else
      if nodes_left $= 0$ **then**
        break
      end if
      if $\boldsymbol{\eta}_j < 2\eta + 1$ **then**
        η_j += 1
      else
        while true **do**
          j −= 1
          if $\boldsymbol{\eta}_j < 2\eta + 1$ **then**
            $\boldsymbol{\eta}_j$ += 1
            break
          end if
        end while
      end if
    end if
  end for
  $\boldsymbol{U}_{\text{opt}} \leftarrow \boldsymbol{U}_{\text{inc}}$
end function

the tree exploration advances, the algorithm will get to the initial candidate's path and re-evaluate it (unless a better candidate is found before), even though the partial radius at each level were calculated at the beginning. On the other hand, given that in the proposed exploration pattern the first path evaluated is always $\boldsymbol{U}_{\text{ini}}$, the SDA_{ep} can instantly start at level $j = n_u \cdot N$, which can reduce in $n_u \cdot N$ the nodes explored by the algorithm. From now on, the terms SDA and SDA_{ep} will be used indistinctly unless explicitly specified.

Finally, it is possible to incorporate the parallelization method presented in [42], by dividing the tree at the first level in $2\eta + 1$ sub–trees, where a different SDA explores each sub–tree. As an example, the parallelization of the search tree in Fig. 4.4b is shown in Fig. 4.5, where three SDAs operate in parallel. Notice that the SDAs can have different exploration patterns, but with the proposed one, the first path explored by SDA_1 will be the initial solution, whereas the other SDAs' first paths (marked in Fig. 4.5) will be equal to the initial solution except for the first element. With this parallelization strategy, the computation time required to find the solution can

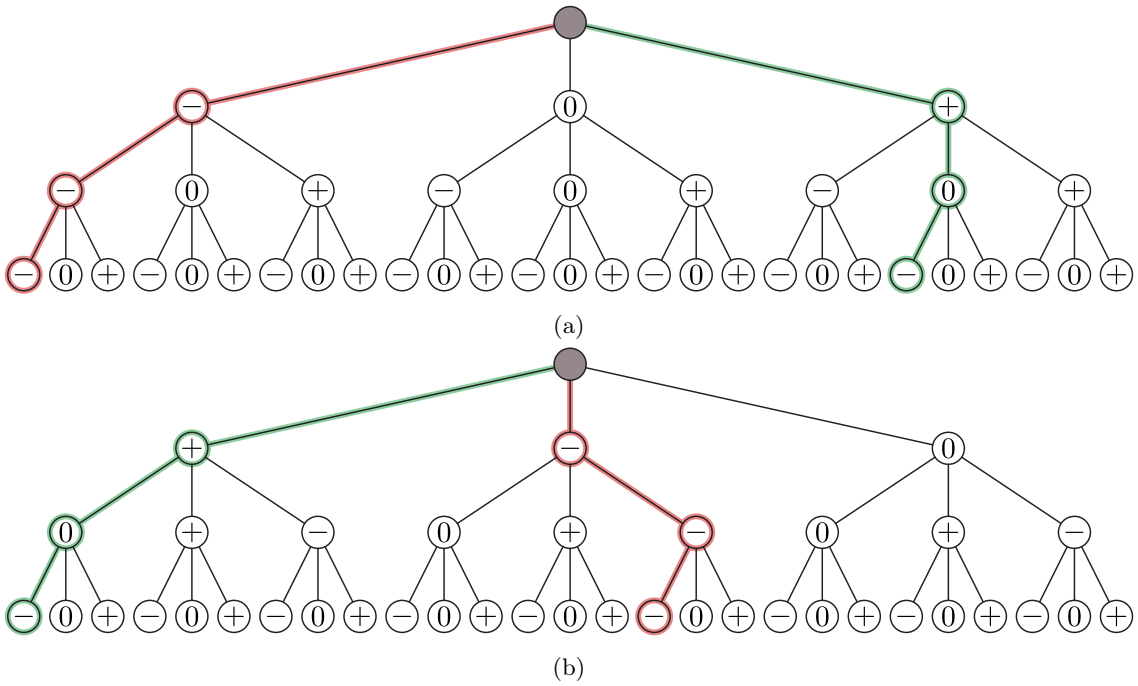


Figure 4.4: Search trees explored by the SDA in the traditional and the proposed strategies. (a) Traditional search tree explored by the SDA. (b) Proposed re–arrangement of the search tree.

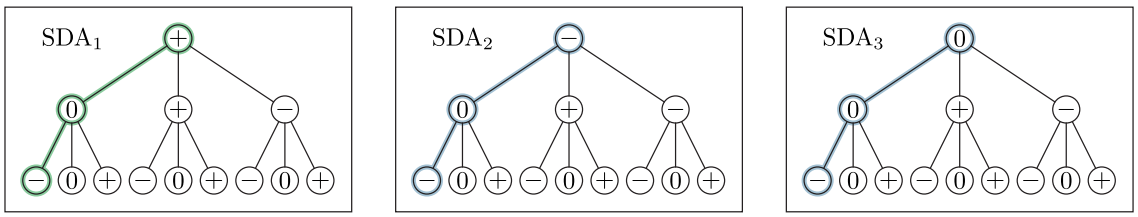


Figure 4.5: Search trees of three SDAs operating in parallel.

be considerably reduced [42].

4.3.5 Transient operation

The SDA looks for potential candidates inside a sphere, which must have a small radius to reduce the number of options so the solution can be found quickly. A small ρ_{ini} reduces the number of candidates located inside the sphere at the beginning of the algorithm in comparison with a large ρ_{ini}, so most of the time, by properly choosing $\boldsymbol{U}_{\text{ini}}$, the number of iterations required to find the solution can be considerably reduced. However, that is not always the case. Let $\mathbb{C}_{\mathbb{U}}$ be the convex–hull of $\mathbb{U}$, so $\mathbb{C}_{\mathbb{U}} := \left\{\boldsymbol{U} \in \mathbb{R}^{n_u \cdot N} : -\mathbf{1} \cdot \eta \preceq \boldsymbol{U} \preceq \mathbf{1} \cdot \eta\right\}$, where $-\mathbf{1}$ is a vector of ones of the proper length. Given that ρ_{ini} depends not only on $\boldsymbol{U}_{\text{ini}}$, but also on $\boldsymbol{U}_{\text{unc}}$, whenever $\boldsymbol{U}_{\text{unc}}$ is located far from $\mathbb{C}_{\mathbb{U}}$, the initial radius will be large, and, as detailed in [43] and [44], this frequently happens during transient operation. In those cases, independently of the initial incumbent, the number of candidates inside the sphere will be large as all points in $\mathbb{C}_{\mathbb{U}}$ are far from the center of the sphere.

In this work, the strategy used in both [43] and [44] is adopted to deal with this issue. It consists in modifying the original optimization problem every time $\boldsymbol{U}_{\text{unc}}$ is far from $\mathbb{C}_{\mathbb{U}}$ (which happens mainly during transient operation). In those cases, the optimization problem solved by

the SDA is

$$\begin{aligned} \min_{\boldsymbol{U}(k)} \;& J_{\text{mod}} = \|\boldsymbol{H}\boldsymbol{U}(k) - \bar{\boldsymbol{U}}_{\text{bc}}(k)\|_2^2 \\ \text{s.t.} \;& \\ & \|\Delta \boldsymbol{u}_{\boldsymbol{abc}}(\ell)\|_\infty \leq 1\,, \quad \forall \ell \in \{k, \ldots, k+N-1\} \\ & \boldsymbol{U}(k) \in \mathbb{U}, \end{aligned} \tag{4.36}$$

where J_{mod} is an approximation of the original cost function J, in which $\bar{\boldsymbol{U}}_{\text{unc}}$ (the center of the sphere) is replaced by $\bar{\boldsymbol{U}}_{\text{bc}} = \boldsymbol{H}\boldsymbol{U}_{\text{bc}}$, and $\boldsymbol{U}_{\text{bc}}$ is the euclidean projection of $\boldsymbol{U}_{\text{unc}}$ over $\mathbb{C}_{\mathbb{U}}$ when $\boldsymbol{U}_{\text{unc}} \notin \mathbb{C}_{\mathbb{U}}$. When $\boldsymbol{U}_{\text{unc}} \in \mathbb{C}_{\mathbb{U}}$, $\boldsymbol{U}_{\text{bc}} = \boldsymbol{U}_{\text{unc}}$. Given that (4.36) is solved only in cases when $\boldsymbol{U}_{\text{unc}} \notin \mathbb{C}_{\mathbb{U}}$, (4.36) is not the same es (4.27). The vector $\boldsymbol{U}_{\text{bc}}$ is the solution of the following box–constrained quadratic programming problem:

$$\min_{\boldsymbol{U}(k) \in \mathbb{C}_{\mathbb{U}}} \quad J = \boldsymbol{U}(k)^\top \boldsymbol{Q}\boldsymbol{U}(k) + 2\boldsymbol{\Theta}^\top \boldsymbol{U}(k), \tag{4.37}$$

which must be solved in real–time before solving (4.36) with the SDA. It is important to mention that [43] and [44] do not solve this problem when the unconstrained solution is not in $\mathbb{C}_{\mathbb{U}}$, but when it is not in the set $\left\{\boldsymbol{U} \in \mathbb{R}^{n_u \cdot N} : \boldsymbol{-2} \cdot \eta \preceq \boldsymbol{U} \preceq \boldsymbol{2} \cdot \eta\right\}$ so the problem is not modified when it is close to the borders of $\mathbb{C}_{\mathbb{U}}$. In this work, the set $\left\{\boldsymbol{U} \in \mathbb{R}^{n_u \cdot N} : \boldsymbol{-\sqrt{3}} \cdot \eta \preceq \boldsymbol{U} \preceq \boldsymbol{\sqrt{3}} \cdot \eta\right\}$ is used, which is the hypercube in which a ball circumscribing $\mathbb{C}_{\mathbb{U}}$ is inscribed.

An efficient strategy to solve (4.37) is required, as most of the available computation time must be left for the SDA to find the optimal actuation. This work uses a simple algorithm based mainly on closed–form expressions and simple operations: The Frank–Wolfe algorithm (FWA) [45]. In simple words, the FWA starts at a feasible point and then advances in the direction of a feasible set's vertex that minimizes the current gradient of the cost function. The step size at which the algorithm moves in the new direction can be determined in many ways; however, in this work, the simplest one is chosen as it has a closed–form expression that can be easily evaluated.

Formally speaking, the algorithm starts at a feasible point $\boldsymbol{U}^0$. Then, the following optimization problem is solved at each t^{th} iteration:

$$\boldsymbol{\zeta}^t \in \underset{\boldsymbol{U} \in \mathbb{C}_{\mathbb{U}}}{\operatorname{argmin}} \; (\boldsymbol{\zeta}^t)^\top \nabla J\left(\boldsymbol{U}^{t-1}\right), \tag{4.38}$$

where

$$\nabla J\left(\boldsymbol{U}^{t-1}\right) = 2\boldsymbol{Q}\boldsymbol{U}^{t-1} + 2\boldsymbol{\Theta}.$$

Notice that, if the j^{th} element of $\Lambda^t = \boldsymbol{Q}\boldsymbol{U}^{t-1} + \boldsymbol{\Theta}$ is non–negative, the j^{th} element of $\boldsymbol{\zeta}^t$ (ζ_j^t) is equal to $-\eta$, whereas, if negative, $\zeta_j^t = \eta$. Once $\boldsymbol{\zeta}^t$ is calculated, the solution is updated as

$$\boldsymbol{U}^t = (1-\gamma)\,\boldsymbol{U}^{t-1} + \gamma^t \boldsymbol{\zeta}^t, \tag{4.39}$$

where the step size γ^t is defined as

$$\gamma^t = \frac{2}{t+2}. \tag{4.40}$$

This process is repeated for $\texttt{N}_{\texttt{iter}}$ iterations, as in Algorithm 3, where the pseudocode of the FWA implementation is shown.

Algorithm 3 Frank–Wolfe algorithm

function $\boldsymbol{U}_{\mathrm{bc}} = \mathrm{FWA}(\boldsymbol{\Theta})$
  $\boldsymbol{U}^0 \leftarrow \mathbf{0}$ ▷ feasible initialization with zeros
  for t = 1 to $\mathtt{N_{iter}}$ **do**
    $\gamma^t \leftarrow \dfrac{2}{t+2}$
    $\boldsymbol{\Lambda}^t \leftarrow \boldsymbol{Q}\boldsymbol{U}^{t-1} + \boldsymbol{\Theta}$
    for j = 1 to $n_u \cdot N$ **do**
      if $\boldsymbol{\Lambda}^t_j \geq 0$ **then**
        $\boldsymbol{\zeta}^t_j = -\eta$
      else
        $\boldsymbol{\zeta}^t_j = \eta$
      end if
    end for
    $\boldsymbol{U}^t = (1-\gamma)\,\boldsymbol{U}^{t-1} + \gamma^t \boldsymbol{\zeta}^t$
  end for
  $\boldsymbol{U}_{\mathrm{bc}} \leftarrow \boldsymbol{U}^t$
end function

Finally, to keep the initial radius small, a convenient initial candidate $\boldsymbol{U}_{\mathrm{sq}}$ is calculated. Following the same procedure as in [43] and [44], this candidate is generated by quantizing $\boldsymbol{U}_{\mathrm{bc}}$ to the closest element in $\mathbb{U}$ and then applying a greedy algorithm to force the quantized vector to meet (4.15d). In a generalized case, for a large η, greedy algorithms may not lead to a feasible candidate or may require an extensive computation time [46]; however, for $\eta = 1$, the algorithm used generates $\boldsymbol{U}_{\mathrm{sq}}$ in a straightforward manner. First, $\boldsymbol{U}_{\mathrm{bc}}$ is quantized by approximating each of its entries to the closest element in $\mathcal{U}$. Then, in a sequential manner, i.e., from $j = 1$ up to $j = n_u \cdot N$, the constraint (4.15d) is evaluated by comparing the switching state with the corresponding previous one. If the entry does not comply with (4.15d), it is forced to the closest value that does, e.g., if $u_4 = 1$ and $u_1 = -1$, then u_4 is forced to $u_4 = 0$ so the amplitude of the transition is equal or smaller than one.

4.4 OBRG: model–based real–time stage

Once the ROS is established, it is possible to design an OBRG–MRS capable of communicating the OBRG–OS with the ROS so all the control objectives are met. The OBRG–MRS task is to generate the tracking references $\tilde{\boldsymbol{Y}}$ and $\tilde{\boldsymbol{U}}$ in real–time using the OPPs calculated by the OBRG–OS. The tracking references $\tilde{\boldsymbol{Y}}$ are the trajectories that the system's outputs would follow if the system operates using OPPs in steady–state. On the other hand, $\tilde{\boldsymbol{U}}$ is the OPP associated with $m^\star$, synchronized through the PLL output (as any reference) and sampled (using the zero–order–hold) along the prediction horizon, so its generation is straightforward.

The first task is to determine the initial element $\tilde{\boldsymbol{y}}(\ell = k+1)$ in the prediction horizon, which is the same as $\tilde{\boldsymbol{y}}(\theta_1)$, where $\theta_1 = \theta_{\mathrm{PLL}} + s_1\omega_g T_s$, θ_{PLL} is the angular position reference calculated by the PLL. This point is part of a steady–state trajectory resulting from periodic inputs (the OPPs), so it can be found using Fourier series and phasors theory. Consider a set $\mathcal{H}_r$ that is the subset of $\mathcal{H}$ comprising the first H_r elements of $\mathcal{H}$, and that those H_r harmonic components of the OPPs are stored for each modulation index. Recall that the OPPs' harmonics only have a sine part, so only the amplitude is needed to store each harmonic (amplitude with sign).

