Adaptive Reference Governor for DC–DC Converters Based on Model Predictive Control

Adaptive Reference Governor for DC-DC Converters based on Model Predictive Control

Gionata Cimini, Riccardo Felicetti, *Member, IEEE*, Francesco Ferracuti, *Member, IEEE*, Luca Cavanini, and Andrea Monteriù, *Member, IEEE*

***Abstract*—In this paper, we propose a time-varying Model Predictive Control (MPC)-based scheme to enhance the dynamic performance of DC-DC converters. The proposed approach employs MPC as a Reference Governor (RG), addressing industrial certification constraints that may limit modifications to the low-level controller. To accommodate the computational limitations of conventional control boards, we introduce a highly efficient real-time optimization algorithm for solving equality constrained Quadratic Programming (QP) problems. The algorithm is based on a tailored QR factorization that outperforms well-known linear algebra libraries, and it is shown to be superior to condensing with states elimination. Furthermore, we implement an efficient Recursive Least Squares (RLS) method to provide a linear-time varying model for the adaptive MPC-based RG. No information regarding the topology of the converter nor the structure of the low-level controller is required for such adaptation, making the proposed method self-tuning and eliminating the need for prior identification steps. The proposed control scheme has been tested on various simulated and real DC-DC converters, demonstrating its computational and memory efficiency, as well as its versatility across different converter topologies.**

***Index Terms*—Reference governor, model predictive control, power converter.**

I. Introduction

DC-DC power converters are pivotal components in modern electronics. They play a foundational role in voltage regulation, efficient conversion of energy and its transfer, and power distribution within electronic systems. For instance, buck, boost, and buck-boost converters are commonly used for voltage regulation in DC microgrids [1], brushless DC motors [2], automotive [3], and consumer electronics [4]. Moreover, they are an enabler in the transition to renewable energy, for instance, in the case of PhotoVoltaic (PV) power generation [5], where multiple PV modules are connected in series to minimize costs, allowing for a single Maximum Power Point Tracking (MPPT) DC-AC power converter.

Hence, understanding and enhancing the performance of DC-DC converters is important for improving efficiency and reliability in many industrial fields. Several advanced control algorithms have been investigated to fight the challenges of power converters' control, among which we cite sliding-mode [6], passivity-based [7] and MPC [8]. Although these methodologies are mature and demonstrate excellent results, standard controllers tend to dominate the manufacturers' production for industrial applications [9], [10]. Replacing a low-level controller of a power converter is often infeasible or unsuitable, and even when it might be physically possible, industrial certifications for stability and robustness might prevent the change. As an example we mention code freezes in production programs or designs that include safety elements out of context (see ISO 26262 for automotive). Therefore, in this paper we address the topic of enhancing the performance, and extend the set of control features, of power converters that come with a controller, either standard or advanced, that cannot be replaced.

Considering the aforementioned factors, *reference governors* have recently been garnering increased interest in the realm of power converter control. The method assumes that a suitable low-level controller, often unknown, is capable of reference tracking, and employs a second control loop, executed at a lower frequency, that regulates the reference of the low-level controller [11]. Most often, such outer loop is designed as an MPC controller, that minimizes a performance index subject to the constraints imposed by the dynamics of the closed-loop system (the so called pre-compensated converter). Examples of reference governors for power converters include buck [12], [13], boost [14], and buck-boost [15] DC-DC converters, as well as AC-DC applications [16]. Additionally, the presence of an MPC loop enables appealing features such as reference preview and fault tolerant control [17].

The above contributions share a common trait: the mathematical model at the basis of the MPC law is assumed to be a time-invariant white-box model. This represents a limitation, because power converters behaviour depends on various factors, including converter topology, its components, the load characteristics, and the operating conditions. Assuming knowledge of all these aspects is unrealistic. Moreover, the reference governor necessitates a closed-loop model, which includes the low-level controller, and whose mathematical law is probably not disclosed by the manufacturer. The objective of this paper is to overcome these limitations by proposing an adaptive reference governor that combines a linear time-variant MPC based on a black-box model that is identified in real-time.

Several contributions deal with online identification of power converters. In [18], [19], the buck-boost converter is identified online by RLS. The closed-loop poles are assigned by solving the associated Diophantine equations. The controller shows an evident ringing effect. In [20], the buck-boost converter is identified online using modified RLS. The same

(Corresponding author: Riccardo Felicetti)

G. Cimini is with ODYS S.r.l., 20159 Milano, Italy (e-mail: gionata.cimini@odys.it).

R. Felicetti, F. Ferracuti, L. Cavanini, and A. Monteriù are with the Department of Information Engineering, Università Politecnica delle Marche, Via Brecce Bianche 12, Ancona, 60131, Italy (e-mail: r.felicetti@univpm.it; f.ferracuti@univpm.it; l.cavanini@univpm.it; a.monteriu@univpm.it).

L. Cavanini is also with Industrial Systems and Control Ltd, Culzean House, 36 Renfield Street, Glasgow G2 1LU, UK (e-mail: l.cavanini@isc-ltd.com).

approach under parameter variations is proposed in [21], where implicit MPC is used for control. However, the online complexity is not discussed. The authors of [22] propose a nonlinear least squares estimation of several DC-DC converters. The method is accurate, but it is run offline and it requires some iterations to converge. All these works assume that the original embedded controller can be replaced, which is the limitation we want to get rid of, and thus the objective is to identify a closed-loop system. Simultaneously, we are interested in the *persistence of excitation*, a well-known problem in system identification and adaptive control: the input must satisfy the excitation conditions for effective and robust parameter identification [23]. Practically speaking, simple signals such as constant or step inputs (or references, in the case of closed-loop systems) are known to be non-persistently exciting [23]. A potential solution is to alter the input to achieve persistent excitation which is impractical in many applications. Hence, some methods to deal with non-persistent excitations include stopping the update of the parameters when the input is not exciting [24], or using concurrent learning [25] where previous samples are introduced in the update of the parameters. To overcome all the above limitations we propose a first order RLS identification (RLS_{FO}) of the entire closed-loop scheme. Non-persistent excitation is addressed by stopping the update of the parameters when a simplified condition on persistent excitation is not met.

The computational feasibility of an MPC controller on the cost-effective control boards typically used for power conversion is a further concern [8], [26], especially due to control frequencies ranging in 1kHz-100kHz. The reference governor has to drive the reference of the inner controller, thus techniques based on Finite Control Set (FCS) are not an option [27]. Similarly we cannot rely on explicit MPC [28], [29] because the model is time-varying. We address this challenge developing a novel real-time optimization algorithm (MPC_{FO}) for equality constrained QP problems which is based on the Schur complement and on a tailored QR factorization. The method is shown to be vastly superior to well-known linear algebra libraries, and to alternative approaches like standard condensing.

What is here proposed and experimentally validated is a plug-and-play solution to improve the transient behaviour of DC-DC converters that:

- fits a wide range of DC-DC converters,
- is self-tuning (i.e. does not require any modeling effort nor prior identification steps),
- is proven to run in real-time on cheap, conventional control boards,
- does not require to alter the original controller.

To our knowledge, this is the first attempt to implement an adaptive reference governor for DC-DC converters with real-time identification, which helps facing different operating conditions and system nonlinearities. The proposed method is well suited for photovoltaic, consumer electronics, and automotive applications, where adaptation is essential to manage parameter variations that may arise from nonlinear behaviours in different operating conditions. It is particularly effective for time-varying voltage references, as seen in photovoltaic systems implementing MPPT, which requires precise transient behaviour to optimize power generation. In consumer electronics, the method could be applied to rapid charging protocols, adapting to the time-varying load provided by lithium batteries and effectively managing output voltage negotiation. For automotive systems, it could help manage voltage fluctuations due to varying electrical loads or regenerative braking protecting devices and maintaining functionality during transients. As a by-product, this paper introduces a very efficient MPC algorithm to control any system (not necessarily a pre-compensated one) that can be modeled as a time-varying first-order model, with closed loop DC-DC converters serving as a primary application example.

The method is tested in simulation on various devices, such as buck-boost, boost, and buck converters, demonstrating its effectiveness. Experimental tests are also conducted on a buck converter. The setup includes a C2000 Delfino MicroController Unit (MCU) LAUNCHXL-F28379D, paired with a Buck Converter BoosterPack by Texas Instruments™. The proposed adaptive reference governor runs flawlessly at 20 kHz. The computational and memory impact of the whole method is also analytically investigated.

The paper is organized as follows. Section II describes the system identification procedure and the RLS_{FO} algorithm. The reference governor formulation is detailed in Section III and the real-time optimization algorithm MPC_{FO} is introduced in IV-B. The simulation and experimental results are proposed in Section V and Section VI, respectively.