Then, the following impedances need to be defined:

$$\boldsymbol{z}^h = R + jLh, \quad \boldsymbol{z}_c^h = R_c - j\tfrac{1}{Ch}, \quad \boldsymbol{z}_g^h = R_g + jL_g h, \quad \boldsymbol{z}_{\mathrm{eq}}^h = \frac{\boldsymbol{z}_c^h \boldsymbol{z}_g^h}{\boldsymbol{z}_c^h + \boldsymbol{z}_g^h} \quad \forall h \in \mathcal{H}_r. \tag{4.41}$$

So, by analyzing the circuit using Kirchhoff's laws, it is possible to determine the phasors for each harmonic component of the outputs as

$$\tilde{\mathbf{I}}^h = \boldsymbol{K}^h \tilde{\boldsymbol{u}}^h, \quad \tilde{\mathbf{V}}_c^h = \boldsymbol{K}_c^h \tilde{\boldsymbol{u}}^h, \quad \tilde{\mathbf{I}}_g^h = \boldsymbol{K}_g^h \tilde{\boldsymbol{u}}^h \quad \forall h \in \mathcal{H}_r, \tag{4.42}$$

where these phasors are calculated considering the converter's output voltage angle as the angular position zero, and $\tilde{\boldsymbol{u}}^h$ is the phasor of the h^{th} harmonic component of the respective OPP (this

phasor has angle zero and its amplitude can be positive or negative). The parameters $\boldsymbol{K}^h$, $\boldsymbol{K}_c^h$ and $\boldsymbol{K}_g^h$ are defined as

$$\boldsymbol{K}^h = \frac{1}{\boldsymbol{z}^h + \boldsymbol{z}_{\text{eq}}}, \quad \boldsymbol{K}_c^h = \boldsymbol{z}_{\text{eq}}\boldsymbol{K}^h, \quad \boldsymbol{K}_g^h = \frac{\boldsymbol{K}_c^h}{\boldsymbol{z}_g^h} \quad \forall h \in \mathcal{H}_r. \tag{4.43}$$

The ideal references of the system outputs, which are the fundamental components of $\tilde{\boldsymbol{y}}(\theta_1)$, are determined as

$$\begin{aligned}
\boldsymbol{i}_{\alpha\beta}^{\star}(\theta_1) &= |\mathbf{I}^{\star}|\boldsymbol{P}\left[\begin{array}{ccc} \sin\left(\angle\underline{\mathbf{I}^{\star}} + \theta_1\right) & \sin\left(\angle\underline{\mathbf{I}^{\star}} + \theta_1 + \frac{4\pi}{3}\right) & \sin\left(\angle\underline{\mathbf{I}^{\star}} + \theta_1 + \frac{2\pi}{3}\right)\end{array}\right]^{\top} \\
\boldsymbol{i}_{g,\alpha\beta}^{\star}(\theta_1) &= |\mathbf{I}_g^{\star}|\boldsymbol{P}\left[\begin{array}{ccc} \sin\left(\angle\underline{\mathbf{I}_g^{\star}} + \theta_1\right) & \sin\left(\angle\underline{\mathbf{I}_g^{\star}} + \theta_1 + \frac{4\pi}{3}\right) & \sin\left(\angle\underline{\mathbf{I}_g^{\star}} + \theta_1 + \frac{2\pi}{3}\right)\end{array}\right]^{\top} \\
\boldsymbol{v}_{c,\alpha\beta}^{\star}(\theta_1) &= |\mathbf{V}_c^{\star}|\boldsymbol{P}\left[\begin{array}{ccc} \sin\left(\angle\underline{\mathbf{V}_c^{\star}} + \theta_1\right) & \sin\left(\angle\underline{\mathbf{V}_c^{\star}} + \theta_1 + \frac{4\pi}{3}\right) & \sin\left(\angle\underline{\mathbf{V}_c^{\star}} + \theta_1 + \frac{2\pi}{3}\right)\end{array}\right]^{\top},
\end{aligned} \tag{4.44}$$

whereas their harmonic parts correspond to

$$\begin{aligned}
\tilde{\boldsymbol{i}}_{\alpha\beta}^{H}(\theta_1) &= \boldsymbol{P}\sum_{h\in\mathcal{H}_r}|\tilde{\mathbf{I}}^h|\left[\begin{array}{ccc} \sin\left(\angle\underline{\tilde{\mathbf{I}}^h} + \theta_a^h\right) & \sin\left(\angle\underline{\tilde{\mathbf{I}}^h} + \theta_b^h\right) & \sin\left(\angle\underline{\tilde{\mathbf{I}}^h} + \theta_c^h\right)\end{array}\right]^{\top} \\
\tilde{\boldsymbol{i}}_{g,\alpha\beta}^{H}(\theta_1) &= \boldsymbol{P}\sum_{h\in\mathcal{H}_r}|\tilde{\mathbf{I}}_g^h|\left[\begin{array}{ccc} \sin\left(\angle\underline{\tilde{\mathbf{I}}_g^h} + \theta_a^h\right) & \sin\left(\angle\underline{\tilde{\mathbf{I}}_g^h} + \theta_b^h\right) & \sin\left(\angle\underline{\tilde{\mathbf{I}}_g^h} + \theta_c^h\right)\end{array}\right]^{\top} \\
\tilde{\boldsymbol{v}}_{c,\alpha\beta}^{H}(\theta_1) &= \boldsymbol{P}\sum_{h\in\mathcal{H}_r}|\tilde{\mathbf{V}}_c^h|\left[\begin{array}{ccc} \sin\left(\angle\underline{\tilde{\mathbf{V}}_c^h} + \theta_a^h\right) & \sin\left(\angle\underline{\tilde{\mathbf{V}}_c^h} + \theta_b^h\right) & \sin\left(\angle\underline{\tilde{\mathbf{V}}_c^h} + \theta_c^h\right)\end{array}\right]^{\top},
\end{aligned} \tag{4.45}$$

where

$$\theta_a^h = \left(\angle\underline{\mathbf{V}^{\star}} + \theta_1\right)h, \quad \theta_b^h = \theta_a^h + \frac{4\pi h}{3}, \quad \theta_c^h = \theta_a^h + \frac{2\pi h}{3} \quad \forall h \in \mathcal{H}_r \tag{4.46}$$

So, the tracking reference vector for the first step of the prediction horizon is

$$\tilde{\boldsymbol{y}}(\theta_1) = \left[\begin{array}{ccc} \left(\tilde{\boldsymbol{i}}_{\alpha\beta}(\theta_1)\right)^{\top} & \left(\tilde{\boldsymbol{i}}_{g,\alpha\beta}(\theta_1)\right)^{\top} & \left(\tilde{\boldsymbol{v}}_{c,\alpha\beta}(\theta_1)\right)^{\top}\end{array}\right]^{\top}, \tag{4.47}$$

where

$$\tilde{\boldsymbol{i}}_{\alpha\beta} = \tilde{\boldsymbol{i}}_{\alpha\beta}^{\star} + \tilde{\boldsymbol{i}}_{\alpha\beta}^{H}, \quad \tilde{\boldsymbol{i}}_{g,\alpha\beta} = \tilde{\boldsymbol{i}}_{g,\alpha\beta}^{\star} + \tilde{\boldsymbol{i}}_{g,\alpha\beta}^{H}, \quad \tilde{\boldsymbol{v}}_{c,\alpha\beta} = \tilde{\boldsymbol{v}}_{c,\alpha\beta}^{\star} + \tilde{\boldsymbol{v}}_{c,\alpha\beta}^{H}.$$

To determine the tracking references for the rest of the prediction horizon, the matrices $\boldsymbol{\Upsilon}_r$ and $\boldsymbol{\Gamma}_r$ are defined as

$$\begin{aligned}
\boldsymbol{\Upsilon}_r &= \left[\begin{array}{ccccc} \boldsymbol{C}\boldsymbol{B}_2 & & & & \\ \boldsymbol{C}\boldsymbol{A}_3\boldsymbol{B}_2 & & \boldsymbol{C}\boldsymbol{B}_3 & & \\ \vdots & & \vdots & & \ddots \\ \boldsymbol{C}\left(\prod_{\ell=N}^{\ell=3}\boldsymbol{A}_\ell\right)\boldsymbol{B}_2 & & \boldsymbol{C}\left(\prod_{\ell=N}^{\ell=4}\boldsymbol{A}_\ell\right)\boldsymbol{B}_3 & \cdots & \boldsymbol{C}\boldsymbol{B}_N\end{array}\right] \\
\boldsymbol{\Gamma}_r &= \left[\begin{array}{cccc} (\boldsymbol{C}\boldsymbol{A}_1 2)^{\top} & (\boldsymbol{C}\boldsymbol{A}_3\boldsymbol{A}_2)^{\top} & \ldots & \left(\boldsymbol{C}\prod_{\ell=N}^{\ell=2}\boldsymbol{A}_\ell\right)^{\top}\end{array}\right]^{\top},
\end{aligned} \tag{4.48}$$

which are derived following the same procedure to determine $\boldsymbol{\Upsilon}$ and $\boldsymbol{\Gamma}$, but starting from step $\ell = k+2$ instead of $\ell = k+1$. Notice that $\boldsymbol{\Upsilon}_r$ is a sub–matrix of $\boldsymbol{\Upsilon}$ so it is not necessary to calculate it. Hence, the tracking references are determined as

$$\tilde{\boldsymbol{Y}}(k) = \left[\begin{array}{cc} \tilde{\boldsymbol{y}}^{\top}(\theta_1) & \tilde{\boldsymbol{Y}}_r^{\top}\end{array}\right]^{\top}, \tag{4.49}$$

where

$$\tilde{\boldsymbol{Y}}_r = \boldsymbol{\Gamma}_r\tilde{\boldsymbol{x}}(\theta_1) + \boldsymbol{\Upsilon}_r\tilde{\boldsymbol{U}}_{[n_u+1:n_u\cdot N]}, \tag{4.50}$$

and

$$\begin{aligned}\tilde{\boldsymbol{x}}(\theta_1) &= \begin{bmatrix} \left(\tilde{\boldsymbol{i}}_{\alpha\beta}(\theta_1)\right)^\top & \left(\tilde{\boldsymbol{i}}_{g,\alpha\beta}(\theta_1)\right)^\top & (\tilde{\boldsymbol{v}}_{c,\alpha\beta}(\theta_1))^\top & (\boldsymbol{v}_{g,\alpha\beta}(\theta_1))^\top \end{bmatrix}^\top \\ \boldsymbol{v}_{g,\alpha\beta}(\theta_1) &= |\mathbf{V}_g|\boldsymbol{P}\begin{bmatrix} \sin(\theta_1) & \sin\left(\theta_1+\frac{4\pi}{3}\right) & \sin\left(\theta_1+\frac{2\pi}{3}\right) \end{bmatrix}^\top.\end{aligned} \tag{4.51}$$

Notice that $\tilde{\boldsymbol{Y}}_r$ is generated by using $\tilde{\boldsymbol{U}}$ as the input in (4.50), which is generated by sampling the OPP using zero–order–hold with different sampling periods along the prediction horizon, depending on $\mathcal{S}$. This means that $\tilde{\boldsymbol{U}}$ is not identical to the respective OPP, therefore $\tilde{\boldsymbol{Y}}_r$ is not the exact trajectory followed by the outputs of the system in steady–state when operating with the respective OPP. However, this is a good approximation when the step sizes considered in the variable–step–size prediction horizon are small.

4.4.1 Efficient implementation

It is easy to notice that the amount of sine operations required to generate the tracking references, in the way shown above, is large. Given that the whole control strategy must run in a short period (e.g. 25 microseconds), it is mandatory to make an efficient implementation of each part of the strategy in both time and resources. To reduce the number of sine operations it is proposed to take advantage of the fact that the OPP only has harmonic components that comply with (2.24). Let us consider $\tilde{\boldsymbol{i}}^H_{\alpha\beta}(\theta_1)$ as an example.

The expression for $\tilde{\boldsymbol{i}}^H_{\alpha\beta}(\theta_1)$ can be written as

$$\tilde{\boldsymbol{i}}^H_{\alpha\beta}(\theta_1) = \boldsymbol{P}\sum_{h\in\mathcal{H}_r}|\boldsymbol{K}^h|\tilde{\boldsymbol{u}}^h\begin{bmatrix} \sin\left(\angle\underline{\boldsymbol{K}^h}+\theta_v h\right) \\ \sin\left(\angle\underline{\boldsymbol{K}^h}+\theta_v h-\frac{2\pi h}{3}\right) \\ \sin\left(\angle\underline{\boldsymbol{K}^h}+\theta_v h+\frac{2\pi h}{3}\right)\end{bmatrix}$$

where $\theta_v = \left(\angle\underline{\mathbf{V}^\star}+\theta_1\right)$ and $h = 6j\pm 1$ for $j\in\mathbb{Z}_{>0}$. Using trigonometric identities, the following can be deduced for phase a

$$\begin{aligned}\sin\left(\angle\underline{\boldsymbol{K}^h}+\theta_v(6j\pm1)\right) =& \sin\left(\angle\underline{\boldsymbol{K}^h}\right)\Big(\cos(6j\theta_v)\cos(\theta_v)\mp\sin(6j\theta_v)\sin(\theta_v)\Big)+ \\ & \cos\left(\angle\underline{\boldsymbol{K}^h}\right)\Big(\sin(6j\theta_v)\cos(\theta_v)\pm\cos(6j\theta_v)\sin(\theta_v)\Big)\end{aligned} \tag{4.52}$$

whereas, for phases b and c, the following holds

$$\begin{aligned}\sin\left(\angle\underline{\boldsymbol{K}^h}\pm\frac{2\pi h}{3}+\theta_v h\right) =& \cos(\theta_v h)\left(\sin\left(\angle\underline{\boldsymbol{K}^h}\right)\cos\left(\frac{2\pi h}{3}\right)\pm\cos\left(\angle\underline{\boldsymbol{K}^h}\right)\sin\left(\frac{2\pi h}{3}\right)\right)+ \\ & \sin(\theta_v h)\left(\cos\left(\angle\underline{\boldsymbol{K}^h}\right)\cos\left(\frac{2\pi h}{3}\right)\mp\sin\left(\angle\underline{\boldsymbol{K}^h}\right)\sin\left(\frac{2\pi h}{3}\right)\right)\end{aligned} \tag{4.53}$$

where $|\boldsymbol{K}^h|$, $\sin\left(\angle\underline{\boldsymbol{K}^h}\right)$ and $\cos\left(\angle\underline{\boldsymbol{K}^h}\right)$ are values that can be stored in tables without requiring large memory resources. Additionally,

$$\cos\left(\frac{2\pi h}{3}\right) = -0.5 \quad \forall h\in\mathcal{H}_r, \quad \sin\left(\frac{2\pi(6j\pm1)}{3}\right) = \pm\frac{\sqrt{3}}{2}$$

so those sines and cosines do not need to be calculated.

This same procedure can be followed for $\tilde{\boldsymbol{i}}^H_{g,\alpha\beta}(\theta_1)$ and $\tilde{\boldsymbol{v}}^H_{c,\alpha\beta}(\theta_1)$, so, to generate the harmonic part of all the tracking references, only $\cos(6j\theta_v)$, $\cos(\theta_v)$, $\sin(6j\theta_v)$, and $\sin(\theta_v)$ need to be calculated online, adding up for a total of $H_r/2+2$ sine/cosine operations (naturally, considering an even value for H_r). To reduce the computational burden even more, recall that

$$\cos(\xi) = \sin\left(\xi+\frac{\pi}{2}\right)$$

so only sine operations would be required, which can be implemented using a single LUT, rendering a considerably faster and cheaper implementation.

Chapter 5

Simulation and analysis

The proposed control framework and strategy developed are the main novelties of this work. In addition, the development of the strategy led to some innovations in the area of OPPs and MFCS. In this chapter, all the improvements and contributions regarding OPPs, MFCS, and OPP–MFCS are analyzed and verified through simulation experiments. For all the experiments, unless otherwise specified, the parameters of the system are the ones in [2].

5.1 System under study

The system under study is a a high–power medium–voltage grid–connected converter as the one depicted in Fig. 2.1 and, unless specified, the parameters used in this and further chapters are the ones in [2], as specified in Table 5.1. The filter was designed so the resonances are located at 262 and 491 Hz, expecting that the switching frequency of the converter's semiconductors is around 250 Hz.

Table 5.1: Parameters of the studied system [2].

Parameter	Value	Parameter	Value
S_{nom}	9 MVA	V_g	3150 V (RMS)
I_{nom}	1649.6 A (RMS)	V_{dc}	4840 V
R	0.3 mΩ	R_g	27.51 mΩ
L	350 μH	L_g	875.6 μH
R_c	4 mΩ	C	420 μF

The system is modeled in per unit using the base values and the relations defined in Chapter 4. The sampling frequency at which the controller operates is $T_s = 25\,\mu$s, and the nominal frequency of the system is $f_g = 50\,$Hz.

5.2 Optimal pulse patterns

The traditional formulation of OPPs assumes that the system can be modeled as a purely inductive load, which is not the case for an LCL filter grid–connected converter. In this work, the model of the LCL filter was incorporated into the cost function of the optimization problem to minimize the TDD of the grid currents directly. Even though this was reported to be done in [2], the formulation of the optimization problem and analysis of the resulting patterns was never presented. Additionally, using the LCL filter model, it was possible to add constraints that guaranteed the resulting patterns to meet grid codes at a low and fixed switching frequency.

A series of simulation tests were held to evaluate and analyze these novelties. First, the traditional formulation of OPPs assuming an L model was compared with the proposed formulation incorporating the LCL filter model in the cost function. Then, the model including the constraints of the grid codes is tested, and the grid code requirements are verified. Finally, the computation time to solve the full model was compared with the model with no grid code constraints to verify its scalability.

5.2.1 Incorporation of LCL filter model

The first test is to compare the performance between the OPPs resulting from the traditional formulation (2.31), with the proposed dedicated OPPs (4.7), that incorporates the model of the LCL filter into the cost function to minimize $I_{g,\text{TDD}}$. To this end, the system under study was simulated using Matlab–Simulink with the PLECS blockset, and both THD and TDD of the grid currents were calculated. Patterns with $d = 5$ were used, implying a semiconductors switching frequency of 250 Hz. Five cases were tested for different modulation indices, resulting in the $I_{g,\text{TDD}}$ and $I_{g,\text{THD}}$ shown in Table 5.2, where the results are summarized, and column "L" corresponds to the results with OPPs from the traditional model (2.31), whereas column "LCL" are the dedicated OPPs from model (4.7). The benefits of incorporating the LCL filter model in the optimization problem's cost function to directly minimize $I_{g,\text{TDD}}$ are clear, with reductions larger than a 50% in the TDD of the grid currents for some cases. For all modulation indices, both TDD and THD of the currents are reduced considerably, where the TDD, in all cases, is smaller than 5%, which is the limit imposed by the IEEE Std. 519–2022 [1] in a system of this characteristics.

Table 5.2: L and LCL model OPPs comparison.

	$I_{g,\text{TDD}}$ (%)		$I_{g,\text{THD}}$ (%)	
m	L	LCL	L	LCL
0.8	5.92	1.31	8.82	1.95
0.9	2.01	1.20	5.02	2.99
1.0	2.99	1.03	23.23	8.03
1.1	7.02	1.67	49.11	11.67
1.2	2.85	2.45	6.87	5.90

5.2.2 Meeting grid codes

A key factor in the operation of grid–connected converters is meeting grid codes as IEEE Std. 519–2022 [1]. For the system under study, which is a medium–voltage high–power system designed mainly as a power generation facility, the amplitude limit on the harmonic content and $I_{g,\text{TDD}}$, in per unit with respect to the maximum current of the grid, are specified in Table 5.3.

Table 5.3: IEEE Std. 519–2022 [1] limits for amplitudes of harmonics and $I_{g,\text{TDD}}$.

	$2 \leq h < 11$	$11 \leq h < 17$	$17 \leq h < 23$	$23 \leq h < 35$	$35 \leq h \leq 50$	$I_{g,\text{TDD}}$
Limit (pu)	0.04	0.02	0.015	0.006	0.003	0.05

Even harmonics of order $h \leq 6$ are limited to half of the value shown in Table 5.3; however, recall that the calculated OPPs, because of quarter–wave symmetry, do not present even harmonics, and, because of the isolated neutral points, triplen order harmonic components are zero in the grid currents. Hence, only differential harmonics that meet (2.24) need to be limited. To guarantee that these limitations are met, the model (4.9) was proposed.

Because of the filter's size, the results from model (4.9) with $d = 5$ are almost identical to the ones of model (4.7), where the main difference is that when including the grid code constraints the problem becomes infeasible from $m = 1.22$, i.e., the grid codes cannot be met at a switching frequency of 250 Hz in the interval $[1.22, \frac{4}{\pi}]$. This is a significant result as it allows knowing the limits for a proper operation at that switching frequency.

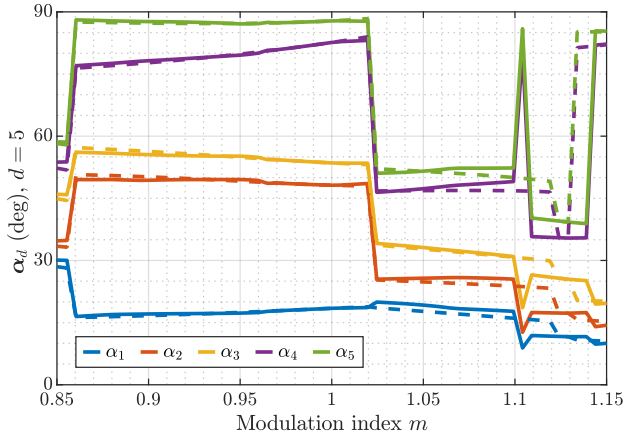


Figure 5.1: Comparison of SAs resulting from models (4.7) (dashed lines) and (4.9) (solid lines).

To show the benefits of model (4.9), the proposed formulation was tested in a modified system, in which the filter capacitor has 70% the capacitance of the original filter, resulting in resonance frequencies of 314 and 587 Hz. For this analysis, the grid code limits $i_h^{\max}$ over the harmonic components were

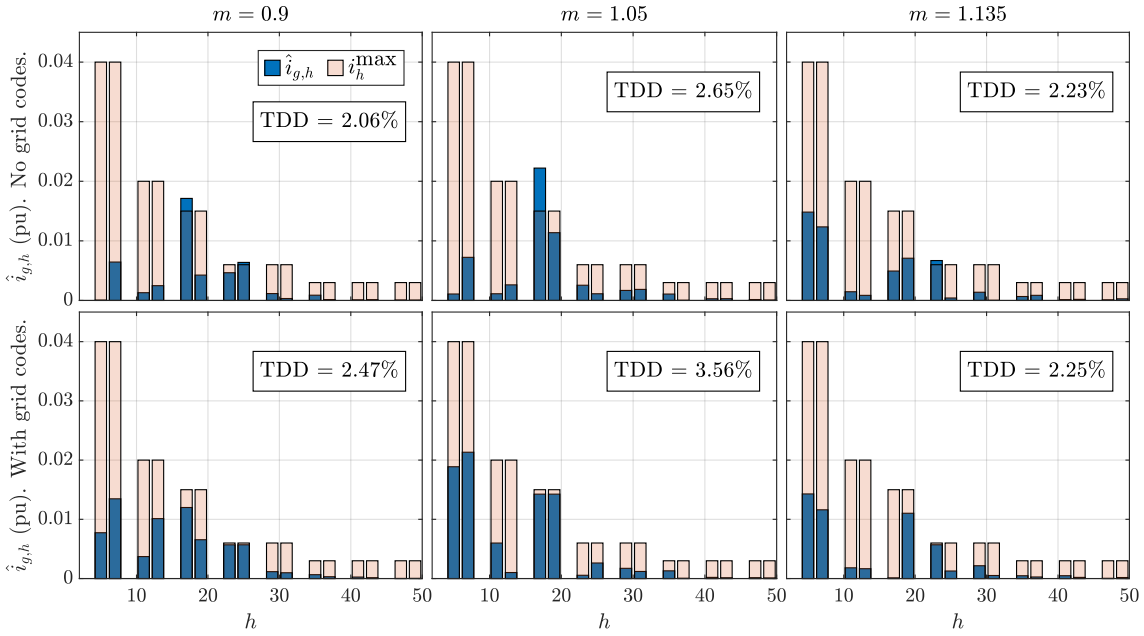


Figure 5.2: Fourier spectra of grid currents. The first row plots result from model (4.7). The second row from (4.9).

applied considering the 95% of their amplitudes, so the problem was even more restricted. The SAs for this case are shown in Fig. 5.1 in the range $m \in [0.85, 1.15]$, so the differences can be clearly noticed. The dashed lines are the SAs with no grid code constraints, whereas the solid lines are the SAs meeting the grid codes.

To test the effectiveness of the proposed model(4.9), it was compared with (4.7) by simulating three cases of modulation indices, which are $m \in \{0.9, 1.05, 1.135\}$, for a switching frequency of 250 Hz ($d = 5$). The Fourier spectra of the resulting grid currents and the grid currents' TDD are shown in the arrangement of plots in Fig. 5.2, where each column corresponds to one of the modulation indices evaluated. The plots in the first row result from the operation using OPPs derived from model (4.7), that do not incorporate grid code constraints. As can be noticed, in the three selected cases, at least one of the harmonic components does not comply with the maximum amplitude allowed by the norm, even if they meet the TDD limit. On the other hand, when incorporating the grid code constraints through model (4.9), all grid code requirements are met in all the cases, as can be noticed in all plots in the second row. By adding the constraints, the spectra of the currents are modified by enlarging some components that are still far from their maximum amplitude allowed and reducing those that are over $i_h^{\max}$. Even though this comes with a rise in the grid current's TDD, the new values are still below the grid code limit, so it is not an issue. From these results, the effectiveness of adding grid code constraints by incorporating the model of the LCL filter into the OPP optimization problem is validated.

5.2.3 Computation time

One concern that may arise when adding the grid code constraints to the OPP problem is the increase in the SAs computation time. To test this, the SAs were calculated for $d \in \{5, 6, 7, 8, 9\}$ using the software AMPL and the solver Artelys Knitro 13.2.0 in a computer with an 11$^{\text{th}}$ Generation Intel(R) Core(TM) i7–11700 @ 2.50 GHz processor, 32 GB of RAM and a Samsung 860 EVO solid–state drive of 2 TB. The total elapsed time, in seconds, for each case, when solving both models (4.7) and (4.9), are summarized in Table 5.4.

Table 5.4: Computation time comparison between solving problems (4.7) and (4.9).

	$d = 5$	$d = 6$	$d = 7$	$d = 8$	$d = 9$
Elapsed time for (4.7) (s)	17.6	19.2	22.5	26.6	32.3
Elapsed time for (4.9) (s)	152.9	142.4	149.7	137.5	144.7

From Table 5.4 it can be noticed how the computation time is considerably increased when comparing the results from solving (4.7) with the ones from (4.9). However, when increasing the number of switching transitions per quarter-wave, the computation times do not vary considerably, and, in some cases, it is even reduced. In that sense, incorporating these constraints is perfectly scalable regarding d. On the other hand, even though the computation times can be eight times greater than the times for OPPs from the model with no grid code constraints, the elapsed times are still small considering that the OPPs are calculated offline and that meeting grid codes in real–time operation is mandatory.

5.3 Multistep finite–control–set MPC

Regarding long–horizon MFCS–MPC, it is necessary to extend the prediction horizon and enhance the efficiency of the SDA. In this line, this work proposes a generalized variable–step–size long prediction horizon formulation to extend the prediction horizon over time with no additional real–time computational burden. Regarding the efficiency of the SDA, a general exploration pattern of the search tree was introduced, which can be understood as a rearrangement of the nodes at each tree level, reducing the number of nodes explored before the solution is found.

To evaluate the proposed improvements, different variable–step–size prediction horizons were tested to visualize the enhancement in the trade–off between switching frequency and $I_{g,\mathrm{TDD}}$ for long horizon MFCS control. Then, the traditional exploration pattern and the one proposed were compared regarding nodes required to find the solution.

5.3.1 Problem formulation

The formulation of the MFCS problem in all these cases is the following:

$$\min_{\boldsymbol{U}(k)} \sum_{\ell=k+1}^{k+N} \|\boldsymbol{y}^{\star}(\ell) - \boldsymbol{y}(\ell)\|^2_{\boldsymbol{\lambda}^y_\ell} + \sum_{\ell=k}^{k+N-1} \lambda^u \|\boldsymbol{u}_{abc}(\ell) - \boldsymbol{u}_{abc}(\ell-1)\|_2^2$$

s.t.

$$\boldsymbol{x}(\ell+1) = \boldsymbol{A}_\ell \boldsymbol{x}(\ell) + \boldsymbol{B}_\ell \boldsymbol{P} \boldsymbol{u}_{abc}(\ell) \qquad \forall \ell \in \{k, \dots, k+N-1\} \tag{5.1a}$$

$$\boldsymbol{y}(\ell) = \boldsymbol{C}\boldsymbol{x}(\ell) \qquad \forall \ell \in \{k+1, \dots, k+N\} \tag{5.1b}$$

$$\boldsymbol{x}(k) = \boldsymbol{x}_0 \tag{5.1c}$$

$$\|\Delta \boldsymbol{u}_{\boldsymbol{abc}}(\ell)\|_\infty \leq 1\,, \quad \forall \ell \in \{k, \dots, k+N-1\} \tag{5.1d}$$

$$\boldsymbol{U}(k) \in \mathbb{U}, \tag{5.1e}$$

where all variables are in per unit.

The objectives are the tracking of ideal references and the penalization of the switching effort. Following the same procedure as in chapters 2 and 4, this formulation can be written as a truncated ILS problem and be solved using the SDA.

Regarding the notation, the apostrophe was dropped when denoting the sampling instants $\{k', k'+1, \dots\}$ for ease of notation. However, the generalized variable–step–size prediction horizon is used in all the following analyses, and the set $\mathcal{S}$ is specified in each case. So, from now on, the apostrophe will be dropped unless explicitly declared.

5.3.2 Selection of MFCS variable–step–size prediction horizon and weighting factors

The generalized variable–step–size prediction horizon adds another degree of freedom to adjust the MFCS controller. Hence, it is relevant to analyze the effect of all degrees of freedom together, i.e., the prediction horizon and the weighting factors. First, the tracking of each output reference is weighted by an element of $\boldsymbol{\lambda}^y_\ell$ at each instant ℓ. Because evaluating different combinations of weighting factors is too complex, a simple and effective solution is that tracking all references is weighted equally at each step. Therefore, the only weighting factor to be adjusted is λ^u, which is kept constant along the prediction horizon.

In the traditional approach with a regularly–sampled prediction horizon, the weighting factor related to the tracking of a reference is kept constant for the whole prediction horizon. This can be

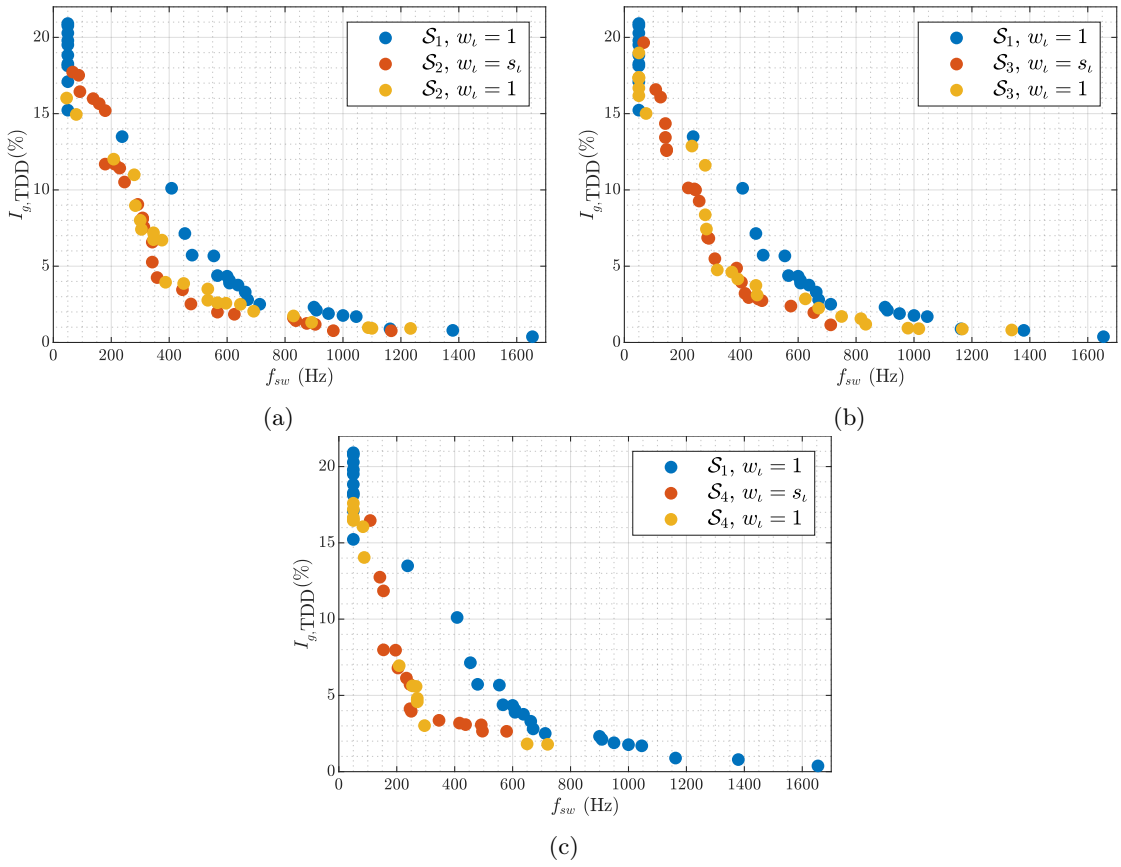


Figure 5.3: Comparison of Pareto fronts between the traditional prediction horizon $\mathcal{S}_1$ and the variable–step–size one proposed. The prediction horizons are (a) $\mathcal{S}_2$, (b) $\mathcal{S}_3$, and (c) $\mathcal{S}_4$.

understood as this objective being equally relevant at each step, i.e., no steps are more important than others. However, as explained in Chapter 4, for the variable–step–size prediction horizon, a constant weighting factor weights less the steps with larger scaling factors, which means that the steps will have different relevance according to their scaling factor.

Two approaches will be analyzed. On one hand, the weighting factor is kept constant along the prediction horizon, so the relevance of the objective at each step will be reduced if the scaling factor is large. On the other hand, the weighting factor is scaled at each step by the corresponding scaling factor, making the objective equally relevant at each step. So, let the matrix $\boldsymbol{\lambda}_\ell^y$ be defined as

$$\boldsymbol{\lambda}_\ell^y = w_\iota \cdot \mathrm{diag}\left\{\lambda_\ell^i,\, \lambda_\ell^i,\, \lambda_\ell^g,\, \lambda_\ell^g,\, \lambda_\ell^c,\, \lambda_\ell^c\right\}, \quad \ell = k + \iota,\, \forall \iota \in \{1,\, 2,\, \ldots,\, N\}. \tag{5.2}$$

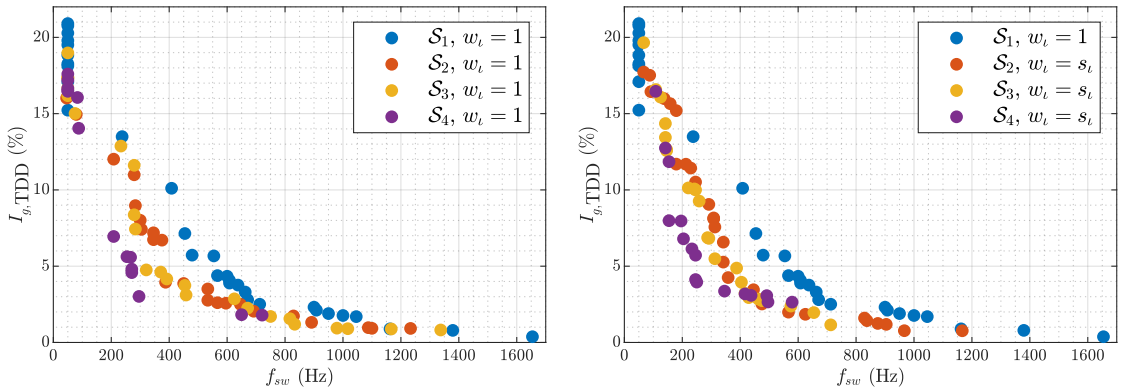
By setting $w_\iota = 1$, the relevance of tracking the references varies along the prediction horizon, whereas it is kept constant for $w_\iota = s_\iota$. Recall that, as stated before, the weighting factors were set as $\lambda_\ell^i = \lambda_\ell^g = \lambda_\ell^c = 1$ for this section, so

$$\boldsymbol{\lambda}_\ell^y = w_\iota \boldsymbol{I}_6, \quad \ell = k + \iota,\, \forall \iota \in \{1,\, 2,\, \ldots,\, N\}. \tag{5.3}$$

Now that the adjustment of the weighting factors is clear, all these degrees of freedom are evaluated for steady–state and transient operation, together with a Fourier spectrum analysis.

Steady–state operation

On steady–state operation, the trade–off between switching frequency of the semiconductors (f_{sw}) and $I_{g,\mathrm{TDD}}$ was assessed. To this end, the prediction horizons tested were $\mathcal{S}_1 = \{1,\, 1,\, 1,\, 1,\, 1\}$, $\mathcal{S}_2 = \{1,\, 2,\, 2,\, 2,\, 2\}$, $\mathcal{S}_3 = \{1,\, 2,\, 2,\, 4,\, 4\}$, and $\mathcal{S}_4 = \{1,\, 4,\, 4,\, 4,\, 4\}$. For all prediction horizons, the cases $w_\iota = 1$ and $w_\iota = s_\iota$ were evaluated (excepting for the traditional horizon $\mathcal{S}_1$, given that $s_\iota = 1,\, \forall \iota \in \{1,\, 2,\, \ldots,\, N\}$). The prediction horizon considered was $N = 5$. The experiment consisted in setting $p^\star = 1$ and $q^\star = 0$ and analyzing the steady–state operation of the system for



(a) (b)

Figure 5.4: Comparison of Pareto fronts between (a) $w_\iota = 1$ and (b) $w_\iota = s_\iota$.

different values of λ^u, simulating 99 cases in the interval $[10^{-3}, 10^0[$. The number of nodes explored by the SDA was not restricted, so the control action applied was always the optimal. Notice that an increasing scaling factor along the prediction horizon implies that, for $w_\iota = 1$, the tracking of references is more relevant in the first steps, losing weight as the steps go further in the horizon.

The results are shown in Fig. 5.3 and Fig. 5.4. In Fig. 5.3, the Pareto fronts for each set $\mathcal{S}$ are compared with the one of traditional prediction horizon $\mathcal{S}_1$. It can be noticed that all the cases working with the proposed variable–step–size prediction horizon result in a better trade–off between $I_{g,\mathrm{TDD}}$ and f_{sw}, as the proposed formulation allows extending the horizon in time. As the prediction horizon is extended further in time, the Pareto front gets closer to the origin and takes more distance from the traditional approach. Notice how the configurations $w_\iota = s_\iota$ and $w_\iota = 1$ render a similar performance, even though the former one is slightly better at low switching frequencies. In any case, both approaches perform better than the traditional one.