A. Notation

We denote by $\mathbb{R}$ the set of real numbers, $\mathbb{R}_{>0} = \{x \in \mathbb{R} \mid x > 0\}$ and $\mathbb{Z}_{\geq j}$ the set of integers greater or equal than j. Given a vector $x \in \mathbb{R}^n$, $\|x\|$ is its Euclidean norm, $\dim(x) \equiv n$ is its cardinality, and $\text{diag}(x)$ is the diagonal matrix of $\mathbb{R}^{n\times n}$ formed from x. We also define $x^{\{m\}} \in \mathbb{R}^{nm} = \begin{bmatrix} x^\intercal & \dots & x^\intercal \end{bmatrix}^\intercal$, $m \in \mathbb{Z}_{\geq 0}$. We have also that $x^{\{0\}}$ is the empty vector. $\mathbb{S}^n_{++}$ denotes the set of positive definite matrices in $\mathbb{R}^{n\times n}$, whereas $\mathbb{D}^n_{++} \subset \mathbb{S}^n_{++}$ is the subset of diagonal positive definite matrices. Given two vectors $x, y \in \mathbb{R}^n$, their inner product is $\langle x, y\rangle = \sum_{i=1}^n x_i y_i$. For $x \in \mathbb{R}^n$ and $W \in \mathbb{S}^n_{++}$, $||x||_W = \sqrt{\langle x, Wx\rangle}$. Given a matrix $A \in \mathbb{R}^{n\times m}$, $A_{i,j}$ is the element of A in the i-th row and j-th column. We also denote by $A_{i:j,k:h}$ the sub-matrix collecting the entries of A that go from row index i to j, and from column index h to k. The matrix $0_m \in \mathbb{R}^{m\times m}$ is a square zero matrix, while $\bar{0}_m \in \mathbb{R}^m$ is the zero column vector. The matrix $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix of dimension n. Given a symmetric matrix $K = \left[\begin{smallmatrix} A & B^\intercal \\ B & C \end{smallmatrix}\right]$ with A nonsingular, the Schur complement of A in K is defined by K/A.

II. Real-Time System Identification

In this section, first, we introduce the well-known RLS algorithm with the forgetting factor. Then we address the problem of persistent excitation, as it affects severely the DC-DC converters, and a lightweight solution is proposed. Finally, we elaborate an algorithm tailored to closed loop DC-DC

converter identification, and its computational complexity is examined.

A. RLS identification

Let us consider a Single-Input Single-Output (SISO) linear system in the autoregressive exogenous form [30]:

$$\underbrace{\left(1-\sum_{i=1}^{n} a_k(i)q^{-i}\right)}_{A_k} y_k = \underbrace{\sum_{i=1}^{m} b_k(i)q^{-i-\tau}}_{B_k} u_k + \eta_k \tag{1}$$

where $y_k \in \mathbb{R}$ is the output, $u_k \in \mathbb{R}$ is the input, $\eta_k \in \mathbb{R}$ is the error term, q^{-1} is the delay operator such that $q^{-i}y_{k+i} \equiv y_k$ and $\tau \in \mathbb{Z}_{\geq 0}$ is the system delay, while $n, m \in \mathbb{Z}_{\geq 0}$ represent the order of the time-varying polynomials A_k and B_k, respectively. Then, (1) is equivalent to:

$$y_k = \theta_k^{*\intercal}\phi_k + \eta_k \tag{2a}$$
$$\theta_k^* = \big(a_k(1), \dots, a_k(n), b_k(1), \dots, b_k(m)\big)^\intercal \tag{2b}$$
$$\phi_k = \big(y_{k-1}, \dots, y_{k-n}, u_{k-\tau-1}, \dots u_{k-\tau-m}\big)^\intercal \tag{2c}$$

where θ_k^* represents the true parameter values and ϕ_k is the regression term.

Let us consider the following optimization problem [30]:

$$\hat{\theta}_k = \arg\min_{\hat{\theta}_k} J(\hat{\theta}_k) = \sum_{i=1}^{k}\left(\prod_{j=i+1}^{k}\lambda_j\right)\varepsilon_i^2 \tag{3a}$$
$$\text{s.t. } \varepsilon_i = y_i - \hat{\theta}_{i-1}^\intercal\phi_i \tag{3b}$$
$$\phi_i = \big(y_{i-1}, \dots, y_{i-n}, u_{i-\tau-1}, \dots, u_{i-\tau-m}\big)^\intercal \tag{3c}$$

where $\hat{\theta}_k$ represents the estimation of θ_k^* at the instant k (i.e., $\hat{\theta}(k|k)$ in the conventional filtering notation, e.g., [31]), ε_k is the one-step prediction error, and $\lambda_k \in (0, 1]$ is the forgetting factor at the time k. Usual values are $\lambda_k > 0.95$, and often $\lambda_k = \lambda > 0.95$. Taking into account the case of constant λ_k for simplicity, the smaller the λ, the lower the weight of the past samples in (3a). The choice of λ is then a trade-off between slow variation of the parameters (large λ) and fast adaptation capabilities (small λ). On the contrary, if $\lambda = 1$ is chosen, all the past errors are weighted equally in (3a), and the so-called ordinary least squares problem follows as a special case.

The general solution of (3) is given by the well known RLS with forgetting factor algorithm [30]:

$$\hat{\theta}_k = \hat{\theta}_{k-1} + K_k\left[y_k - \phi_k^\intercal\hat{\theta}_{k-1}\right] \tag{4a}$$
$$K_k = P_k\phi_k \tag{4b}$$
$$P_k = \frac{1}{\lambda_k}\left[P_{k-1} - \frac{P_{k-1}\phi_k\phi_k^\intercal P_{k-1}}{\lambda_k + \phi_k^\intercal P_{k-1}\phi_k}\right] \tag{4c}$$

B. Identification of pre-compensated DC-DC converters

Let us consider the closed-loop DC-DC converter denoted with a dashed box in Fig. 1. In light of the operation of controlled DC-DC converters, we propose the following set of assumptions.

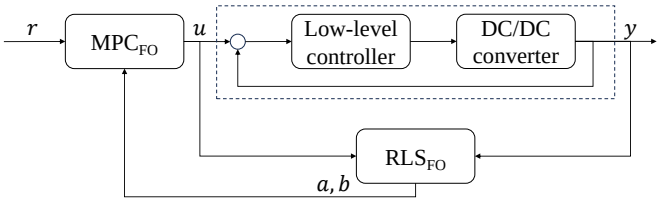


Fig. 1: Complete control scheme.

Assumption 1: The reference voltage r_k is piecewise constant.

Assumption 2: The system may undergo abrupt parameter variations (e.g., piecewise constant load resistance).

Assumption 3: The closed-loop system (i.e., the DC-DC converter and its control loop) shows a unitary gain in steady state.

Assumption 4: The closed-loop system dynamics can be modeled using a SISO first order linear system, possibly time variant.

Assumptions 1–2 characterize many practical scenarios where DC-DC converters are employed, such as driving a DC motor at constant speed (with possible load variations) or charging/discharging a DC battery (with possible connection/disconnection of the loads), where both the reference voltage and the load can be assumed piecewise constant. Moreover, in the literature, it is customary to test DC-DC converters under step-wise variations of references and loads (see for example [13], [32]). These assumptions, which describe usual operational conditions, do not simplify the problem; on the contrary, they pose a challenge for system identification, as simple signals such as constant or step inputs are known to be non-persistently exciting [23]. The persistence of excitation problem is faced in Section II-C.

Assumption 3 means that the low-level controller (commonly a PI controller) is assumed to make the system stable and is capable of eventually tracking the given voltage reference. In practice, low-level controllers directly manipulate the duty cycle and enforce appropriate limitations on it. In case Assumption 3 is false, i.e., the controller is not able to track the reference, the reference governor cannot work properly. However, Assumption 3 is clearly valid when the device is faultless and the operating conditions are feasible. Finally, we note that the input/output magnitude ratio can affect the convergence speed of identification algorithms [33]. However, according to Assumption 3, the input/output ratio is unitary, thus ideal for identification.

With hierarchical control structures, simplification of the low-level closed-loop model is crucial to capture the dominant dynamics of interest for the upper level, while keeping the complexity of the prediction model tractable [34]. First-order approximation, often used in robotics, chemical or biological systems, is one of the possible choices [35], [36]. We show that a linear first-order model adjustable in real-time (Assumption 4) works well in terms of dominant dynamics explainability for several converters topologies. The simplification cannot account for secondary effects, such as parasitic dynamics, high-frequency switching noise or protection cir-

cuits, but those aspects are commonly handled by the faster, low-level, controller. Literature contributions have shown that Assumption 4 is reasonable [12], [13] for basic converters' topologies, and that will be evaluated as well while testing the proposed algorithm. Under Assumption 4, model (1) boils down to:

$$y_k - a_k y_{k-1} = b_k u_{k-1} \tag{5}$$

with $a_k \in \mathbb{R}$ and $b_k \in \mathbb{R}$, thus defining:

$$\phi_k = \begin{bmatrix} y_{k-1} & u_{k-1} \end{bmatrix}^\intercal \tag{6a}$$
$$\theta_k = \begin{bmatrix} a_k & b_k \end{bmatrix}^\intercal \tag{6b}$$

and with the RLS routine in (4), the estimation $\hat{\theta}_k$ is obtained.