On the other hand, in Fig. 5.4, the different prediction horizons are compared based on the setting of the weighting factors through time, i.e., $w_\iota = 1$ or $w_\iota = s_\iota$. Both plots clearly show how the Pareto fronts get closer to the origin as the prediction horizon is extended in time, and how the operation at low switching frequencies is improved. In general, it is clear that even for $\mathcal{S}_2$, which is the shorter (in time) variable–step–size prediction horizon analyzed, the performance over the traditional approach is considerably better, achieving similar values of $I_{g,\mathrm{TDD}}$ with switching frequencies 200 Hz smaller in the range of 200–700 Hz, for the case $w_\iota = s_\iota$.

Transient operation

To test the speed of the dynamic response for different prediction horizons, the system operation is initialized with $p^\star = 1$ and $q^\star = 0$, and simultaneous steps on active and reactive power references are applied so the new references are $p^\star = -0.8$ and $q^\star = -0.8$. The same prediction horizons are evaluated. Each weighting factor is set so the operation point is the closest one to $f_{sw} = 600$ Hz in the previously determined Pareto front. The results for each prediction horizon with both configurations are shown in Fig. 5.5. It can be clearly noticed how all variable–step–size prediction horizon cases perform better than the traditional approach. Regarding the weighting factor selection, the case $w_\iota = s_\iota$, in general, performs better than the case $w_\iota = 1$, as the overshoot tends to be reduced and the oscillations after reaching the new reference are smaller.

On the other hand, Fig. 5.6 allows comparing the prediction horizons when $w_\iota = s_\iota$, which showed to be the selection of weighting factors that performs better. From Fig. 5.6 it is easy to notice how the transient response is improved as the variable–step–size prediction horizon extends in time. Overshoots and undershoots are reduced, the new references are reached faster, and the oscillations after getting to the new references are smaller as the prediction horizon is extended, proving that the proposed strategy not only enhances the steady–state performance of MFCS controllers but also makes the dynamic response faster. Notice that the active and reactive power references are properly tracked in every case.

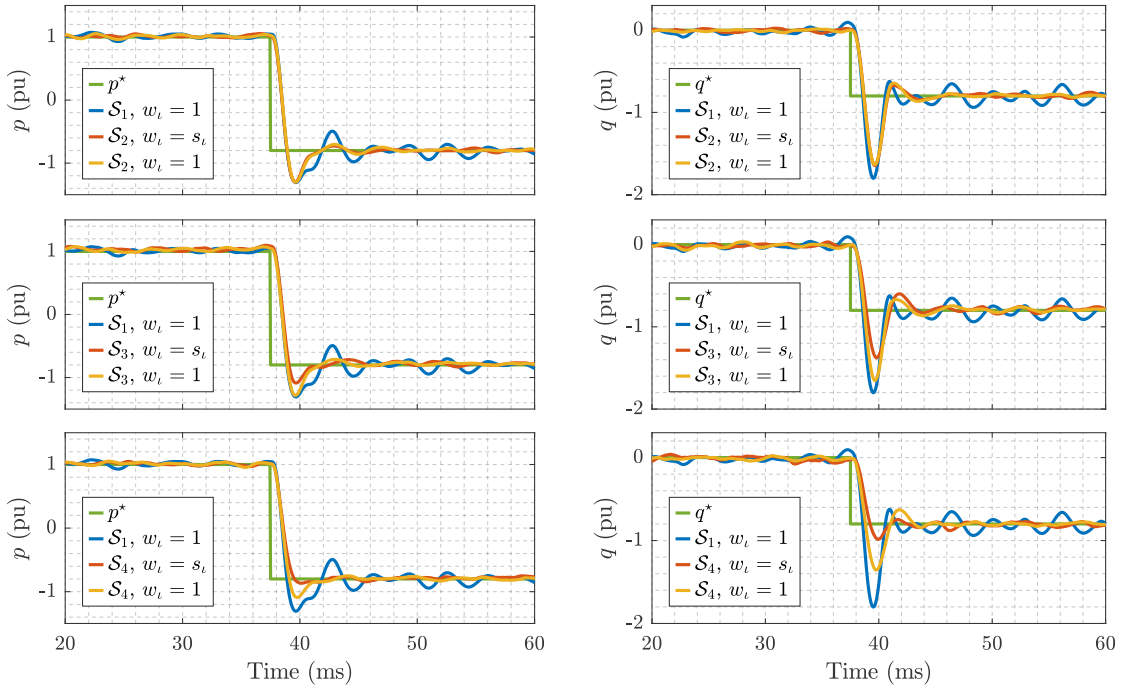


Figure 5.5: Dynamic behavior of instant active and reactive power after steps in their references. (a) Step in active power reference. (b) Step in reactive power reference.

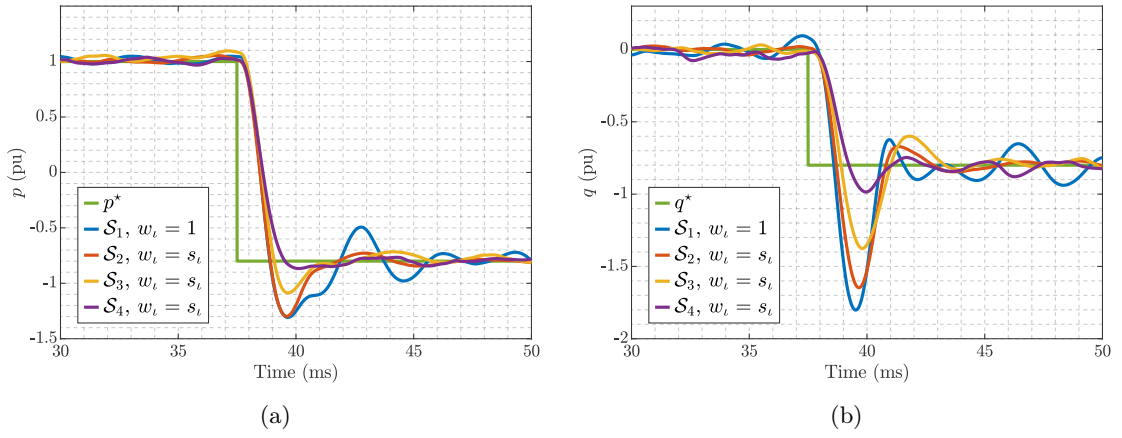


Figure 5.6: Comparison of dynamic behavior between prediction horizon in case $w_\iota = s_\iota$. (a) Step in active power reference. (b) Step in reactive power reference.

Fourier spectrum and grid codes

Regarding grid codes, the proposed variable–step–size horizon was tested in the case $p^\star = 1$ and $q^\star = 0$. The horizons $\mathcal{S}_1$ and $\mathcal{S}_4$ with $w_\iota = s_\iota$ were tested on steady–steady state operation. The Fourier spectrum of one grid current was calculated for every horizon, as shown in Fig. 5.7, where the fundamental component was removed to obtain a clearer graphic. It can be noticed that both spectra are continuous, meaning that many interharmonics are present. This is a well–known issue of MFCS–MPC. Because the applied voltages are neither periodic nor symmetric between the three phases, interharmonics and triplen harmonics appear. It can also be noticed how the amplitude of low order harmonics is considerably reduced when comparing the proposed horizon $\mathcal{S}_4$ with the traditional one $\mathcal{S}_1$, showing the benefit of the proposed approach.

On the other hand, Fig. 5.8 shows the Fourier spectra without all interharmonics, so only components considered in Table 2 of the IEEE Std. 519–2022 [1] are shown, together with the respective limits imposed. The TDD of the grid current analyzed is 3.61% for horizon $\mathcal{S}_1$ and 1.96%

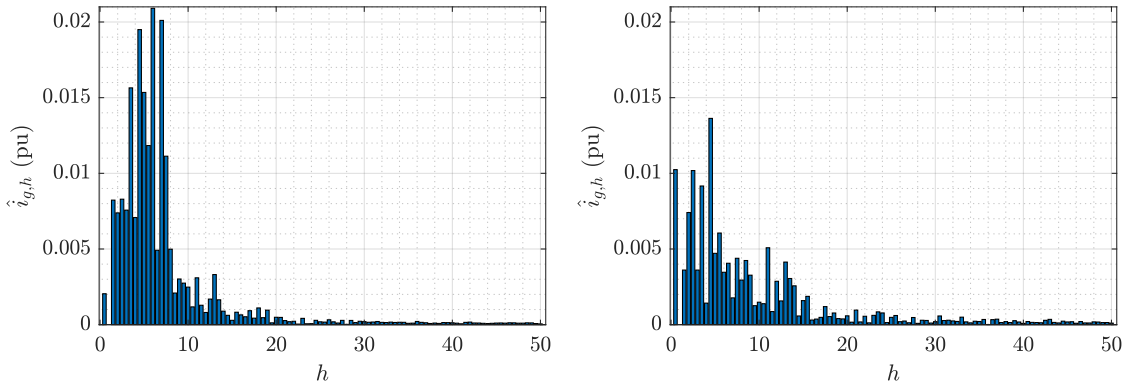


Figure 5.7: Comparison of Fourier spectra (25 Hz resolution) of grid current $i_{g,a}$ between horizons (a) $\mathcal{S}_1$ and (b) $\mathcal{S}_4$ with $w_\iota = s_\iota$.

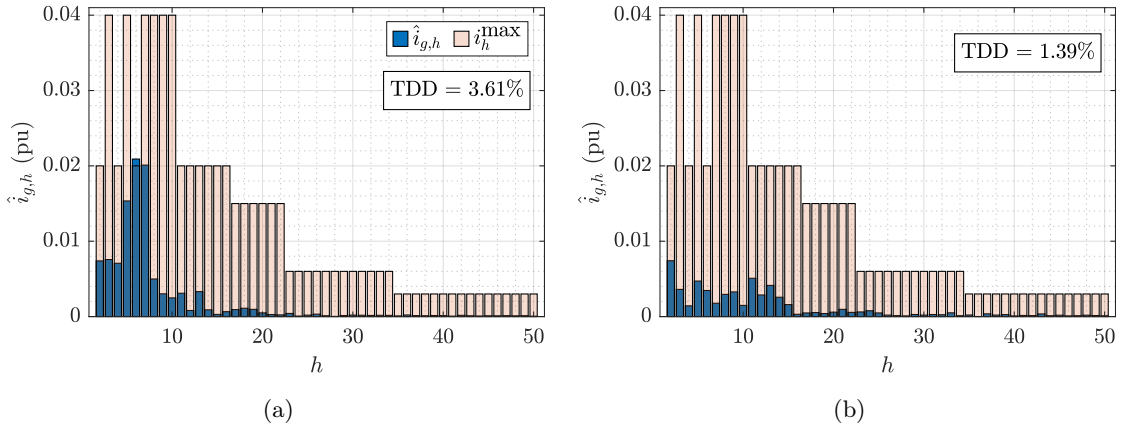


Figure 5.8: Grid code limits and Fourier spectra of grid current $i_{g,a}$ for horizons (a) $\mathcal{S}_1$ and (b) $\mathcal{S}_4$ with $w_\iota = s_\iota$.

for horizon $\mathcal{S}_4$. As can be noticed, horizon $\mathcal{S}_1$ does not meet the grid code as the 6^{th} harmonic component is over the allowed limit, even though it meets the current TDD limit. In contrast, the variable–step–size horizon $\mathcal{S}_4$ meets both harmonics amplitudes and TDD limit. However, this standard also specifies that limits for interharmonics should be evaluated case–by–case, meaning that, depending on the specific application, these results may not meet the grid codes. Even though the variable–step–size prediction horizon improves both steady–state and transient performance, MFCS controllers cannot be generalized as a suitable strategy for grid–connected applications as it is impossible to predict if grid–codes will be met because of the continuous and unpredictable Fourier spectra of the resulting control actions.

5.3.3 SDA search–tree exploration pattern analysis

Regarding the efficiency of the SDA used to solve the optimization problem of MFCS–MPC, the novel exploration pattern developed to reduce the nodes explored before the solution is found was tested. 5601 cases were generated from the operation of the proposed strategy with $\mathcal{S} = \{1, 4, 4, 4, 4\}$, $w_\iota = s_\iota$, and the same weighting factor as in the transient analysis. The system was simulated during seven fundamental cycles, where the problem formulated at each control period, every 25 μs, was stored, resulting in the 5601 cases mentioned.

The two metrics analyzed are the nodes the algorithm requires to find the solution and the total nodes explored until the algorithm finishes. Recall that the algorithm finishes when no nodes are left to explore or when the node limit is reached. Therefore, in general, the algorithm keeps exploring nodes after the solution is found. For this analysis, the algorithm never stopped by node limit, so the problem was always solved to optimality. The histograms showing the probability

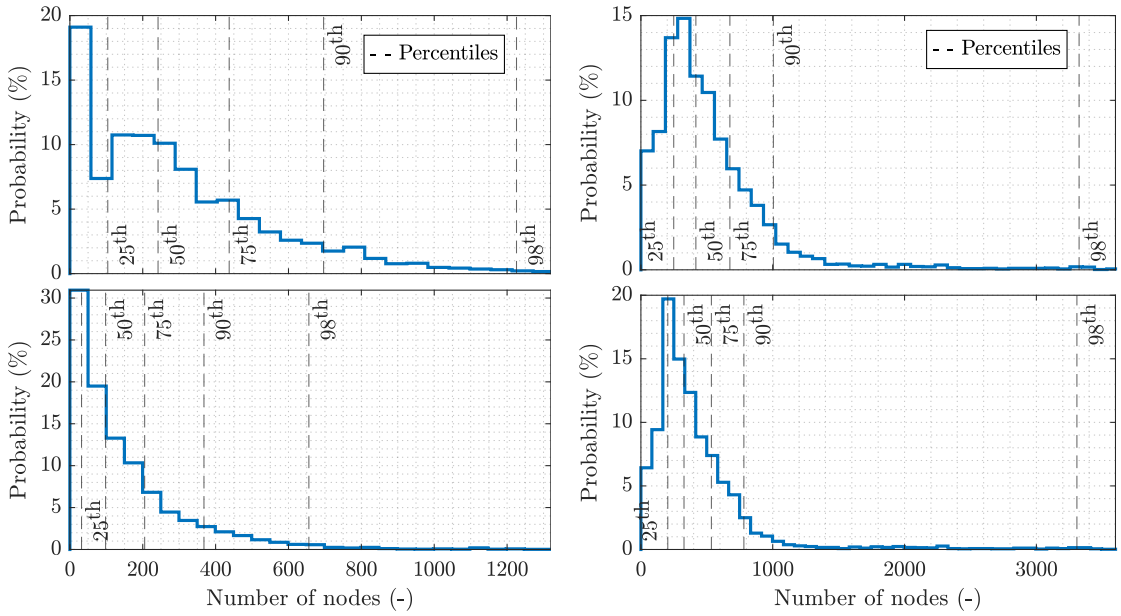


Figure 5.9: Histograms of nodes explored. Upper plots are from the traditional exploration pattern. Lower plots are from the proposed approach. (a) Nodes required to find the solution. (b) Total nodes explored until the SDA finishes.

density function (PDF), in percentage, are shown in Fig. 5.9, where the upper plots result from the traditional exploration pattern, whereas the lower ones are the results from the proposed approach. As can be noticed from Fig. 5.9b, with the proposed approach, the number of nodes explored before finding the solution tends to be smaller, and its PDF presents a shorter tail than the traditional approach, with all marked percentiles located in smaller numbers of nodes explored. In 67.5% of the cases analyzed, the proposed approach found the solution exploring a smaller number of nodes, and in 85% of the cases, it required fewer or the same amount. These results prove that the proposed exploration pattern allows reducing the limit of nodes explored `Max_Nodes N`, as the solution is found quicker, reducing the computational burden of the SDA.

Something similar to the nodes required to find the solution happens to the total nodes explored until the SDA finishes. All marked percentiles are smaller in the proposed approach, and its tail is shorter, concentrating most cases on the lower side of the distribution.

One key benefit of the proposed exploration pattern is the possibility of starting the SDA from level $j = n_u \cdot N$ in the search tree, as the first path is the initial incumbent. In all cases where the nodes required to find the solution is non–zero, this initialization reduces in $n_u \cdot N$ the number of nodes explored, whereas the total nodes explored by the SDA is always reduced. In this case, $n_u \cdot N = 15$, leading the proposed exploration pattern to perform better than the traditional pattern in 69.4% of the cases and better or equal in 87%, improving the performance even more. Even though this may seem like a small improvement, reducing in $n_u \cdot N$ the nodes required to find the solution is a considerable improvement when stopping the SDA based on the explored nodes limit because of the computational burden, as will be shown later.

5.4 Proposed control strategy: OPP–MFCS

Operating the system using OPPs as a modulator would allow meeting grid codes at a low and fixed switching frequency, but it would lead to a slow dynamic response. On the other hand, MFCS controllers entail a fast dynamic response and a low $I_{g,\mathrm{TDD}}$, but, as shown in the previous section, the resulting Fourier spectra of the grid currents is continuous, and therefore MFCS–MPC cannot be generalized as a strategy suitable for grid–connected converters. Based on two optimization stages, the proposed control framework was used to develop a controller (OPP–MFCS) that leverages the benefits of both OPP and MFCS controllers. The proposed strategy must meet the grid codes on steady–state while operating at a low and fixed switching frequency

of the semiconductors, with periodic and symmetrical voltages between the three phases, which would traduce in a discrete Fourier spectrum. Also, the dynamic response of the strategy must be fast. Additionally, as OPPs are being combined with MFCS control, the SDA can be initialized with $\tilde{\boldsymbol{U}}(k)$ as the initial candidate, which results in requiring a smaller number of nodes explored to find the solution. To evaluate the benefits of the proposed OPP–MFCS controller, its degrees of freedom must be adjusted, which will be the first step. Then, an evaluation of the initial candidate's impact on the number of nodes explored before finding the solution is done. Finally, the capacity of OPP–MFCS to meet the grid codes is tested by analyzing the steady–state operation of the system. The control formulation used in this section is detailed in Chapter 4.

5.4.1 Selection of MFCS variable–step–size prediction horizon and weighting factors

As in Section 5.3, the first task is to adjust the weighting factors along the prediction horizon. Regarding $J_y(\boldsymbol{Y}(k))$, as done for the MFCS–MPC analysis, the weighting factors are set as $\lambda_\ell^i = \lambda_\ell^g = \lambda_\ell^c = 1$, so, for the variable–step–size prediction horizon, the weighting factor matrix related to tracking the outputs' references is defined, at each step in the horizon, as

$$\boldsymbol{\lambda}_\ell^y = w_\iota \boldsymbol{I}_6, \quad \ell = k + \iota, \forall \iota \in \{1, 2, \ldots, N\}. \tag{5.4}$$

In the proposed formulation (4.15), the same thing has to be done regarding the tracking of $\tilde{\boldsymbol{U}}(k)$, which is the objective defined by $J_u(\boldsymbol{U}(k))$. To this end, it is proposed to set the weighting factors along the prediction horizon as

$$\boldsymbol{\lambda}_\ell^u = \lambda^u w_\iota \boldsymbol{I}_6, \quad \ell = k + \iota, \forall \iota \in \{1, 2, \ldots, N\}, \tag{5.5}$$

where λ^u is a parameter that must be tuned. The tuning process is simpler than in the traditional MFCS–MPC problem, as calculating the Pareto front is not required; this is because the tracking references $\tilde{\boldsymbol{U}}(k)$ result from sampling, at each instant k, the OPP used to generate $\tilde{\boldsymbol{Y}}(k)$, and it is sampled according to the variable–step–size prediction horizon used. Because $\tilde{\boldsymbol{U}}(k)$ is a sampled version of the OPP, the control action can perfectly track it, as it considers the discrete nature of the converter's output voltage and the discrete nature of the instants at which the actuation can change (recall that MFCS controllers do not use a modulator. Therefore, the actuation can only change at the end of each control period). Hence, by adjusting λ^u in a large enough value, the output of the controller will perfectly track $\tilde{\boldsymbol{U}}(k)$ in steady–state operation, given that applying the first element of $\tilde{\boldsymbol{U}}(k)$ would result in system outputs $\boldsymbol{y}$ that are consistent with the tracking references $\tilde{\boldsymbol{y}}(k)$, which would imply a small tracking error of the outputs.

Recall that OPPs are steady–state voltages, so directly applying them would result in a slow dynamic response. Hence, by adjusting λ^u, the speed of the closed loop is also adjusted, where, in general, a larger value of λ^u would result in a slower dynamic response. In contrast, a smaller λ^u would allow more aggressive control actions, where the aggressiveness is understood as the switching effort, i.e., a more significant number of additional switching transitions would occur during transient responses.

As a large λ^u guarantees that $\tilde{\boldsymbol{U}}(k)$ is perfectly tracked in steady–state operation, the tuning process of λ^u is as simple as, for a given prediction horizon, setting a small enough λ^u that entails a fast dynamic response while guaranteeing that $\tilde{\boldsymbol{U}}(k)$ is perfectly tracked on steady–state.

The same step response as in Section 5.3 was tested, and for the same cases, excluding the prediction horizon $\mathcal{S}_2$, which was not considered as longer horizons rendered a better performance in the MFCS controller analysis. The parameter λ^u was adjusted so the step response was fast, while the voltages would entail a fixed switching frequency of the semiconductors on steady–state. The system's steady– state behavior is analyzed in detail in a further section. The results from these simulations are shown in Fig. 5.10. As in the analysis of the MFCS controller, with the OPP–MFCS strategy, longer horizons in time resulted in faster dynamic response and a better capability of damping the oscillations introduced by the filter. The overshoots in the dynamic response are clearly reduced, being $\mathcal{S}_4$ the variable–step–size horizon that results in the fastest response with the smallest overshoots, in particular for $w_\iota = s_\iota$, which in all cases performed better than setting $w_\iota = 1$.

The switching patterns $\boldsymbol{u}_{abc}$ applied by the converter, for the horizon $\mathcal{S}_4$ with $w_\iota = s_\iota$ are shown in Fig. 5.11. It can be noticed that the system starts operating in steady–state with symmetric

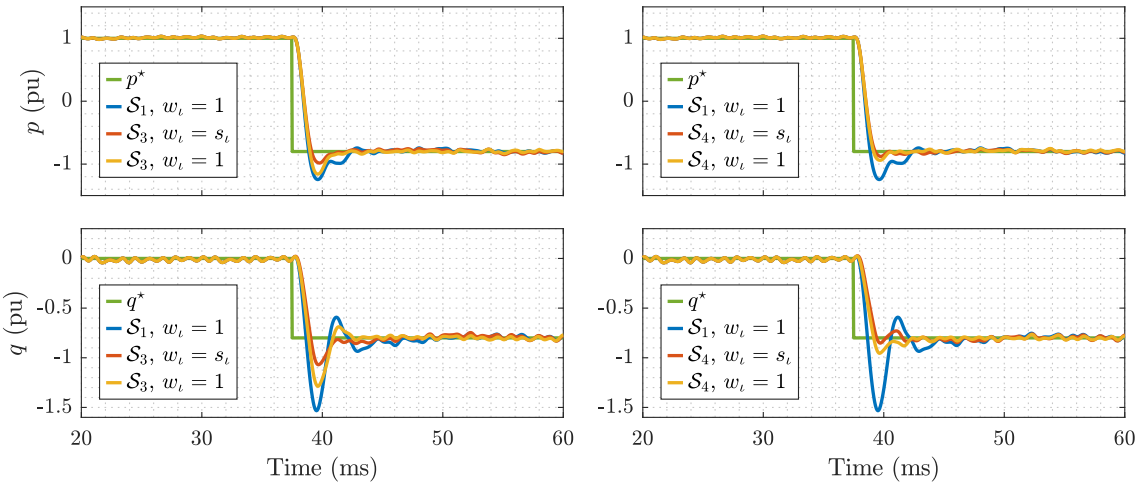


Figure 5.10: OPP–MFCS response to steps in both active and reactive power references.

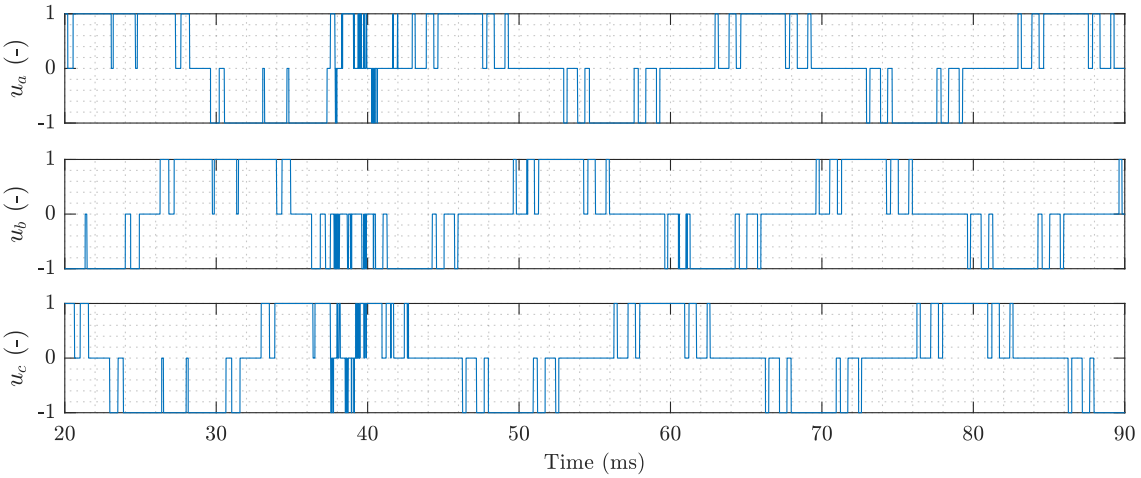


Figure 5.11: Switching patterns resulting from OPP–MFCS in step response, for $\mathcal{S}_4$ with $w_\iota = s_\iota$.

voltages between the three–phases and at a fixed and low switching frequency. Right after the step in active and reactive power references, additional switching transitions occur, which allow the controller to render a faster dynamic response. However, once the system reaches the references, the switching frequency returns to the value defined by the OPPs, and the voltages between the three phases become symmetric again. Naturally, if λ^u is set on a larger value, the steady–state pattern would be reached faster. However, the tracking of the references would be slower, and the oscillations induced by the filter would not be properly damped. It is important to highlight that additional switching transitions are needed to render a fast dynamic response. Nevertheless, an excessive number can damage the converter. To deal with this issue, a term penalizing the switching effort, as the one in [41], can be added to the cost function with its own weighting factor. This analysis is left as future research.

5.4.2 SDA initial solution

The formulation proposed for OPP–MFCS has another benefit in terms of SDA efficiency. As $\tilde{\boldsymbol{U}}(k)$ can be perfectly tracked by the control action, it can also be used as the initial incumbent for the SDA if it meets constraint (4.15d). Recall that, even though OPPs are calculated considering (4.15d), during transient operation, getting to $\tilde{\boldsymbol{u}}(\ell = k)$ (the first element in $\tilde{\boldsymbol{U}}(k)$) from the previous actuation may require transitions of size larger than one. However, when $\tilde{\boldsymbol{u}}(\ell = k)$ is feasible and can be used as the initial incumbent, the term $J_u(\boldsymbol{U}(k))$ in the cost function becomes zero, which may result in a considerable reduction in the initial radius of the sphere. Additionally, in steady–state, $\tilde{\boldsymbol{U}}(k)$ is the solution of the MFCS problem, therefore initializing the algorithm

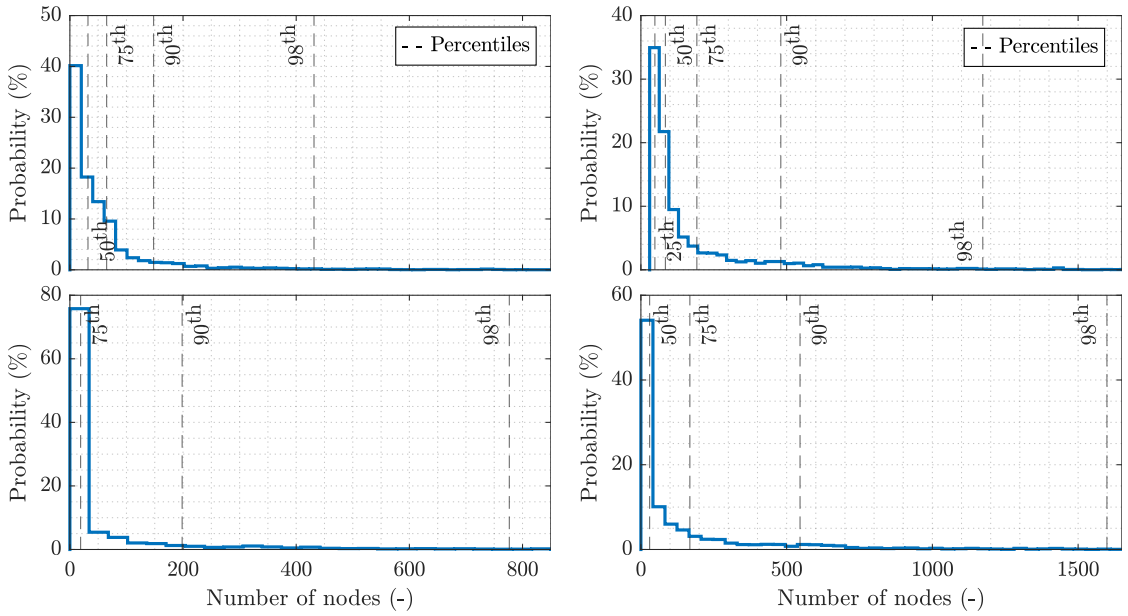


Figure 5.12: Histograms of nodes explored. Upper plots result from using the educated guess as initial candidate. Lower plots result from the proposed initialization strategy. (a) Nodes required to find the solution. (b) Total nodes explored until the SDA finishes.

with the actual solution.

The proposed initialization method is straightforward. Every time $\tilde{\boldsymbol{U}}(k)$ is feasible, the SDA is initialized with it. When not, the traditional educated guess is used. It must be highlighted that this procedure is not the same as evaluating the cost function for both options and choosing the one rendering the lower cost, as doing that would require evaluating the cost function of a second candidate, which would result in an additional computational burden, equivalent to evaluating a whole full–depth path in the search–tree.

To test the proposed initialization strategy, 5601 cases were generated and solved using the traditional educated guess initialization and the proposed strategy. The proposed exploration pattern and starting point of the search tree were used, and the prediction horizon was set as $\mathcal{S} = \{1, 4, 4, 4, 4\}$ with $w_\iota = s_\iota$. The histograms of the PDF, in percentage, are shown in Fig. 5.12. For the proposed initialization, percentiles 1 to 74 of the nodes required to find the solution are equal to zero, meaning that the proposed strategy starts the algorithm with the solution. In the case of the traditional educated guess, only percentiles 1 to 27 are equal to zero. The PDF of the nodes required to find the solution of the proposed initialization is mainly concentrated near the origin. However, its tail is longer as the worst case during transient entails more nodes explored. The proposed initialization explored the same or fewer nodes to find the solution in 87.07% of the cases analyzed (in 62.04% were fewer). Regarding the nodes explored before the SDA is finished, the percentiles 1 to 52 of the proposed initialization PDF are equal to 30, the minimum number of nodes explored to finalize the algorithm. For the traditional approach, it happens only in percentiles 1 to 16. In 85.29% of the cases, the proposed initialization explored the same or fewer nodes before terminating the SDA (in 58.79% were fewer). These results show the effectiveness of the proposed initialization.

5.4.3 Meeting grid codes in steady–state

The last test for the OPP–MFCS control strategy is to verify that it is capable of meeting with the IEEE Std. 519–2022 [1] grid currents requirements while operating the converter at a low and fixed switching frequency with symmetric voltages between the three phases. To this end, four cases with different active and reactive power references were run in steady–state, using $d = 5$, i.e., $f_{sw} = 250\,\text{Hz}$, and the proposed OPP formulation considering grid code constraints. The prediction horizon was set as $\mathcal{S} = \{1, 4, 4, 4, 4\}$, and the weighting factor as $w_\iota = s_\iota$ and λ^u as

the one used for the step response analysis.

First, the Fourier spectrum of one grid current is plotted for each case using a granularity of 25 Hz, i.e., the harmonic components are $\mathcal{H} = \{1.5, 2, 2.5, \ldots, 49.5, 50\}$. These results are shown in Fig. 5.13, where it can be noticed that in all cases, the respective Fourier spectrum is discrete, and only differential components that meet with (2.24) have a noticeable amplitude. It can be seen that small interharmonics exist in some cases, which can originate because patterns can never be perfectly symmetric nor periodic because the simulation is not exempt from numerical errors during its execution. However, those interharmonics are always smaller than 0.001 pu.

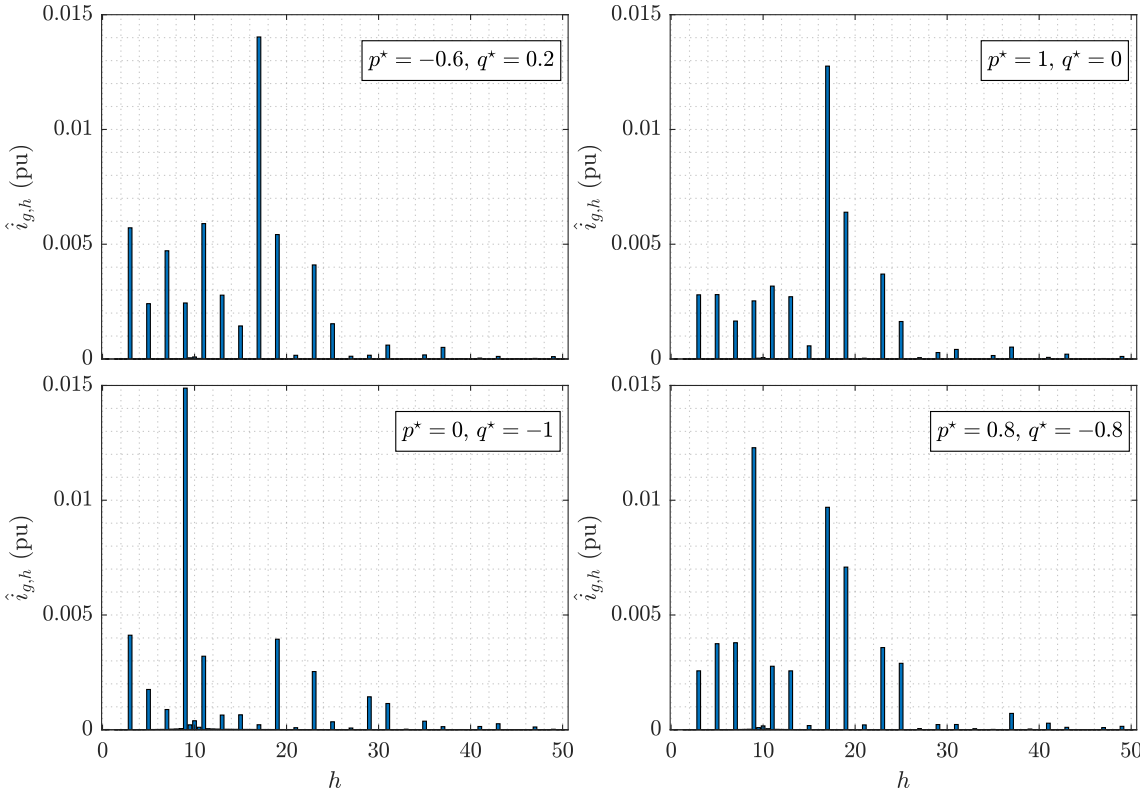


Figure 5.13: Fourier spectrum of $i_{g,a}$ for different operational conditions, for OPP–MFCS.