Please note that taking into account the constraint in Assumption 3 (i.e., unitary gain in steady state):

- improves the significance of the model,
- improves the convergence of the parameter estimation,
- reduces the computational complexity if tackled correctly, as will be clearer in the following.

The gain in steady state for (5) is:

$$G_k = \frac{b_k}{1 - a_k} \tag{7}$$

so, substituting b_k with $1 - a_k$ in (5) to force a unitary gain $G_k = 1$, we get:

$$y_k - u_{k-1} = a_k(y_{k-1} - u_{k-1}) \tag{8}$$

Hence, re-defining

$$\phi'_k = y_{k-1} - u_{k-1} \tag{9a}$$
$$y'_k = y_k - u_{k-1} \tag{9b}$$
$$\theta_k = a_k \tag{9c}$$

and noting that u_k, y'_k, ϕ'_k, a_k, b_k, $\theta_k \in \mathbb{R}$, we can apply the algorithm (4), which reduces to:

$$\hat{\theta}_k = \hat{\theta}_{k-1} + K_k \left[y'_k - \phi'_k \hat{\theta}_{k-1} \right] \tag{10a}$$
$$K_k = P_k \phi'_k \tag{10b}$$
$$P_k = \frac{P_{k-1}}{\lambda_k + P_{k-1}\phi'^2_k} \tag{10c}$$

with P_k, K_k, $\hat{\theta}_k$ clearly scalar by construction.

C. Persistence of excitation

Simple input signals, such as constant or step inputs, are known to be non-persistently exciting [23]: if unmanaged, the outcome is parameter drift and/or covariance blowup [24]. Among the possible solutions to overcome this issue, altering the input to achieve persistent excitation is unacceptable for DC-DC converters. To minimize the computational effort, we employ a straightforward method, which consists of stopping the parameters update when the input is not exciting [24].

A generic regressor X_k is said to be persistently exciting [24] if there exists $c_1 > 0$ and $c_2 > 0$ such that:

$$c_1 I \leq \sum_{i=k-p+1}^{k} X_i X_i^\intercal \leq c_2 I \tag{11}$$

Algorithm 1 RLS$_{\text{FO}}$: RLS first order identification imposing unitary gain.

Input: Data $y_{k-1}, u_{k-1}, y_k \in \mathbb{R}$, former coefficients $a, b \in \mathbb{R}$, former covariance $P \in \mathbb{R}$, parameters $\epsilon_{id}, \sigma_{id}, \lambda, a_{max} \in \mathbb{R}_{>0}$.

$\phi \leftarrow y_{k-1} - u_{k-1}$;
$\delta \leftarrow \phi^2$;
if $\left(\delta > \sigma_{id}(u_{k-1}^2 + \epsilon_{id})\right)$ **then**
  $P \leftarrow P/(\lambda + P\delta)$;
  $a \leftarrow a + P\phi(y_k - u_{k-1} - a\phi)$;
  if $a < 0$ **then**
    $a \leftarrow 0$;
  else if $a > a_{max}$ **then**
    $a \leftarrow a_{max}$;
  end if
  $b \leftarrow 1 - a$;
end if

Output: Updated coefficients $a, b \in \mathbb{R}$, updated covariance $P \in \mathbb{R}$.

for all k and for some $p \geq \dim(\phi(\cdot)) = n + m$. Testing (11) is rarely employed in practice (and especially in real time on control boards) because it is not computationally feasible in general [24]. However, in the identification of a pre-compensated DC-DC converter under Assumptions 1–4, the regressor ϕ'_k is scalar, as shown in (9a). Therefore, (11) boils down to requiring a finite $\phi'_k \neq 0$ with $p = 1$. According to Assumption 3, in steady state, $\phi'_k = 0$ holds and (11) is not satisfied. So, the system must not be in steady state for system identification, while transients carry useful information. In order to enforce a sufficient excitation, we enable system identification when the following condition is satisfied:

$$\phi'^2_k > \sigma_{id}(u_{k-1}^2 + \epsilon_{id}) \tag{12}$$

where $\sigma_{id} \in (0, 1)$ is a suitable threshold and $\epsilon_{id} \in \mathbb{R}_{>0}$ is an arbitrarily small regularization coefficient to avoid fitting noise and ripples. Neglecting ϵ_{id}, (12) is essentially equivalent to (11), where $c_1 = \sigma_{id}(u_{k-1}^2 + \epsilon_{id})$ is time-varying and $c_2 \in \mathbb{R}_{>0}$ is implicitly supposed to be finite, as the closed-loop system is stable and with unitary gain.

The closed-loop estimated model can be finally rewritten in the state-space form:

$$x_{k+1} = a_k x_k + b_k u_k \tag{13a}$$
$$y_k = c_k x_k \tag{13b}$$

where, according to (10), $a_k = \hat{\theta}_k$, $b_k = 1 - a_k$, and $c_k = 1$. Algorithm 1 collects the RLS$_{\text{FO}}$ steps.

Proposition 2.1: The total amount of FLoating-point OPerations (FLOPs) required by Algorithm 1 is 15*, and the memory allocation needed to run is* 12 *floats, including input data.*

The amount of FLOPs, defined as the number of atomic arithmetic operations on floating-point numbers, is used as a measure of computational intensity. Conventionally, addition, subtraction, multiplication, and division each count as 1 FLOP [37], while memory assignments and logical comparisons are

excluded, contributing 0 FLOPs. In contrast, the number of floating-point variables measures memory usage.

III. Optimal Control Problem and Schur Complement Solution

Let us assume that the closed-loop system dynamics is modeled by (13). At each sampling time k, we want to solve the following optimization problem

$$\min_{u_i,x_j} \ \frac{1}{2}\sum_{i=0}^{p-1}\left(\|y_j - r_k\|^2_{w_y} + \|u_i - u_{i-1}\|^2_{w_\delta}\right) \tag{14a}$$

$$\text{s.t.} \ \ x_j = a_k x_i + b_k u_i \tag{14b}$$

$$y_j = c_k x_j \tag{14c}$$

$$i = 0,\dots,p-1, \quad j = i+1$$

where $w_y, w_\delta \in \mathbb{R}_{>0}$ are weights on output tracking error and control increment, respectively, $x_i \equiv x_{k+i|k}$ denotes the prediction of x at time $k+i$ based on information up to time k, and the same applies to y_i and u_i. The prediction horizon is denoted by $p \geq 1$, $u_{k-1|k} = u_{k-1}$ is the input applied at the plant at time $k-1$, and $x_{k|k} = \hat{x}_k$ with $\hat{x}_k$ the state estimation at time k. The goal is to efficiently compute the optimal sequence of pairs (u_i^*, x_j^*), $i = 0,\dots,p-1$, $j = 1,\dots,p$, that minimizes problem (14). More specifically, we are interested in u_0^*, which is the actual input that is eventually applied to the system. For this purpose, in the following we assume $c_k \neq 0$; otherwise, no information about x_k can be obtained from y_k. However, this assumption is automatically satisfied, as the method detailed in Section II sets $c_k = 1$.

Let us define the input increment $\delta_k = u_k - u_{k-1}$, and let $z = [\delta_k \ x_{k+1} \ \dots \ \delta_{k+p-1} \ x_{k+p}]^\mathsf{T}$ be the vector of optimization variables at time k collecting both the input increments and the corresponding states defined over a prediction of p steps. We have that $z \in \mathbb{R}^m$, with $m = 2p$. Problem (14) can be rewritten as the equivalent parametric QP problem

$$z^* = \arg\min_{z\in\mathbb{R}^m} \ \frac{1}{2} z^\mathsf{T} H z + z^\mathsf{T} F\theta \tag{15a}$$

$$\text{s.t.} \qquad Ez = e \tag{15b}$$

where $H \in \mathbb{D}^m_{++}$, $F \in \mathbb{R}^{m\times p}$, $\theta = r_k^{\{p\}}$ and such that

$$H = \text{diag}(\begin{bmatrix} w_\delta^2 & c_k^2 w_y^2 & \dots & w_\delta^2 & c_k^2 w_y^2 \end{bmatrix}^\mathsf{T}) \tag{16a}$$

$$F = -\begin{bmatrix} 0 & c_k w_y^2 & 0 & \dots & \dots & 0 \\ \vdots & \ddots & 0 & c_k w_y^2 & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & & \dots & & 0 & c_k w_y^2 \end{bmatrix}^\mathsf{T} \tag{16b}$$

The equality constraints (15b) with $E \in \mathbb{R}^{p\times m}$ and $e \in \mathbb{R}^p$ embed the dynamics of model (13) over p prediction steps, and have the following form

$$\begin{bmatrix} b_k & -1 & 0 & & \dots & & 0 \\ \vdots & a_k & b_k & -1 & 0 & \dots & \vdots \\ \vdots & & \vdots & & & & 0 \\ b_k & 0 & b_k & \dots & a_k & b_k & -1 \end{bmatrix} z_k = \begin{bmatrix} -a_k x_k - b_k u_{k-1} \\ -b_k u_{k-1} \\ \vdots \\ -b_k u_{k-1} \end{bmatrix} \tag{17}$$

We present here the generic steps of a *direct* method to solve the QP problem (15). Such a method will be then numerically analyzed in the next sections to ultimately derive a throughput and memory efficient algorithm that exploits the structure of the QP matrices.