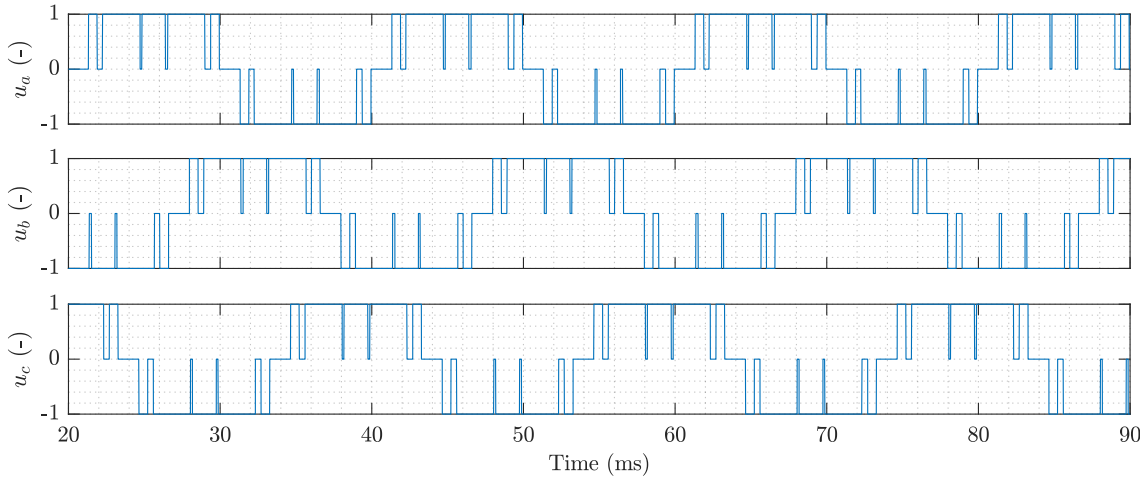


Figure 5.14: Control actions $\boldsymbol{u}_{abc}$ of the case $p^\star = -0.6$, $q^\star = 0.2$.

The discrete spectra obtained by the proposed strategy are a key improvement over traditional MFCS control, and are owed to the periodic and symmetric voltages obtained in steady-state operation. To exemplify, the control actions for the case $p^\star = -0.6$ and $q^\star = 0.2$ are depicted in

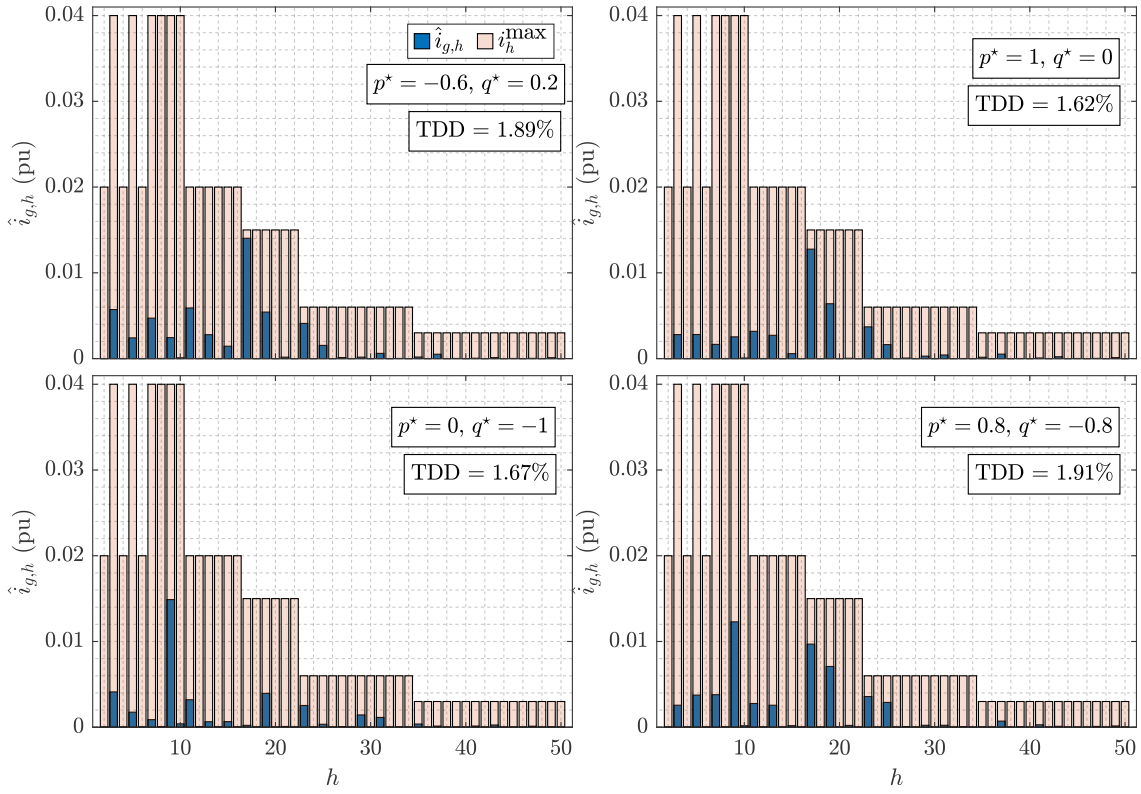


Figure 5.15: Fourier spectrum of $i_{g,a}$, $I_{g,\mathrm{TDD}}$ and IEEE Std. 519–2022 [1] limits, operating with OPP–MFCS.

Fig. 5.14. It can be noticed how the patterns in all phases are periodic, have a fixed switching frequency of 250 Hz, and are symmetric between the three phases. Finally, the TDD and the Fourier spectrum of a grid current are shown together with the grid code limits for the harmonic components amplitudes in Fig. 5.15 for each one of the cases. It can be noticed that, in all cases, each harmonic fulfills the amplitude limit requirement indicated in the IEEE Std. 519–2022 [1], while the TDD of the grid current is smaller than the respective 5% limit indicated in the same standard. With this evaluation, it is proved that the proposed OPP–MFCS strategy is capable of controlling an LCL filter 3L–NPC grid–connected converter while meeting grid codes and achieving symmetrical voltages between the three phases and a fixed and low switching frequency.

Chapter 6

Real–time implementation and tests

A key objective of this work is that the proposed strategy can be executed in real–time by a commercial control platform. In this chapter the whole control strategy is implemented using Xilinx's Vitis HLS, considering a standard commercial FPGA. Then, using C/RTL co–simulation, it is verified that the strategy can be executed in fewer than 25 μs with the resources available in the FPGA. Finally, the whole strategy was tested in a HIL experimental setup using an OPAL–RT OP4510, proving its capacity to meet all the control objectives.

6.1 Computational burden analysis

One of the objectives of this work is for the proposed strategy to be executable in real—time by commercial platforms, which are usually based on some type of processor unit, such as DSPs and FPGAs. The SDA has been widely implemented in FPGAs for real–time applications, being validated as a viable solver for MFCS–based controllers. The next step was to evaluate the viability of the whole strategy to run in real–time deployed in a commercial FPGA. High–level synthesis (HLS) was used for the evaluation. HLS is a process in which the design of an algorithm, programmed in high–level languages (as C or C++), is taken by a tool that is capable of generating the register transfer level (RTL) hardware description. Using HLS tools allows avoiding the complex design process of using a hardware description language (HDL) to implement the algorithm, reducing the prototyping time considerably. Nevertheless, HLS tools, in general, will not generate the most efficient implementation of the design in terms of resources employed and computation time, even though they usually count with some instructions (e.g., of the `#pragma` type), that allow the designer to enhance the implementation in terms of computation time of resources required. In this line, HLS tools permit obtaining a fast estimation of the algorithm's viability if it were to be deployed in a commercial control platform.

In this work, Xilinx's tool, Vitis HLS, was used to evaluate the viability of the proposed OPP–MFCS strategy. From a C or C++design, Vitis HLS is capable of determining the execution time of the design and the resources required, where the resources considered are block RAMs (BRAM), DSP slices (DSP–s), flip–flops (FF), and look–up–tables (CLUT[1], used as the basic element for combinational circuits). When a design has a variable execution time (as the SDA, where the number of nodes explored depends on each case), Vitis HLS has the capability of executing a cycle–accurate simulation, called C/RTL co–simulation, that determines the time (clock cycles) required to execute the algorithm for a set of cases introduced by the designer.

In this section, Vitis HLS is used to analyze the resources and execution time of the proposed strategy when running in a commercial platform. Even though this process will not evaluate the most efficient implementation, an upper bound of the computational burden can clearly be defined.

[1]Here, the acronym CLUT is used to differentiate the FPGA element with the LUTs used to implement some functions in the design.

6.1.1 Setting hyperparameters

The computational burden of the OPP–MFCS real–time implementation directly depends on the values set for the hyperparameters of the different controller's entities. The hyperparameters of this strategy are the number of harmonic components considered by the OBRG–MRS H_r, the number of iterations executed by the FWA $\texttt{N}_{\texttt{iter}}$ until the algorithm finishes, and the limit of nodes explored by each of the SDA that run in parallel `Max_Nodes`. It is important to highlight that the number of operations executed and the elapsed computation time by both OBRG–MRS and FWA can be perfectly determined *a priori* once H_r and $\texttt{N}_{\texttt{iter}}$, respectively, are defined. On the other hand, the computational burden related to each node in the SDA depends on the level at which the node is located, so even if `Max_Nodes` is set, it is still necessary to run cases to determine a worst–case computation time.

The hyperparameters are adjusted so the whole controller can fit in a commercial platform FPGA and run in less than $25\,\mu$s. The first hyperparameter is H_r. Let H_r^i be the number of harmonic components used to generate $\tilde{\boldsymbol{i}}_{\alpha\beta}$, H_r^c the number to generate $\tilde{\boldsymbol{v}}_{c,\alpha\beta}$, and H_r^g the number for $\tilde{\boldsymbol{i}}_{g,\alpha\beta}$. As expected, because of the LCL filter action, the harmonic content of both $\tilde{\boldsymbol{v}}_{c,\alpha\beta}$ and $\tilde{\boldsymbol{i}}_{g,\alpha\beta}$ is considerably lower than the harmonic content of $\tilde{\boldsymbol{i}}_{\alpha\beta}$. This led to the conclusion that H_r^c and H_r^g can be set as zero without deteriorating the strategy's performance, as the control actions in steady–state do not change and the dynamic response is still fast. Therefore, the ideal references $\boldsymbol{v}_{c,\alpha\beta}^{\star}$ and $\boldsymbol{i}_{g,\alpha\beta}^{\star}$ can be directly used in the MFCS problem without having a big impact in the performance, as the ideal references are a great approximation to the tracking references because of the filter action. On the other hand, the converter currents always have the largest harmonic content, so H_r^i cannot be zero, so it was set to $H_r^i = 34$. Regarding the FWA, $\texttt{N}_{\texttt{iter}} = 20$ is enough to guarantee a fast dynamic response. The last hyperparameter to set is `Max_Nodes`. The proposed exploration pattern and the SDA parallelization strategy considerably reduced the number of nodes explored to find the solution. By setting `Max_Nodes` $= 20$, it was noticed that the strategy's performance is not considerably affected, only making the dynamic response slower. It is important to highlight the impact of initializing one of the SDA at level $j = n_u \cdot N$ with the initial incumbent, which was possible because of the proposed exploration pattern. In this case, the nodes explored are reduced in $n_u \cdot N = 15$, which is 75% of `Max_Nodes`. This reduction makes the strategy viable by running in a commercial platform in real–time applications. Additionally, it is important to mention that stopping the SDA before the solution is found implies applying a suboptimal decision. When using the traditional educated guess to define the initial incumbent, which depends on the previous actuation, applying a suboptimal actuation not only affects the actuation, but also deteriorates the initial incumbent at the next control step. In contrast, $\tilde{\boldsymbol{U}}(k)$ is defined at each step independently from the previous actuation, so the initial incumbent is not affected by the suboptimality of the previous decision when using $\tilde{\boldsymbol{U}}(k)$ to start the SDA. Therefore, the proposed strategy to choose the initial incumbent is more robust to the selection of `Max_Nodes` when compared to the traditional initialization strategy.

The performance of the proposed SDA_{ep} when node limits are imposed, considering the exploration pattern, the initialization at the last level, and the parallelization shown in Fig. 4.5, was assessed by generating a set of cases and evaluating the following metric

$$\Delta J = \frac{J_{\text{NL}} - J_{\text{opt}}}{J_{\text{opt}}}, \tag{6.1}$$

where J_{NL} is the value of the cost function obtained when solving the optimization problem with the proposed strategy considering the limit set by `Max_Nodes`, and J_{opt} is its value when solving to optimality. The cases were generated by simulating the behavior of the system operating with the hyperparameters specified in the paragraphs above. The performance

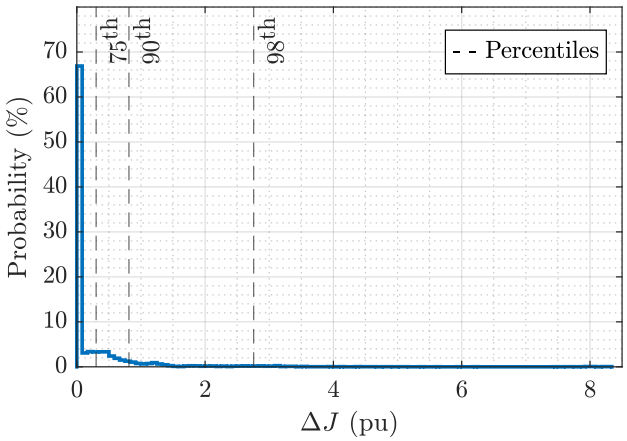


Figure 6.1: Histogram of ΔJ.

of the OPP–MFCS strategy with the hyperparameters specified is shown in Fig. 6.2, where $\mathcal{S} = \{1, 4, 4, 4, 4\}$ and $w_\iota = s_\iota$. Notice that the references are reached fast after the reference step. However, because of the suboptimality produced by `Max_Nodes` during transients, the controller applies the steady–state actuation (the OPPs) before damping the oscillations of the filter, generating power fluctuations around the references as the operation is similar to open–loop operation after reaching the references. It can also be noticed how ΔJ is mainly zero when the control action has reached the references. However, right after a step in the active or reactive power references, the suboptimal incumbents render high cost function values, which keeps happening during the whole transient behavior. However, as the oscillations are reduced, the amplitude of ΔJ gets lower until reaching steady–state and $\tilde{\boldsymbol{U}}(k)$ becomes the solution. The histogram illustrating the ΔJ PDF is shown in Fig. 6.1. It can be noticed that the distribution is mainly located close to the origin, which shows how the suboptimal solution, in most cases, renders a cost function that is close to the minimum. It is important to highlight that this is only a broad approach to the implementation of the OPP–MFCS controller, and by increasing `Max_Nodes` the performance can get highly improved, which can be achieved with a more efficient implementation of the strategy, that can be obtained, for example, by implementing the strategy directly using an HDL. However, developing an efficient implementation is out of the scope of this work, as the objective was only to prove its viability.

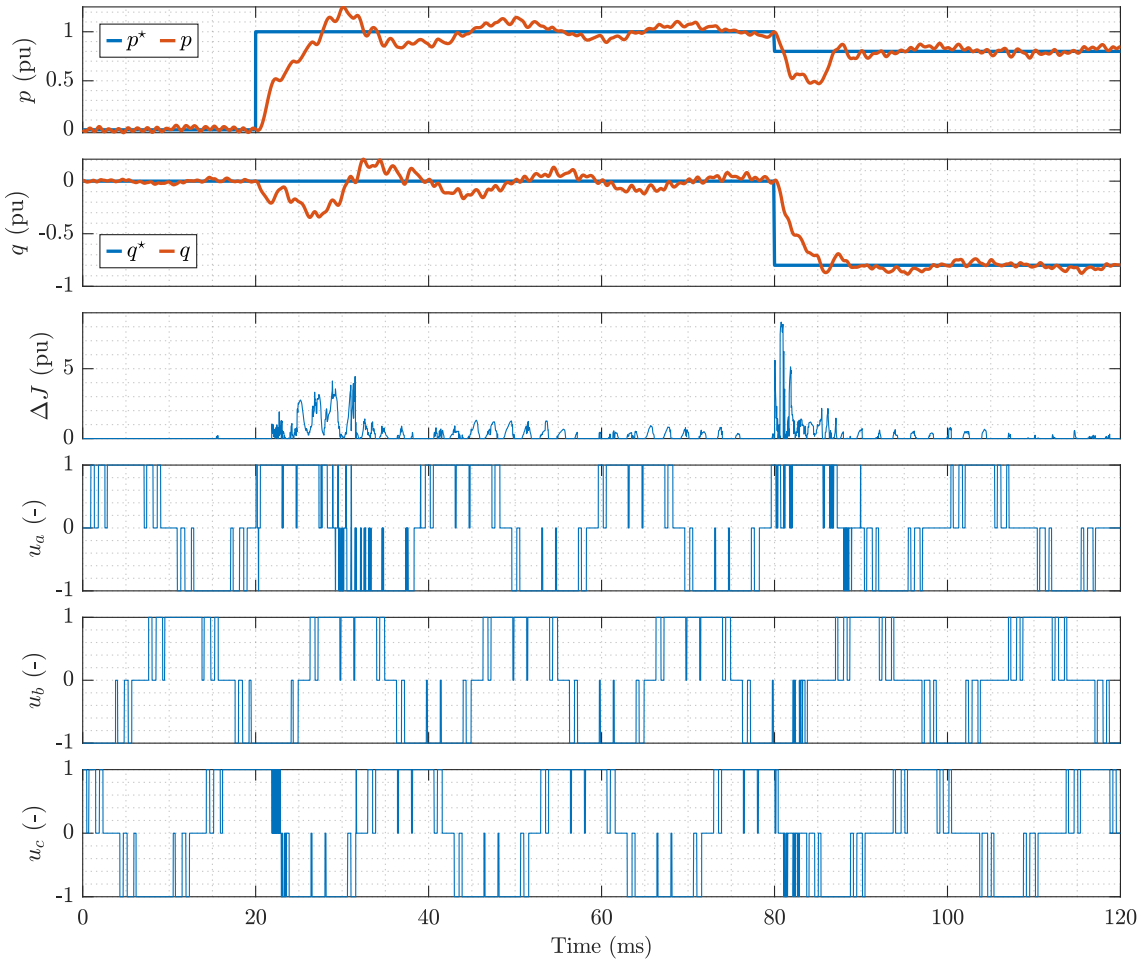


Figure 6.2: p, q, ΔJ and $\boldsymbol{u}_{abc}$ from simulation used as cases generator to set the hyperparameters.

6.1.2 High–level synthesis results

Once the expected controller's performance was verified, the implementation was synthesized using the Vitis HLS tool to corroborate that running it in real–time is viable using a commercial control platform with a 25 μs control period. To this end, a Xilinx XC7Z020 Zynq–7000 FGPA was considered, which is a standard low–cost FPGA, with 220 DSP–s (with a single 25×18 bit multiplier), 53200 CLUTs, 106400 FFs, and 140 BRAM. All the controller was implemented using

a 100 MHz clock and fixed–point arithmetic, so, except for the SDA, 18–bit words were used for all variables that are not integers, as it allows multiplications to be executed by a single DSP–s. At the same time, it is also a standard word length [2]. For the SDA, a clock of 200 MHz was used, and to adjust the word lengths, a testbench with 3900 cases was run, and the lengths were set to get the right solution in all cases. With that configuration, the worst elapsed time with the hyperparameters defined in the previous section was 9.75 μs.

The control strategy was separated into sections. It was assumed that the different sections were all executed sequentially, so the worst case was analyzed. The sections considered are shown in Table 6.1.

Table 6.1: Sections of the OPP–MFCS strategy to generate the FPGA implementation.

Section	Description
Pre–process	Includes the delay compensation stage, the PLL, and the calculation of $m^\star$ and the phasors to determine $\tilde{\boldsymbol{x}}(\theta_1)$.
$\tilde{\boldsymbol{U}}(k)$ generation	Generates the sampled OPP from the LUT storing the SAs.
$\tilde{\boldsymbol{Y}}(k)$ generation	Generates $\tilde{\boldsymbol{Y}}(k)$ using the efficient process described in Chapter 4.
Pre–FWA	Calculates $\boldsymbol{\Theta}$ and $\boldsymbol{U}_{\text{unc}}$, and determines if (4.37) must be solved according to the location of $\boldsymbol{U}_{\text{unc}}$.
$\bar{\boldsymbol{U}}_{\text{unc}}$ generation	Calculates $\bar{\boldsymbol{U}}_{\text{unc}}$.
FWA	Executes the FWA and generates the respective initial candidate $\boldsymbol{U}_{\text{sq}}$ for the SDA.
EP generation	Generates the proposed exploration pattern for the SDA.
$\boldsymbol{U}_{\text{ini}}$ calculation	Determines the initial candidate according to the proposed approach.
SDA	Executes the parallelized SDA. One SDA was generated to be instantiated three times.

All `sine` and `atan2` functions used to generate the tracking references were implemented by LUTs to reduce the computation time and the DSP–s required. All sections were implemented in Vitis HLS, determining their respective usage of resources and computation times. The summary of the resource usage is presented in Table 6.2, where the resources used are specified for each section. The last row presents the total resources used by the implementation and the computation time required as a percentage of 25 μs. Recall that these results are determined by assuming that all sections are executed sequentially and that the resource usage is an upper bound for the resources required, as a more efficient implementation can be done if the strategy were designed using an HDL instead of HLS, which, as stated before, reduces the development time but does not generate the most efficient implementation in terms of resources and computation time.

From the results shown in Table 6.2, it can be concluded that the OPP–MFCS is viable to run in real–time using a commercial control platform, as the upper bound of resources required and computation time are within limits.

Table 6.2: FPGA resources utilization and computation time of each part of the control strategy.

Section	BRAM	DSP–s	FF	CLUT	Time (μs)
Pre–process	3	36	5646	11931	2.51
$\tilde{\boldsymbol{U}}(k)$ generation	0	5	743	1053	0.49
$\tilde{\boldsymbol{Y}}(k)$ generation	1	45	3395	3228	4.08
Pre–FWA	0	54	2814	4917	1.26
$\bar{\boldsymbol{U}}_{\text{unc}}$ generation	0	15	1310	1453	0.33
FWA	0	14	525	4078	2.94
EP generation	0	0	1419	1915	0.23
$\boldsymbol{U}_{\text{ini}}$ calculation	0	0	35	641	0.11
SDA	0	51	28839	22176	9.75
Total	4	220	44726	51392	21.7
Total (%)	2.9	100.0	42.0	96.6	86.8

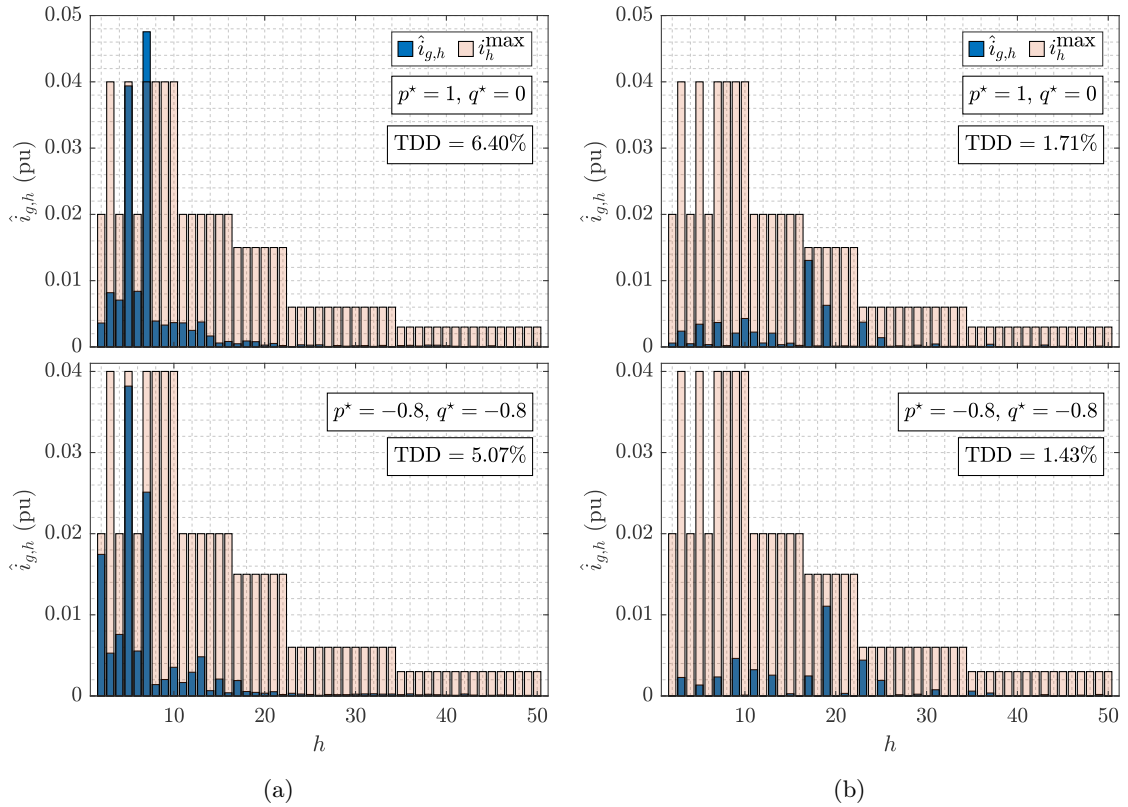


Figure 6.3: Fourier spectra comparison between both control strategies with $\mathcal{S} = \{1, 4, 4, 4, 4\}$ and $w_\iota = s_\iota$. (a) MFCS controller. (b) Proposed OPP–MFCS controller.

6.2 Hardware–in–the-loop results

The proposed control strategy was tested using an OPAL–RT real–time simulator in a HIL experimental setup. The model used is an OP4510, which is equipped with a Xilinx FPGA of the Kintex 7 family and an Intel Xeon E3–1240 V6 Quad–Core @3.7 GHz. The real–time simulation of the system ran in the FPGA of the platform through the eHSx64 electrical solver. The eHSx64 can run the real–time simulation of an electrical system composed of elements from a subset of the Simscape Electrical Matlab library. Using the eHSx64 solver, it is possible to simulate the system under study with a time step of hundreds of nanoseconds. The measurements of the system variables need to be taken out from the simulator; however, the OP4510 used comes with 16 analog inputs and 64 digital outputs, so an OP4520 expansion unit, with 16 analog outputs and 32 analog inputs, was required. The real–time execution of the controller is done in the processor of the OP4510, running at a control period of 25 μs. The measurements, coming from the analog outputs of the OP4520, are fed back to the processor through the analog inputs of the OP4520. The control actions, which are the gating signals of the semiconductors, are taken out from the processor using the digital outputs and are fed to the eHSx64 solver using the digital inputs.

The efficient implementation of the OBRG–MRS, described in Chapter 4, was used so it was possible to run the proposed control strategy in the processor. The hyperparameters were set as $H_r^i = 34$, $H_r^c = H_r^g = 0$, $\texttt{N}_{\texttt{iter}} = 20$, and $\texttt{Max\_Nodes} = 60$. The SDA was not parallelized in the processor, so a larger `Max_Nodes` was required to guarantee a similar performance as in the case analyzed using Vitis HLS. The prediction horizon was set as $\mathcal{S} = \{1, 4, 4, 4, 4\}$ with $w_\iota = s_\iota$. The proposed strategy's dynamic response and steady–state performance were compared with the performance of an MFCS controller adjusted with the same hyperparameters and prediction horizon. The weighting factor of the MFCS controller was adjusted with the value of the Pareto front determined in Chapter 5 that is closest to the 600 Hz of switching frequency, whereas the for the OPP–MFCS controller, it was set to guarantee that fixed switching frequency and symmetric voltages between the three phases were obtained. For the OPP–MFCS, OPPs of $d = 5$ were used, i.e., a switching frequency of 250 Hz.

First, the $I_{g,\text{TDD}}$ and grid codes were evaluated for two steady–state operation points, set by

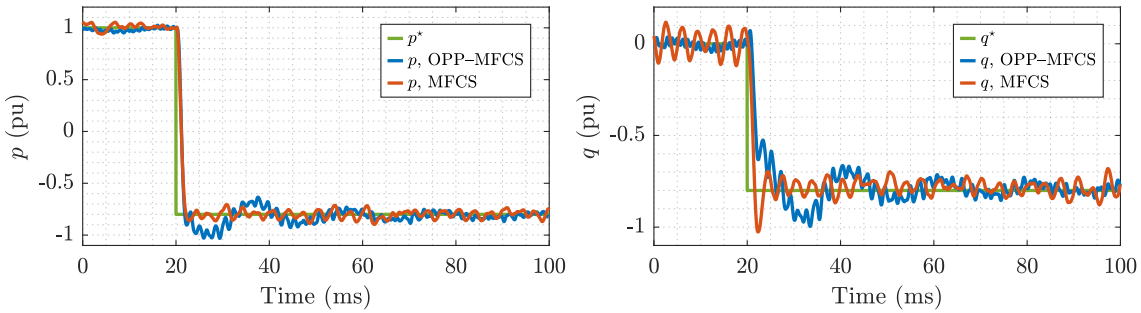


Figure 6.4: Step response comparison between both control strategies with $\mathcal{S} = \{1, 4, 4, 4, 4\}$ and $w_\iota = s_\iota$.

the references $p^\star = 1$ and $q^\star = 0$, and $p^\star = q^\star = -0.8$. The Fourier spectra of a grid current for both operation points are shown in Fig. 6.3. It can be noticed that the MFCS–MPC strategy is capable of meeting the grid codes only in one of the cases, whereas the proposed OPP–MFCS strategy achieves that goal in both cases. Notice how the $I_{g,\mathrm{TDD}}$ is not only smaller than 5% but also considerably smaller than the MFCS controller, getting a reduction of more than 70%. Notice that in the $p^\star = 1$ and $q^\star = 0$ case, some even and triplen harmonics appear in the grid currents. This results from the fact that measurement noise can induce tracking error, which the controller fixes by slightly modifying the steady–state pattern, leading to a slight loss of symmetry between the three phases and the pattern's quarter and half–wave symmetry. However, those harmonics are of low order, and their amplitudes are smaller than the limit defined by the grid codes.

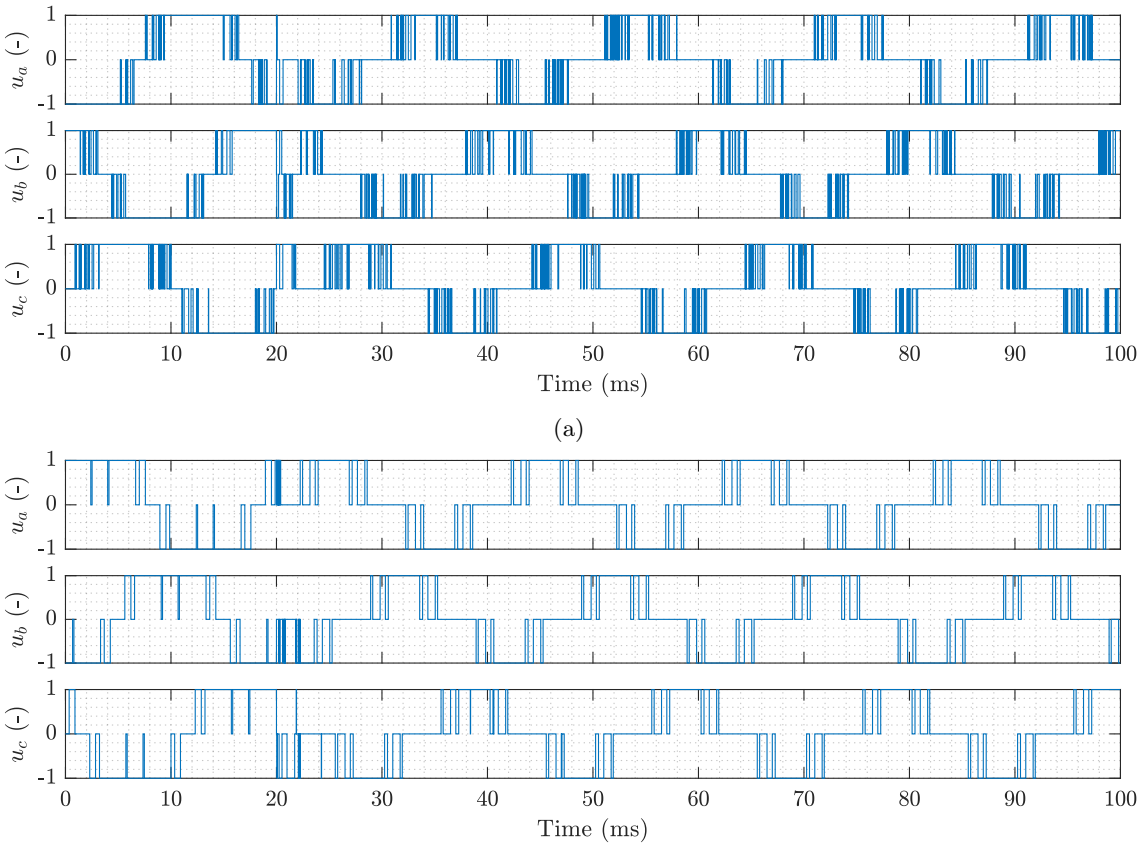


Figure 6.5: Actuation comparison between both control strategies with $\mathcal{S} = \{1, 4, 4, 4, 4\}$ and $w_\iota = s_\iota$. (a) MFCS controller. (b) Proposed OPP–MFCS controller.

The step response of both strategies when transitioning between those two operation points is

shown in Fig. 6.4. It can be noticed that both strategies reach the new references in a fast manner, even though the MFCS–MPC strategy renders a faster response than the proposed OPP–MFCS, which shows some oscillations before getting to steady–state, which is caused by the following. In the HIL implementation, when feeding the measurements of the variables from the FPGA back to the processor unit of the platform, a variable time delay appeared. Even though an additional delay compensation stage slightly compensated it, the errors introduced by the variable delay made the OPP–MFCS strategy modify the steady–state pattern, preventing its performance from being as similar to the OPPs' steady–state performance. Therefore, increasing the amplitude of λ_ℓ^u was necessary to reduce the strategy's sensitivity to the introduced error and guarantee that the steady–state patterns are almost identical to the OPPs. Recall that a larger λ_ℓ^u makes the tracking of the OPPs more relevant in the cost function, forcing the actuation to be similar to the steady–state pattern. This leads to the control actions shown in Fig. 6.5, where the actuation of both strategies is depicted. It is clear how the switching frequency of the MFCS controller is higher than OPP–MFCS, and that the proposed strategy only adds additional switching transitions during transients and quickly applies symmetric patterns with fixed and low switching frequency, with the cost of a slower dynamic response.

With all these tests and results, the performance of the proposed OPP–MFCS strategy was assessed, verifying and concluding that all the control objectives stated in Chapter 3 were fulfilled.

Chapter 7

Conclusions and future work

Grid–connected converters are crucial elements in the integration of new energy sources and renewable energies into the power system, as they allow transferring power from a wide variety of generation plants. However, this transference must always meet demanding requirements stated in technical standards, usually called grid codes. The principal grid codes are related to the currents at the connection point to the system, imposing limits to the amplitudes of specific harmonic components and the currents TDD. Both harmonics amplitudes and TDD can be reduced by including an LCL filter between the converter and the grid, attenuating the distortion of the grid currents at the cost of posing new control challenges, as low resonance frequencies induce oscillations during transient operation, limiting the speed of the controllers to prevent those large oscillations from appearing during transient behavior. Another aspect to consider is the system's efficiency, where low switching frequencies of the converter semiconductors are desired as switching losses are significant in high–power systems. Additionally, a fixed switching frequency is necessary as variable switching frequencies lead to conservative decisions in the design of the filter, increasing its cost.

This work proposed a control framework based on two optimization stages, which was used to develop a control strategy capable of dealing with all objectives mentioned in the paragraph above. The first optimization stage deals with long–term control objectives such as fixed switching frequency, reduced $I_{g,\mathrm{TDD}}$, and symmetrical voltages between the three phases. As long–term objectives are based on steady–state metrics, the optimization problem entails an enormous computational burden. Because of this reason, the entity that executes this stage is the OBRG–OS and is executed offline. In this case, it is based on OPPs. Regarding OPPs, the state–of–the–art formulation assumes that the system dynamics can be modeled as a purely inductive load. In the case of an LCL filter grid–connected, which is a third–order system, approximating the model to a purely inductive load, a first–order model, would lead to suboptimal patterns that do not minimize the TDD of the grid currents. In this work, the transfer function between the grid currents and the converter voltage was included in a novel formulation that directly minimizes the $I_{g,\mathrm{TDD}}$, resulting in optimal patterns that guarantee to minimize the TDD of the grid currents. As the model of the LCL filter was included in the cost function of the proposed OPP formulation, it was possible to use this model to add constraints that force the grid current's harmonic components to be smaller than the limits set by grid codes. This formulation not only minimizes the $I_{g,\mathrm{TDD}}$ but also guarantees that grid codes are met. The formulation that minimizes the $I_{g,\mathrm{TDD}}$ was tested and compared with the traditional one using an L model, showing it can reduce the grid currents TDD even up to the 22% of the traditional one. On the other hand, the proposed formulation, including grid code constraints, was tested considering the limits defined by the IEEE Std. 519–2022. This test proved that the SAs are modified to guarantee that grid code constraints are met by slightly sacrificing the $I_{g,\mathrm{TDD}}$. The results showed that the resulting grid currents TDD and harmonic components were smaller than the limits imposed, which did not happen when the grid code constraints were not part of the problem. To guarantee the scalability of the formulations regarding the number of decision variables and the required computation time, the proposed formulation with grid code constraints was tested for different numbers of SAs per quarter–wave d. The results showed that increasing d did not considerably impact the time required to solve the OPP problem with grid code constraints, proving that the proposed optimization model is perfectly scalable.