Theorem 3.1: Let us define $D = \text{diag}(\ell^{\{p\}})$, with $\ell = \begin{bmatrix} w_\delta^{-1} & (c_k w_y)^{-1} \end{bmatrix}^\mathsf{T}$ and $D \in \mathbb{D}^m_{++}$, $h = F\theta$ and $\bar{h} = DDh$. Let λ be the vector of Lagrange multipliers for QP problem (15), and

$$DE^\mathsf{T} = QR, \quad Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}, \quad R = \begin{bmatrix} \bar{R} \\ 0_p \end{bmatrix} \tag{18}$$

where Q is an orthogonal matrix, $Q^\mathsf{T}Q = I_m$, $Q_1 \in \mathbb{R}^{p\times p}$, $Q_2 \in \mathbb{R}^{p\times p}$, and $\bar{R} \in \mathbb{R}^{p\times p}$ is an upper triangular matrix. Then the optimal pair (z^*, λ^*) can be derived by solving the system of equations

$$\bar{R}^\mathsf{T}\bar{R}\lambda^* = E\bar{h} + e \tag{19a}$$

$$z^* = DD(E^\mathsf{T}\lambda^* - h) \tag{19b}$$

Proof: We want to show that QP problem (15) is minimized by (19b). z^* is the optimal solution if there exists a collection of Lagrange multipliers λ^* such that the Karush–Kuhn–Tucker (KKT) system of equations

$$\underbrace{\begin{bmatrix} H & E^\mathsf{T} \\ E & 0 \end{bmatrix}}_{K} \begin{bmatrix} z^* \\ -\lambda^* \end{bmatrix} = \begin{bmatrix} -h \\ e \end{bmatrix} \tag{20}$$

is satisfied, with K the *KKT matrix*. Given $H \in \mathbb{D}^m_{++}$ and $Ez^* = e$, we can rewrite the first equation in (20) as

$$EH^{-1}(Hz^* - E^\mathsf{T}\lambda^*) = e - EH^{-1}E^\mathsf{T}\lambda^* = -EH^{-1}h \tag{21}$$

Given the model dynamics constraints (17), we also have that $\text{rank}(DE^\mathsf{T}) = p$ being E full-row rank, which means the QR factorization (18) always exists with $\bar{R}$ nonsingular. From the definition of D and H, given (18), we can write $EH^{-1}E^\mathsf{T} = (ED)(ED)^\mathsf{T} = \bar{R}^\mathsf{T}\bar{R}$, which proves (19a). Once λ^* is obtained, it is then easy to prove (19b) by simply solving the first KKT equation with respect to z^*. ■

The z^* computed in (19) is the unique optimizer that minimizes problem (15). The optimal input u_k^* applied to the system can be then computed as $u_k^* = u_{k-1} + \delta_k^* = u_{k-1} + z_0^*$.

Remark 3.1: Computing (z^*, λ^*) from system (19) can be seen as solving *directly* the KKT system with a *Schur complement method* based on QR-factorization and pivoting on H. Indeed, $K/H = 0 - EH^{-1}E^\mathsf{T} = -\bar{R}^\mathsf{T}\bar{R}$ and we can analogously derive (19) by a block backsolve on the following system

$$\begin{bmatrix} H & E^\mathsf{T} \\ 0 & K/H \end{bmatrix} \begin{bmatrix} z^* \\ -\lambda^* \end{bmatrix} = \begin{bmatrix} -h \\ e \end{bmatrix} \tag{22}$$

■

The Schur method is rarely applied in practice, because it requires the nonsingularity of H, and K/H is generically hard to compute, unless H^{-1} is known, or the numbers of rows of E is small. However, the structure of the present optimization problem, as well as the optimizer reformulation in (δ_i^*, x_j^*),

$i = 0, \dots, p-1$, $j = 1, \dots, p$ make H^{-1} easy to compute. Additionally, in the next section, we prove the existence of a tailored QR factorization for Step (18), which significantly reduces the computational and memory requirements of solving the linear system in λ^*. With these premises, solving (15) with a method based on Schur complement turns out to be very effective. It is worth noticing that any ordering of z preserves the favorable conditions for computing K/H, however as it will be clear in the following, the specific definition of z here adopted aims at reducing the computations necessary for the factorization in (18).

IV. Real-Time Optimization Algorithm

In Theorem 3.1 we have outlined the steps of an efficient optimization algorithm for problem (14). The throughput for computing z^* with (19) is dominated by the QR factorization of $DE^\intercal$. Let us recall that E is full row rank and Q does not need to be formed explicitly. A standard Q-less factorization of a matrix of dimension $2p \times p$, without rank-revealing properties, has a complexity of $\mathcal{O}(\frac{10}{3}p^3)$, if we assume the most common method based on Householder reflections [37]. In the next, we present a new algorithm that drastically reduces such complexity. Furthermore, we introduce implementation measures aimed at increasing the throughput of the overall optimization algorithm, while keeping a minimal memory footprint.

A. QR factorization

The objective of this section is to present a new factorization algorithm tailored for computing $\bar{R}$ under the assumption of the sparsity pattern in (17). We based our method on Givens rotations, which are efficient at zeroing specific elements below the diagonal of a given matrix [38]. Such selective column annihilation is usually preferred over Householder reflectors when dealing with sparse matrices. Let $v = \begin{bmatrix} v_1 & v_2 \end{bmatrix}^\intercal$ be a vector with $\|v\|_2 = \rho > 0$, then a Givens rotation is the matrix $T \in \mathbb{R}^{2\times 2}$ such that

$$\underbrace{\begin{bmatrix} \gamma & \sigma \\ -\sigma & \gamma \end{bmatrix}}_{T} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \rho \\ 0 \end{bmatrix} \tag{23}$$

where $\rho = h(v_1, v_2)$, $\gamma = \rho^{-1} v_1$, $\sigma = \rho^{-1} v_2$ and h is the *hypot* function that computes $\sqrt{v_1^2 + v_2^2}$ avoiding underflow or overflow. For the following theorem, we introduced the notation of the matrix-constructor $S(s_1, s_2, s_3)$, with $S \in \mathbb{R}^{p\times m}$ a matrix solely defined by the scalars s_1, s_2, $s_3 \in \mathbb{R}$, and with the following structure

$$S(s_1, s_2, s_3) = \begin{bmatrix} s_1 & s_2 & 0 & & \dots & & 0 \\ \vdots & s_3 & s_1 & s_2 & 0 & \dots & \vdots \\ \vdots & & \vdots & & & & 0 \\ s_1 & 0 & s_1 & \dots & s_3 & s_1 & s_2 \end{bmatrix} \tag{24}$$

Theorem 4.1: Let $a_k, b_k, c_k \in \mathbb{R}$ be the coefficients of the linear dynamics (13), and thus defining matrix $E \in \mathbb{R}^{p\times m}$ of the equality constraints (17). Let $w_y, w_\delta \in \mathbb{R}$ be the weights of the objective function (14a), defining the diagonal entries of $D \in \mathbb{D}^m_{++}$, with $DD = H^{-1}$. Then, the upper triangular matrix $\bar{R} \in \mathbb{R}^{p\times p}$ of the factorization $DE^\intercal = Q \begin{bmatrix} \bar{R} \\ 0_p \end{bmatrix}$ can be formed by applying Algorithm 2 (that we address as QR_{FO}) to the matrix $S(b_k w_\delta^{-1}, -(c_k w_y)^{-1}, a_k (c_k w_y)^{-1})^\intercal$.

Proof: We first show that there exists a QR decomposition of S for which the $\bar{R}$ computed by QR_{FO} is the square upper triangular matrix. We proceed by induction. Let us introduce matrix $G \in \mathbb{R}^{2q\times q}$, a lower staircase matrix such that if we define $\tilde{G}_i \in \mathbb{R}^{2\times q}$, $i = 1, \dots, q$, as:

$$\tilde{G}_j = \begin{bmatrix} 0^{\{j-1\}} & 0^{\{j-1\}} \\ g_1 & g_2 \\ g_3^{\{q-j\}} & g_4 \\ & g_5^{\{q-j-1\}} \end{bmatrix}^\intercal, \ j = 1, \dots, q-1 \tag{25a}$$

$$\tilde{G}_q = \begin{bmatrix} 0^{\{q-1\}} & 0^{\{q-1\}} \\ g_1 & g_2 \end{bmatrix}^\intercal \tag{25b}$$

then $G = \begin{bmatrix} \tilde{G}_1^\intercal & \dots & \tilde{G}_q^\intercal \end{bmatrix}^\intercal$ holds true. We indicate by $\mathbb{G}^q$ the set of real matrices of dimension $2q \times q$ that have the structure (25). Let us also introduce $\Gamma \in \mathbb{R}^{2q\times q}$ computed as:

$$\Gamma = \begin{bmatrix} (T\tilde{G}_1)^\intercal & \dots & (T\tilde{G}_q)^\intercal \end{bmatrix} \tag{26}$$

where T is the Givens rotation corresponding to the *hypot* function $h(g_1, g_2)$. We then have that $i)$ $\Gamma_{1:q,1}$ has a single non-zero element, and that is on the diagonal of Γ, $ii)$ $\Gamma_{q,1:q}$ is a zero row, $iii)$ $\tilde{\Gamma} = \Gamma_{2:2q-1,2:q} \in \mathbb{G}^{(q-1)}$. Properties $i)$ and $ii)$ directly follow from (25) and the definition of T, as $T \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} \rho \\ 0 \end{bmatrix}$, $\rho > 0$. The sub-matrix $\tilde{\Gamma}$ can be rewritten as (25) by assuming a column space of $q-1$ and an update function for its entries

$$\begin{bmatrix} g_1 & g_3 \\ g_4 & g_5 \end{bmatrix} \leftarrow T \begin{bmatrix} g_4 & g_3 \\ g_3 & g_5 \end{bmatrix} \tag{27a}$$

$$g_2 \leftarrow \rho \tag{27b}$$

If we define $g_1 = g_3 = s_1$, $g_2 = s_2$, $g_4 = s_3$ and $g_5 = 0$ then $S^\intercal \in \mathbb{G}^p$ clearly applies. Therefore we can triangularize $S^\intercal$ by first initializing $R = S^\intercal$, and then recursively update

$$R_{j:2p-j+1,j:p} \leftarrow \Gamma(R_{j:2p-j+1,j:p}), \ \ j = 1, \dots, p \tag{28}$$

where given $A \in \mathbb{G}^q$, $q \leq p$ we denote by $\Gamma(A)$ the operation of computing Γ through (26). Lastly, $\bar{R}$ is obtained by collecting the first p rows of A. The update step (28) encloses the main source of computational efficiency of the proposed method, which is the need to compute only p different T rotations. That is in contrast with standard methods, which perform $2p^2 - \sum_{i=2}^{p+1} i$ rotations if S is assumed dense by the factorization algorithm, or $\sum_{i=1}^{p} i$ if we assume a sparse algorithm with exact knowledge of the sparsity pattern. The QR_{FO} algorithm is an efficient implementation of the recursive update (28), derived by exploiting the properties of successive applications of operator $\Gamma(A_t)$, $A_t \in \mathbb{G}^{p-t}$, $t = 0, \dots, p-1$, and particularized for the case where $A_0 = S$. In practice, it is indeed unnecessary to compute (26) explicitly. Given the initial values for g_j, $j = 1, \dots, 5$, each update step i requires the computation of T, the update of g_j through (27), and the assignment of $\bar{R}_{i,i:p}$.

We have shown that QR_{FO} is a computational efficient way of computing a Q-less factorization of a generic $G \in \mathbb{G}^q$, and that $S^\intercal \in \mathbb{G}^p$. In order to prove Theorem 4.1 it is left to show that $DE^\intercal$ is a specific realization of $S^\intercal$. This easily follow from the fact that $E = S(b_k, -1, a_k)$, and that D is block-diagonal, with 2×2 blocks $\begin{bmatrix} w_\delta^{-1} & 0 \\ 0 & (c_k w_y)^{-1} \end{bmatrix}$. Pre-multiplying $E^\intercal$ by D does not alter its sparsity and structure, and we get that $DE^\intercal = S(b_k w_\delta^{-1}, -(c_k w_y)^{-1}, a_k (c_k w_y)^{-1})^\intercal$. ■

Proposition 4.1: Assuming that the hypot *function, which is required to compute γ, σ and ρ, as defined in (23), incurs a computation cost of 4 FLOPs, the total amount of FLOPs required by QR_{FO} is*

$$\psi(\text{QR}_{\text{FO}}) = 4p + 9(p-1) + \max(0, 6(p-2)). \tag{29}$$

Proposition 4.1 highlights a computational complexity of $\mathcal{O}(21p)$ for QR_{FO} when $p \geq 2$, which is significantly lower than the $\mathcal{O}(\frac{10}{3}p^3)$ one would get with a standard Q-less decomposition based on Householder reflections. Similarly to throughput, memory occupation has a key role in embedded applications. QR decomposition algorithms with a concern on memory occupation, most often base their machinery on the assumption that given a matrix $A \in \mathbb{R}^{2p \times p}$ to factorize, the upper triangular matrix R is iteratively built on the memory storage reserved to A. This translates into the need to store at least $2p^2$ floating point entries. On the contrary, from Theorem 4.1 we get that E is not stored explicitly, rather we just form $\bar{R}$ entries in place, directly starting from the knowledge of model parameters a_k, b_k and c_k, and cost parameters w_δ and w_y. As it can easily derived from Theorem 4.1 steps, we also note that $\bar{R}$ has the form

$$\bar{R} = \begin{bmatrix} \rho_1 & r_1 & r_2 & \dots & \dots & r_2 \\ 0 & \rho_2 & r_3 & r_4 & \dots & r_4 \\ \vdots & & \vdots & & & \vdots \\ 0 & \dots & \dots & 0 & \rho_{p-1} & r_{2p-3} \\ 0 & \dots & \dots & \dots & 0 & \rho_p \end{bmatrix} \tag{30}$$

with $r_i \in \mathbb{R}$, $i = 1, \dots, 2p-3$, and ρ_j, $j = 1, \dots, p$ the nonzero element of the rotations (23).

Proposition 4.2: Let us assume a compressed storage for $\bar{R}$, where only r_i, and ρ_j, entries are stored, $i = 1, \dots, 2p-3$, $j = 1, \dots, p$. Then the memory allocation needed to run QR_{FO} is $3(p-1)$ floats.

We finally note that compressing the storage of $\bar{R}$ requires trivial modifications to QR_{FO}, which therefore are not detailed here.

Example 4.1: Consider a matrix $S \in \mathbb{R}^{p \times m}$ having the structure defined in (24), and the QR factorization $Q \begin{bmatrix} \bar{R}^\intercal & 0_p \end{bmatrix}^\intercal = S^\intercal$. From Proposition 4.1 we know that computing $\bar{R}$ with QR_{FO} is extremely efficient, if compared to a standard Q-less decomposition algorithm. However, in both desktop and embedded applications, it is frequent to rely on linear algebra libraries, which provide heavily optimized math routines, often CPU-specific. Such algorithms offer superior performance when compared to naive implementation of state-of-the-art routines. For this reason, in this example, we show a `C++` benchmark where QR_{FO} is compared to

Algorithm 2 QR_{FO}: efficient QR-decomposition of $S^\intercal$

Input: Scalars $s_1, s_2, s_3 \in \mathbb{R}$ defining matrix $S(s_1, s_2, s_3)$ as in (24), prediction horizon $p \in \mathbb{Z}_{\geq 1}$.