The OBRG–MRS takes the solutions derived from the first optimization stage to produce tracking references for the next optimization stage, which is the ROS. The ROS runs in real–time and

uses MFCS–MPC, as it has already been proven to be a viable option for real–time applications using commercial control platforms. To this end, the related optimization problem is solved using the SDA. Regarding MFCS controllers and SDA, it is important to highlight that for high–order systems, long prediction horizons in MFCS controllers have been proven to render a better performance regarding the trade–off between switching frequency and TDD. Additionally, long prediction horizons do not require active damping strategies to deal with the oscillations induced by the LCL filter. This work proposed a generalized variable–step prediction horizon to extend the prediction horizon over time at no additional computation burden in real–time. Each step in the horizon can be generated using a different sampling period with this formulation, allowing longer sampling periods in steps that are far in the horizon while keeping a fine granularity in the first step, which is fundamental to render a fast dynamic response and steady–state performance. This variable–step prediction horizon was tested by doing extensive analyses. In this case, the two degrees of freedom to analyze are the configuration of the prediction horizon and the adjustment of the weighting factors. The results clearly showed the impact of extending the prediction horizon over time. The steady–state performance was improved by shifting the Pareto front of the trade–off between switching frequency and grid currents TDD closer to the origin. On the other hand, faster dynamic responses, with smaller oscillations, overshoots, and undershoots, were achieved by extending the prediction horizon using the proposed strategy.

The efficiency of the SDA is key in extending the prediction horizon in MFCS controllers and developing strategies that combine the MFCS with other entities, which is the case of this strategy. As the OBRG–MRS requires more computation time than a traditional reference generator, reducing the time the SDA uses to find the solution is fundamental. To this end, a generalized exploration pattern of the SDA search tree was proposed, so the nodes explored first are those closer to the initial incumbent, which is usually selected to render a small cost function. This exploration pattern is compatible with the parallelization method proposed in the literature. Additionally, the proposed exploration pattern allows initializing the SDA at the last level of the search tree, reducing in $n_u \cdot N$ the number of nodes explored. The analysis done showed how the proposed exploration pattern and initialization significantly impact the number of nodes explored before the solution is found, exploring the same or fewer nodes than the traditional exploration pattern in 87% of the cases analyzed, proving its effectiveness.

Once the proposed approaches to OPPs and MFCS control, with the corresponding improvements, were verified, the proposed OPP–MFCS strategy was tested. Regarding the proposed control strategy, the steady–state performance of OPP–MFCS can meet grid codes while operating at a low and fixed switching frequency and obtaining symmetric and periodic voltages between the three phases. As the tracking of the sampled OPP is included in the cost function of the MFCS optimization problem, the resulting control actions in steady–state are the sampled OPPs. Therefore, the OBRG–OS guarantees the rendered steady–state performance in the first optimization stage, where the OPPs are calculated. Because the steady–state performance is guaranteed as long as the OPPs are properly tracked, the variable–step prediction horizon and weighting factors were adjusted so the dynamic response of the system was fast, and the oscillations introduced by the filter were quickly attenuated. The OPP–MFCS proved to render a fast dynamic response and a great attenuation of the oscillations while guaranteeing the steady–state performance required.

Including the tracking of the sampled OPP in the cost function of the MFCS problem allows using it as the initial incumbent for the SDA as long as it is feasible. The proposed initial incumbent selection consisted in using the sampled OPP every time it is feasible and using the traditional educated guess when it is not. With this selection process, the SDA found the solution exploring the same or fewer nodes than the traditional educated guess selection in 87.07% of the 5601 cases analyzed, proving the improvement in computational efficiency using the proposed approach.

The steady–state performance of the OPP–MFCS was analyzed in detail. It was shown that voltages are periodic signals and symmetric between the three phases, resulting in discrete grid currents' Fourier spectra that are also predictable because of the usage of OPPs in the strategy. The steady–state analysis proved that the proposed strategy can meet grid codes while operating at a low and fixed switching frequency of the semiconductors.

The final part of the analysis evaluated the strategy's real implementation. Using Vitis HLS, it was proved that running the proposed OPP–MFCS strategy in real–time using a commercial control platform is viable. The results obtained by synthesizing the strategy in Vitis HLS proved how it is possible to fit the whole design in a Xilinx XC7Z020 Zynq–7000 FPGA, which is a low–cost FPGA. It was also proved that the FPGA considered is capable of executing the algorithm

in less than 25 μs, which is the target control period. These results concluded that the proposed control strategy is a viable option for control of power converters. Once the strategy's viability was proved, it was tested in a HIL experimental setup using the OPAL–RT OP4510 real–time simulator. The proposed strategy was compared with the proposed variable–step prediction horizon MFCS controller. The OPP–MFCS strategy outperformed the MFCS controller in steady–state operation, as it could meet the grid codes while operating a low and fixed switching frequency, with periodic and symmetric voltages between the three phases. Regarding the dynamic response, the OPP–MFCS strategy rendered a fast dynamic response; however, the MFCS controller was faster, as the switching frequency was higher. With these experimental results, it was concluded that the proposed control strategy can meet all the defined control objectives.

The analysis done in this work proved the concept of the OPP–MFCS strategy for grid–connected applications, and from the results obtained, it can be concluded that continuing researching and improving this strategy is worth it. As future work, the first step would be developing an efficient FPGA implementation so it is possible to extend the number of nodes explored by the SDA, therefore reducing the solution suboptimality during transients. In the same line, comparing the real–time efficiency of the FWA with other state–of–the–art methods is let as future work. From the results shown, it is expected that the number of additional explored nodes required is small. In the same line, it would be beneficial to evaluate different cost functions in the MFCS problem. A different cost function may reduce the number of additional switching transitions during transients so the losses are not temporarily raised. Additionally, benchmarking the strategy with other OPP–based MPC strategies, such as the small–signal MP3C, is necessary. This would give a big picture of the qualities and limitations of the strategy.

In general, to introduce new technologies and control strategies to the industry, the performances of the strategies in "corner cases" must be evaluated. Testing the OPP–MFCS strategy during faults, and voltage sags and unbalances in the grid is mandatory in the next stage of this research path.

As a further stage, extending this strategy to other topologies would pose new challenges that may drive the research. In particular, applying the OPP–MFCS controller or the proposed framework to the control problem of modular multilevel converters in different applications would be a great challenge. If the OPP–MFCS controller is used, the first step would be developing dedicated OPPs for the specific topology. Then, adding other control layers, such as circulating currents control, capacitors voltages balances, and other challenges that will depend on the approach taken (e.g., clusters energy balances, total energy balance, etc.). Finally, develop a real–time controller based on MFCS–MPC to meet the application's control objectives properly.

Bibliography

[1] IEEE standard for harmonic control in electric power systems. *IEEE Std 519-2022 (Revision of IEEE Std 519-2014)*, pages 1–31, Aug 2022.

[2] Tinus Dorfling, Hendrik du Toit Mouton, and Tobias Geyer. Generalized model predictive pulse pattern control based on small-signal modeling—part 2: Implementation and analysis. *IEEE Transactions on Power Electronics*, 37(9):10488–10498, Sep. 2022.

[3] U.S. State Renewables Portfolio & Clean Electricity Standards: 2023 Status Update. `https://emp.lbl.gov/publications/us-state-renewables-portfolio-clean` (Accessed: 2023-09-04).

[4] REN21. Renewables 2022: Global Status Report. `https://www.ren21.net/gsr-2022/` (Accessed: 2023-09-04).

[5] Ministerio de Energía. Energía 2050: Política energética de Chile, 2016.

[6] J.M. Carrasco, L.G. Franquelo, J.T. Bialasiewicz, E. Galvan, R.C. PortilloGuisado, M.A.M. Prats, J.I. Leon, and N. Moreno-Alfonso. Power-electronic systems for the grid integration of renewable energy sources: A survey. *IEEE Transactions on Industrial Electronics*, 53(4):1002–1016, June 2006.

[7] F. Blaabjerg, Zhe Chen, and S.B. Kjaer. Power electronics as efficient interface in dispersed power generation systems. *IEEE Transactions on Power Electronics*, 19(5):1184–1194, Sep. 2004.

[8] Zhongting Tang, Yongheng Yang, and Frede Blaabjerg. Power electronics: The enabling technology for renewable energy integration. *CSEE Journal of Power and Energy Systems*, 8(1):39–52, 2022.

[9] IEEE standard for interconnection and interoperability of distributed energy resources with associated electric power systems interfaces. *IEEE Std 1547-2018 (Revision of IEEE Std 1547-2003)*, pages 1–138, April 2018.

[10] IEEE recommended practice for monitoring electric power quality. *IEEE Std 1159-2019 (Revision of IEEE Std 1159-2009)*, pages 1–98, Aug 2019.

[11] IEEE standard for power electronics open system interfaces in zonal electrical distribution systems rated above 100 kw. *IEEE Std 1826-2020 (Revision of IEEE Std 1826-2012)*, pages 1–44, Nov 2020.

[12] Aleksandr Reznik, Marcelo Godoy Simões, Ahmed Al-Durra, and S. M. Muyeen. *LCL* filter design and performance analysis for grid-interconnected systems. *IEEE Transactions on Industry Applications*, 50(2):1225–1232, March 2014.

[13] Michael Bierhoff, Ramy Soliman, and José R. Espinoza C. Analysis and design of grid-tied inverter with LCL filter. *IEEE Open Journal of Power Electronics*, 1:161–169, 2020.

[14] Erika Twining and D.G. Holmes. Grid current regulation of a three-phase voltage source inverter with an LCL input filter. *IEEE Transactions on Power Electronics*, 18(3):888–895, May 2003.

[15] Tobias Geyer. *Model Predictive Control of High Power Converters and Industrial Drives*. Wiley, 2016.

[16] R. R. Negenborn and J. M. Maestre. Distributed model predictive control: An overview and roadmap of future research opportunities. *IEEE Control Systems Magazine*, 34(4):87–97, Aug 2014.

[17] Petros Karamanakos, Eyke Liegmann, Tobias Geyer, and Ralph Kennel. Model predictive control of power electronic systems: Methods, results, and challenges. *IEEE Open Journal of Industry Applications*, 1:95–114, 2020.

[18] T. Geyer and D. E. Quevedo. Multistep finite control set model predictive control for power electronics. *IEEE Transactions on Power Electronics*, 29(12):6836–6846, Dec 2014.

[19] Petros Karamanakos and Tobias Geyer. Guidelines for the design of finite control set model predictive controllers. *IEEE Transactions on Power Electronics*, 35(7):7434–7450, July 2020.

[20] Annika Birth, Tobias Geyer, Hendrik du Toit Mouton, and Martinus Dorfling. Generalized three-level optimal pulse patterns with lower harmonic distortion. *IEEE Transactions on Power Electronics*, 35(6):5741–5752, June 2020.

[21] T. Geyer, N. Oikonomou, G. Papafotiou, and F. D. Kieferndorf. Model predictive pulse pattern control. *IEEE Transactions on Industry Applications*, 48(2):663–676, March 2012.

[22] Javier Napoles, Jose I. Leon, Ramon Portillo, Leopoldo G. Franquelo, and Miguel A. Aguirre. Selective harmonic mitigation technique for high-power converters. *IEEE Transactions on Industrial Electronics*, 57(7):2315–2323, July 2010.

[23] Kamran Sharifabadi, Lennart Harnefors, H.-P Nee, Staffan Norrga, and Teodorescu. *Design, control and application of modular multilevel converters for HVDC transmission systems*. 2016.

[24] J. Luo, K. Lin, J. Li, Y. Xue, and X. Zhang. Cost analysis and comparison between modular multilevel converter (mmc) and modular multilevel matrix converter (m3c) for offshore wind power transmission. In *15th IET International Conference on AC and DC Power Transmission (ACDC 2019)*, pages 1–6, Feb 2019.

[25] S. Alepuz, S. Busquets-Monge, J. Bordonau, J. Gago, D. Gonzalez, and J. Balcells. Interfacing renewable energy sources to the utility grid using a three-level inverter. *IEEE Transactions on Industrial Electronics*, 53(5):1504–1511, Oct 2006.

[26] M. Li, H. Yang, R. Zhao, T. Zheng, C. Si, Y. Lu, and Y. Yang. Comparative study on the operating area of M3C and b2b MMC for soft open point application. In *2019 10th International Conference on Power Electronics and ECCE Asia (ICPE 2019 - ECCE Asia)*, pages 1205–1212, May 2019.

[27] Jayabal Gayathrimonicka Subarnan. 4 - multilevel inverters: an enabling technology. In A. Hina Fathima, N. Prabaharan, K. Palanisamy, Akhtar Kalam, Saad Mekhilef, and Jackson. J. Justo, editors, *Hybrid-Renewable Energy Systems in Microgrids*, Woodhead Publishing Series in Energy, pages 61–80. Woodhead Publishing, 2018.

[28] G. Konstantinou and B. Hredzak. 6 - power electronics for hybrid energy systems. In Ersan Kabalci, editor, *Hybrid Renewable Energy Systems and Microgrids*, pages 215–234. Academic Press, 2021.

[29] M. Liserre, F. Blaabjerg, and S. Hansen. Design and control of an LCL-filter-based three-phase active rectifier. *IEEE Transactions on Industry Applications*, 41(5):1281–1291, Sep. 2005.

[30] Hernan Miranda, Remus Teodorescu, Pedro Rodriguez, and Lars Helle. Model predictive current control for high-power grid-connected converters with output LCL filter. In *2009 35th Annual Conference of IEEE Industrial Electronics*, pages 633–638, Nov 2009.

[31] Rafael Peña-Alzola, Marco Liserre, Frede Blaabjerg, Rafael Sebastián, Jörg Dannehl, and Friedrich Wilhelm Fuchs. Analysis of the passive damping losses in LCL-filter-based grid converters. *IEEE Transactions on Power Electronics*, 28(6):2642–2646, June 2013.

[32] Joerg Dannehl, Marco Liserre, and Friedrich Wilhelm Fuchs. Filter-based active damping of voltage source converters with LCL filter. *IEEE Transactions on Industrial Electronics*, 58(8):3623–3633, Aug 2011.

[33] Poh Chiang Loh and D.G. Holmes. Analysis of multiloop control strategies for lc/cl/LCL-filtered voltage-source and current-source inverters. *IEEE Transactions on Industry Applications*, 41(2):644–654, 2005.

[34] James Scoltock, Tobias Geyer, and Udaya 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*, 30(10):5926–5937, Oct 2015.

[35] Joanie M. C. Geldenhuys, Hendrik du Toit Mouton, Arnold Rix, and Tobias 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)*, pages 1–6, June 2016.

[36] Gene F. Franklin, Michael L. Workman, and Dave Powell. *Digital Control of Dynamic Systems*. Addison-Wesley Longman Publishing Co., Inc., USA, 3rd edition, 1997.

[37] T. Dorfling, H. du Toit Mouton, T. Geyer, and P. Karamanakos. Long-horizon finite-control-set model predictive control with nonrecursive sphere decoding on an fpga. *IEEE Transactions on Power Electronics*, 35(7):7520–7531, July 2020.

[38] Leopoldo Garcia Franquelo, Javier Napoles, RamÓn C. Portillo Guisado, JosÉ Ignacio Leon, and Miguel A. Aguirre. A flexible selective harmonic mitigation technique to meet grid codes in three-level pwm converters. *IEEE Transactions on Industrial Electronics*, 54(6):3022–3029, Dec 2007.

[39] T. Geyer and D. E. Quevedo. Performance of multistep finite control set model predictive control for power electronics. *IEEE Transactions on Power Electronics*, 30(3):1633–1644, March 2015.

[40] Ricardo P. Aguilera, Pablo Acuña, Pablo Lezana, Georgios Konstantinou, Bin Wu, Steffen Bernet, and Vassilios G. Agelidis. Selective harmonic elimination model predictive control for multilevel power converters. *IEEE Transactions on Power Electronics*, 32(3):2416–2426, March 2017.

[41] Cristóbal González, Alejandro Angulo, and Fernando Mancilla–David. Enhancing multistep finite control set performance of 3L–NPC converters using optimal pulse patterns. In *IECON 2021 – 47th Annual Conference of the IEEE Industrial Electronics Society*, pages 1–6, Oct 2021.

[42] Eduardo Zafra, Sergio Vazquez, Carlos Regalo, Vicente Baena Lecuyer, Abraham Marquez Alcaide, Jose I. Leon, and Leopoldo G. Franquelo. Parallel sphere decoding algorithm for long-prediction-horizon FCS-MPC. *IEEE Transactions on Power Electronics*, 37(7):7896–7906, 2022.

[43] Pablo Acuna, Christian A. Rojas, Roky Baidya, Ricardo P. Aguilera, and John E. Fletcher. On the impact of transients on multistep model predictive control for medium-voltage drives. *IEEE Transactions on Power Electronics*, 34(9):8342–8355, 2019.

[44] Roky Baidya, Ricardo P. Aguilera, Pablo Acuña, Tobias Geyer, Ramón A. Delgado, Daniel E. Quevedo, and Hendrik du Toit Mouton. Enabling multistep model predictive control for transient operation of power converters. *IEEE Open Journal of the Industrial Electronics Society*, 1:284–297, 2020.

[45] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956.

[46] Alberto Del Pia, Santanu S. Dey, and Marco Molinaro. Mixed-integer quadratic programming is in np. *Math. Program.*, 162(1–2):225–240, mar 2017.