$\bar{R} \leftarrow 0_p$, $\rho \leftarrow s_1$, $s_4 \leftarrow 0$;
for $i = 1, \dots, p$ **do**
  $R_{i,i} \leftarrow h(\rho, s_2)$;
  if $i < p$ **then**
    $\gamma \leftarrow \bar{R}_{i,i}^{-1} \rho$;
    $\sigma \leftarrow \bar{R}_{i,i}^{-1} s_2$;
    $\bar{R}_{i,i+1} \leftarrow \gamma s_1 - \sigma s_3$;
    $\rho \leftarrow \sigma s_1 + \gamma s_3$;
    if $i < p-1$ **then**
      $\bar{R}_{i,i+2:p} \leftarrow \left[\gamma s_1 - \sigma s_4\right]^{\{p-i-1\}}$;
      $s_1 \leftarrow \sigma s_1 + \gamma s_4$;
      $s_3 \leftarrow \bar{R}_{i,i+1}$;
      $s_4 \leftarrow \bar{R}_{i,i+2}$;
    end if
    $s_2 \leftarrow \bar{R}_{i,i}$;
  end if
end for

Output: Upper triangular matrix $\bar{R}$ that admits the existence of a matrix Q, with $QQ^\intercal = I_{2p}$, such that $Q \begin{bmatrix} \bar{R}^\intercal & 0_p \end{bmatrix}^\intercal = S(s_1, s_2, s_3)^\intercal$.

two well-known algebra suites, *LAPACK* and *Eigen*. The test is performed on a Intel® Core™ i7-8750H @ 2.20 GHz, and consists in factorizing matrices of different dimensions, namely $p \in \{4i \,|\, i = 1, \dots, 10\}$. For each p value, we perform 10^6 factorizations and acquire the average elapsed time through the high resolution clock from *chrono* library. Fig. 2 collects the throughput results obtained with the different algorithms. Please note that for the *Eigen* library we tested both the dense and the sparse decomposition. The latter however implements a column-pivoting strategy, and therefore even if S is sufficiently sparse, the dense routine is outperforming on small-size matrices. The results confirm that QR_{FO} is vastly faster than optimized algebra libraries, with a reduction of computational complexity that is up to 40 times with respect to the best alternative. We finally note that for the example, we have assumed that S is already available in memory. However, allocating all the non-zero entries of S (which is done at each time step in time-varying applications) adds complexity. Such memory allocation time is absent in QR_{FO} given that S is never formed explicitly, and only the three scalars uniquely defining the matrix are required.

B. Computation of the Optimal Control Law

We recall that the implementation of the MPC control requires us to solve the system of equations (19). In the specific, given that our goal is to implement just the first move of the optimal input sequence, (19a) has to be fully solved in λ^*, whereas for (19b) our interest is limited to $z_0^* = \delta_k^*$. In the following, we derive insights on the quantities involved for solving (19) to derive a memory and computationally efficient method to compute z_0^*. The computational steps are collected

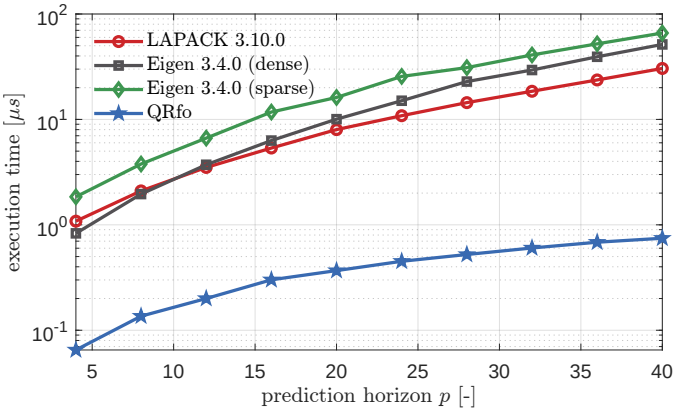


Fig. 2: Execution time for different routines computing the QR decomposition of $S \in \mathbb{R}^{p\times m}$, with S having the structure in (24). For each p we collect the average time over 10^6 algorithm executions.

Algorithm 3 MPC_{FO}: computes the optimal move at time k

Input: Scalars a_k, b_k, $c_k \in \mathbb{R}$ defining the dynamics (13), weights w_δ, $w_y \in \mathbb{R}_{>0}$ defining cost (14a), state x_k, input u_{k-1}, set-point r_k and prediction horizon $p \in \mathbb{Z}_{\geq 1}$.

1: $t \leftarrow \bar{0}_p$, $\lambda^* \leftarrow \bar{0}_p$;
2: $\bar{R} \leftarrow \text{QR}_{\text{FO}}(b_k w_\delta^{-1}, -(c_k w_y)^{-1}, a_k(c_k w_y)^{-1}, p)$;
3: $\ell \leftarrow \begin{bmatrix} \frac{r_k}{c_k} - a_k x_k - b_k u_{k-1} \\ [\frac{r_k}{c_k}(1-a_k) - b_k u_{k-1}]^{\{p-1\}} \end{bmatrix}$;
4: $t_1 \leftarrow \frac{\phi_1}{\bar{R}_{1,1}}$, $t_i \leftarrow \frac{1}{\bar{R}_{i,i}}\left(\phi_i - \bar{R}_{i-1,i} t_{i-1}\right)$, $i = 2, \ldots, p$;
5: $\lambda_j^* \leftarrow \frac{1}{\bar{R}_{j,j}}\left(t_j - \sum_{v=j+1}^{p} \bar{R}_{j,v}\lambda_v\right)$, $j = p, \ldots, 1$;
6: $\delta^* \leftarrow \frac{b_k}{w_\delta^2}\sum_{i=1}^{p} \lambda_i^*$;

Output: The optimal control move $u_k^* = u_{k-1} + \delta^*$.

in Algorithm 3, which we call MPC_{FO}. The main source of complexity, that is the factorization of $\bar{R}$ has been addressed in Section IV-A. Therefore, we first focus on the computation of the right-hand side of (19a). We have that $E\bar{h} = EH^{-1}h$. By combining (16a) and (17) it directly follows that

$$EH^{-1} = S(b_k w_\delta^{-2}, -(c_k w_y)^{-2}, a_k(c_k w_y)^{-2}) \tag{31}$$

Moreover, given $h = F\theta$ it follows that

$$h = \begin{bmatrix} 0 \\ -c_k w_y^2 r_k \end{bmatrix}^{\{p\}} \tag{32}$$

and, with simple matrix computations, we obtain the compressed representation

$$E\bar{h} = \begin{bmatrix} \frac{r_k}{c_k} \\ [\frac{r_k}{c_k}(1-a_k)]^{\{p-1\}} \end{bmatrix} \tag{33}$$

Vector e can be written in the same compressed form, and therefore the right-hand side of (19a), denoted as ℓ, can be computed efficiently, as highlighted in step 2 of MPC_{FO}. Computing λ_k^* requires the solution of two linear systems, with a forward and a backward substitution step on matrix $\bar{R}$. Having $\bar{R} \in \mathbb{R}^{p\times p}$ the special structure in (30), we can derive a forward solve algorithm that reduces the standard $\mathcal{O}(p^2)$ complexity. Let $\ell \in \mathbb{R}^p$ be a vector in the form $\ell = \begin{bmatrix} \ell_1 \\ \ell_2^{\{p-1\}} \end{bmatrix}$, $t \in \mathbb{R}^p$, we introduce the iterative operator ϕ_i, with $i = 1, \ldots, p$, and defined as

$$\phi_i = \begin{cases} \ell_1, & \text{if } i = 1 \\ \ell_2, & \text{if } i = 2 \\ \phi_{i-1} - \bar{R}_{i-2,i} t_{i-2}, & \text{if } i > 2 \end{cases} \tag{34}$$

Then, the solution of the linear system $\bar{R}^\intercal t = \ell$ can be efficiently computed as

$$t_1 = \frac{\phi_1}{\bar{R}_{1,1}} \tag{35a}$$

$$t_i = \frac{1}{\bar{R}_{i,i}}\left(\phi_i - \bar{R}_{i-1,i} t_{i-1}\right), \quad i = 2, \ldots, p \tag{35b}$$

Performing a forward substitution as in (35) amounts at $3p-2$ FLOPs. The backward solve that solves $\bar{R}\lambda_k^* = t$ cannot exploit any structure of $\bar{R}$, and therefore its complexity is $\mathcal{O}(p^2)$. Finally, given that $L_{1:p,1} = \left[w_\delta^{-2} b_k\right]^{\{p\}}$ with $L = EH^{-1}$, and that $h_0 = 0$ retrieving z_0^* from λ_k^* amounts at computing $z_0^* = w_\delta^{-2} b_k \sum_{i=1}^{p} \lambda_i^*$, which are $p+2$ FLOPs.

Proposition 4.3: In practice MPC_{FO} *does not need to store* ℓ, t *and* λ_k^* *separately. Indeed the operations in which such quantities are involved make evident the possibility to store a single vector of* p *elements. Given this assumption, and the result from Proposition 4.2, the total memory occupation for running* MPC_{FO} *is* $4p-3$ *floats.*

Please note that the memory requirements stated in Proposition 4.3 disregard the data defining problem (14). Their impact is minimal, as they consist of just 6 floats. Furthermore, these floats would be common to any method selected for solving the optimal control problem.

Proposition 4.4: Given the computational analysis here detailed, regarding the different steps for solving (19), and the results from Proposition 4.1 we can conclude that, in the non-trivial cases for which $p \geq 2$*, the total amount of FLOPs required by* MPC_{FO} *is:*

$$\psi(\text{MPC}_{\text{FO}}) = p^2 + 23p - 5 \tag{36}$$

Throughout Section III and IV we have assumed that w_δ and w_y are constants, however the algorithm here proposed is not impacted at all if we assume the weights are time-varying, that is we replace them with $w_{\delta,k}$ and $w_{y,k}$. This can be helpful when impactful nonlinearities arise in the closed-loop model, or simply if the control objective changes for different conditions.

Example 4.2: An efficient and widely adopted strategy to solve QP problem (15) is *standard condensing* which is a specific variable elimination, consisting in replacing the predicted states x_k with the corresponding predictions

$$x_{k+j|k} = a_k^j x_k + \sum_{i=0}^{j-1} a_k^i b_k u_{k+i}, \quad j = 1, \ldots, p \tag{37}$$

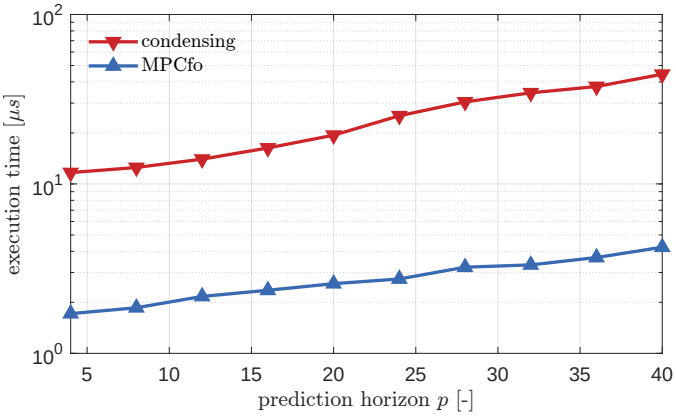


Fig. 3: Execution time comparison for computing the optimal control move with the tailored condensing routine in (40) and with MPC$_{\text{FO}}$. For each p we collect the average time over 10^6 executions.

and then solving a reduced unconstrained QP in $\tilde{z}_k \in \mathbb{R}^p$, $\tilde{z}_k = \begin{bmatrix}\delta_k & \dots & \delta_{k+p-1}\end{bmatrix}$. It is worth noticing that the particular structure of (15) can be exploited to reduce the computational load also when approaching the problem with standard condensing. We are therefore interested in first deriving a computationally efficient condensed approach, and then comparing its execution speed with respect to MPC$_{\text{FO}}$, to show the superior performance of the latter. Let us define $\gamma \in \mathbb{R}^p$, $\rho \in \mathbb{R}^p$, such that $\gamma_i = c_k a_k^i$, $\rho_i = \sum_{j=0}^{i-1} a_k^j b_k$, $i = 1, \dots, p$, and $V \in \mathbb{R}^{p\times p}$, $V = \begin{bmatrix}v_1, \dots, v_p\end{bmatrix}$, with

$$v_1 = \rho \tag{38a}$$

$$v_j = \begin{bmatrix} 0^{\{j-1\}} \\ \rho_{1:p-i+1} \end{bmatrix}, \quad j = 2, \dots, p \tag{38b}$$

then it is possible to cast problem (15) into the reduced QP

$$\tilde{z}^* = \arg\min_{\tilde{z}\in\mathbb{R}^p} \frac{1}{2}\tilde{z}^\intercal \tilde{H}\tilde{z} + \tilde{z}^\intercal \tilde{h} \tag{39}$$

where, given $V_y = V^\intercal \operatorname{diag}((\omega_y^2)^{\{p\}})$, we have

$$\tilde{H} = \operatorname{diag}((\omega_\delta^2)^{\{p\}}) + V_y V \tag{40a}$$

$$\tilde{h} = V_y(\rho u_k + \gamma x_k - r_k^{\{p\}}) \tag{40b}$$

$$\tilde{z}^* = \tilde{H}^{-1}\tilde{h} \tag{40c}$$

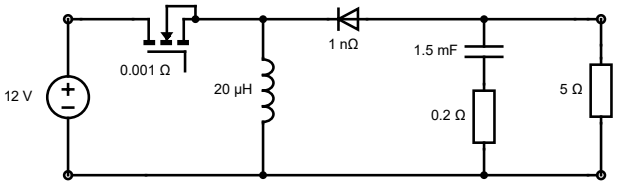
with $\tilde{H}$ positive definite by construction, and $u_k^* = u_{k-1} + \tilde{z}_1^*$. In Fig. 2 we compare the throughput for computing the component δ_k of the z^* optimizing (15), through MPC$_{\text{FO}}$ and concerning the well-known condensing, even efficiently implemented according to (40). The hardware setup is the same as Example 4.1, and here as well we collect the average time over 10^6 executions for each p. The results highlight the superior performance of the here proposed MPC$_{\text{FO}}$, showing an execution time improvement of up to tenfold.

V. Simulation Results

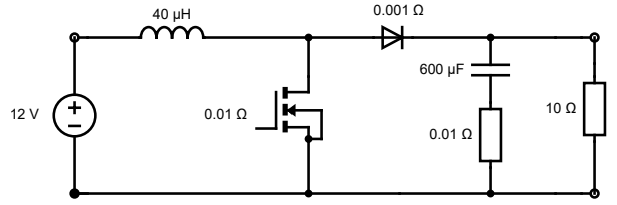
The algorithm is first tested in simulation on three DC-DC power converters: a buck-boost converter, a boost converter, and a synchronous buck converter. In the following, we assume the low-level controller to be a PI controller. This assumption does not impose a limitation, as any other control law could be used in its place, and the MPC$_{\text{FO}}$ could still be employed in such a closed-loop system to enhance transient behavior or to enable reference preview and fault tolerant control. The performance of the MPC$_{\text{FO}}$ is then compared to that of the baseline PI controller.

The devices are modeled in Simulink® using Simscape™. The simulation time step is set to $T_s = 0.05\,\mu\text{s}$, in order to model the actual switching of the transistors. The devices are controlled in Voltage Mode Control (VMC) and the sample time for the inner PI control loop is kept to $T_{sc} = 5\,\mu\text{s}$. The identification is active only when the reference is sufficiently exciting, according to Section II-C. To show that MPC$_{\text{FO}}$ does not need any tuning depending on the chosen DC-DC converter, the parameters for MPC$_{\text{FO}}$ are the same for all devices, and they are collected in Table I. Increasing the ratio between w_y and w_δ reduces the rise time but increases the overshoot, and vice versa.

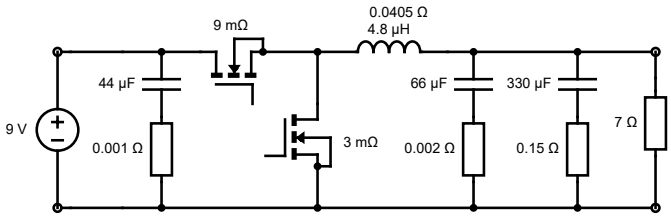
Each converter undergoes testing in two distinct scenarios. The first scenario involves a *switching-on* transient, which highlights the enhanced dynamics achieved by MPC$_{\text{FO}}$. The second is a collection of *step responses* that aids in recognizing the algorithm's adaptation ability. The following subsections analyze the tests on the different converters, and for a comprehensive summary, please refer to Table II for the first scenario, and Table III for the second.



(a) Buck-boost converter.



(b) Boost converter.



(c) Synchronous buck converter.

Fig. 4: Schematics of the three converters on which MPC$_{\text{FO}}$ is being tested in simulation.

TABLE I: MPC_{FO} tuning parameters.

Parameter	Value
w_y	1
w_δ	0.5
p	6
λ	0.9
σ_{id}	0.000625
ϵ_{id}	0.04

TABLE II: Simulation results in the *switching-on* scenario.

	IAE [V·ms]	**Peak current** [A]	**Rise time** [ms]	**Overshoot** [%]
Buck-Boost				
PI	93.7	3.899	11.025	0.0
MPC_{FO}	37.4	3.918	4.265	0.2
Boost				
PI	30.3	2.547	0.360	17.0
MPC_{FO}	25.2	2.262	0.340	3.9
Buck				
PI	1.87	4.547	1.070	0.0
MPC_{FO}	1.08	5.041	0.415	0.9

A. Buck-boost converter

The first device to be tested is a buck-boost converter. The device parameters are detailed in Fig. 4a. The parameters of the PI regulator are $K_p = 0.02$ and $K_i = 7$, where the calculated duty cycle is limited in $[0, 0.85]$ due to saturation constraints and clamping is implemented to avoid windup. The sample time for both RLS_{FO} and MPC_{FO} is set to $500\,\mu\text{s}$.

In Fig. 5a, the solid blue line represents the output voltage, the dotted yellow line is the original reference, and the dashed orange line is the reference modified by MPC_{FO}. For comparison, the output voltage without MPC_{FO} is represented by a solid black line. The original closed-loop system is overdamped: the voltage without MPC_{FO} reaches the reference after approximately 25 ms, and the Integral of Absolute Error (IAE) is 93.7. Introducing the MPC_{FO}, the reference is automatically increased by the MPC_{FO} to decrease the rise time (-61.3%): the IAE drops to 37.4, thus providing a -60.0% improvement. The overshoot is absent using the baseline PI controller, while it slightly increases to 0.2% with MPC_{FO}. Accordingly, the peak inductor current slightly increases by 0.5%. Fig. 5a also reports the outcome of the RLS_{FO}, where parameters a and b are represented by a solid blue line and a dotted red line, respectively, while the shaded area highlights the activation of parameter identification. The parameters converge in a few steps and, as soon as the system is in steady state, the identification is turned off as (12) does not hold. To highlight the adaptation capabilities, we also report in Fig. 6a a more extensive simulation, showing the response to a reference composed of subsequent steps. The abrupt changes in the operating point are sufficient to modify the system parameters: the parameter b, identified by the RLS_{FO}, is correlated to the output voltage, showing a clear dependency of the system model on the operating conditions. The IAE, the peak current, and the ranges of the estimated parameters are reported in Table III. The time-varying model can adapt to the nonlinear and/or unmodeled dynamics in the entire operating range: using MPC_{FO} and neglecting the initial transient (switching on) for clarity, the parameter b varies in the range $[0.03,\ 0.38]$ as the operating point changes. The consistency of the estimated parameters supports the validity of the time-varying SISO model for the buck-boost converter.

TABLE III: Simulation results in the *step responses* scenario.

	IAE [V·ms]	**Peak current** [A]	α **param.** [min,max]	β **param.** [min,max]
Buck-Boost				
PI	330	4.324	n.a.	n.a
MPC_{FO}	141	4.367	[0.62, 0.97]	[0.03, 0.38]
Boost				
PI	128	2.639	n.a.	n.a.
MPC_{FO}	77.6	2.326	[0.42,0.86]	[0.14,0.58]
Buck				
PI	10.2	4.842	n.a.	n.a.
MPC_{FO}	6.36	5.532	[0.68,0.78]	[0.22,0.32]

B. Boost converter

The second device to be tested is a boost converter. Its parameters are detailed in Fig. 4b. The PI regulator parameters are $K_p = 0.03$ and $K_i = 16$. Clamping is introduced to deal with duty cycle saturation in $[0, 0.85]$. The sample time for both RLS_{FO} and MPC_{FO} is set to $500\,\mu\text{s}$.

The original closed-loop system overshoots due to an aggressive PI tuning (Fig. 5b): the voltage without MPC_{FO} overcomes the reference before it decreases again. The corresponding IAE is 30.3. Enabling the MPC_{FO}, both the overshoot and the rise time are significantly reduced: the overshoot drops from 17.0% to 3.9%, while the rise time improves by -5.6%. Consequently, the peak current is reduced by -11.2%. The IAE drops from 30.3 to 25.2 (-17.0%). The RLS_{FO} algorithm converges in a few steps.

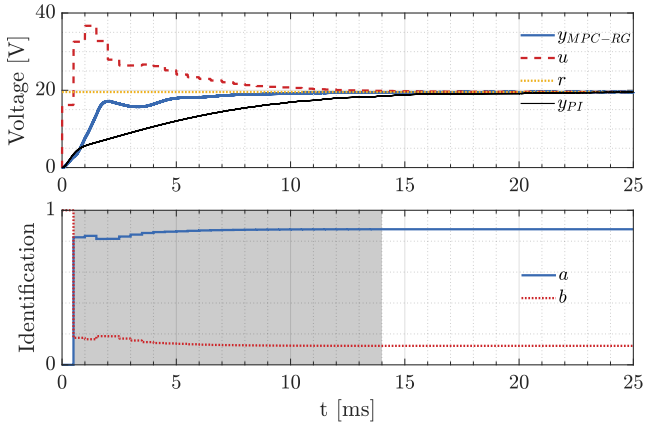
Fig. 6b shows a more extensive simulation. Note that parameter identification also depends on the step height: RLS_{FO} performed on the first step yields parameters inconsistent with those obtained from subsequent smaller steps; therefore, parameter estimation and adaptivity are essential.

C. Buck converter

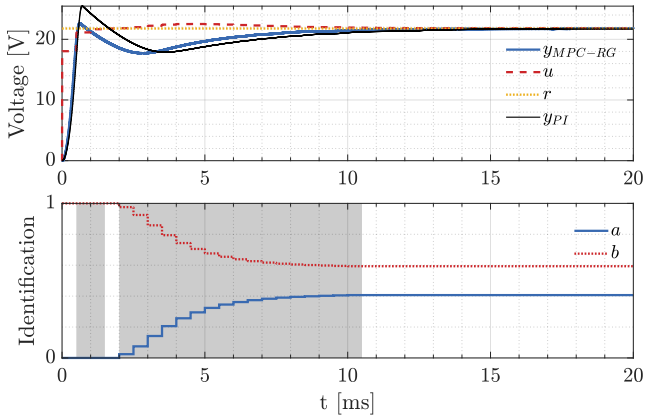
The third device to be tested is a buck converter. The simulated model is a replica of the device being experimentally tested in Section VI. The device parameters are reported in Fig. 4c. The PI regulator parameters are $K_p = 0.1$ and $K_i = 0.008$. Clamping is introduced to deal with duty cycle saturation in $[0, 0.6]$. The device has a faster transient than the previous ones, consequently the sample time for both RLS_{FO} and MPC_{FO} is 5 times smaller, i.e., $100\,\mu\text{s}$.

The original closed-loop system is well-tuned, and the voltage without MPC_{FO} (black line in Fig. 5c) settles after approximately 2 ms. The corresponding IAE is 1.87. Using MPC_{FO}, the rise time is reduced by -61.2% and the IAE reduces to 1.08 (-42.3%). Consequently, the overshoot increases to 0.9% and the peak current increases by 10.9%.

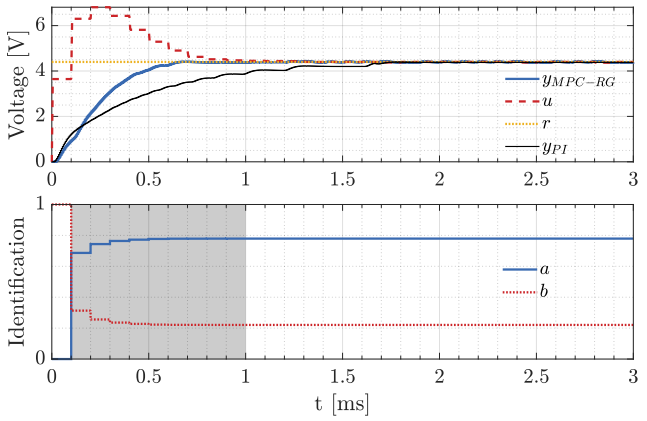
Fig. 6c reports a more extensive simulation. Since the operating range of the buck converter is narrower than that of the boost and buck-boost converters, the parameters obtained using the two methods do not vary significantly, and the output voltage is comparable.



(a) Buck-boost converter.



(b) Boost converter.

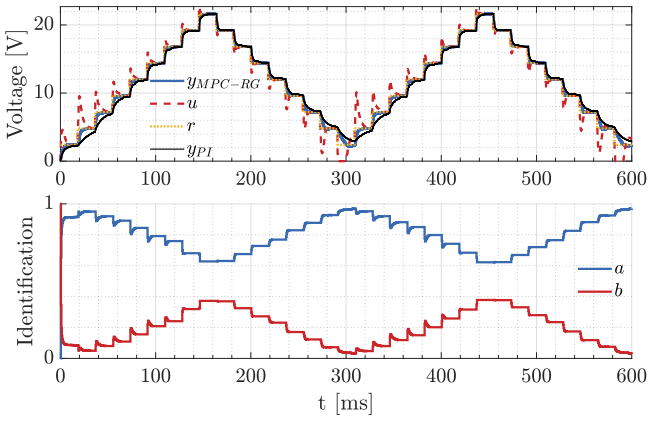


(c) Synchronous buck converter.

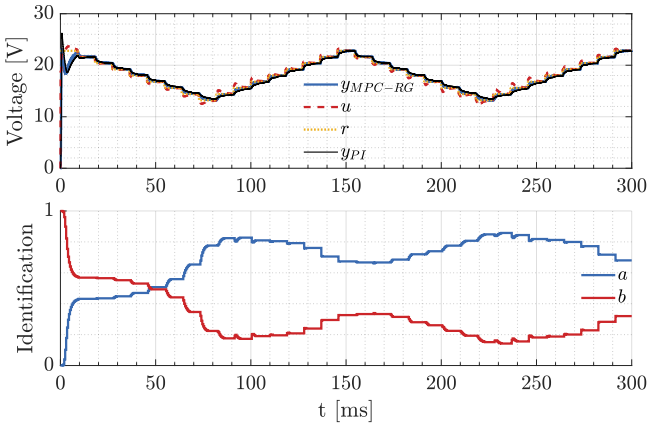
Fig. 5: Power converters: switching on (simulation).

VI. Experimental results

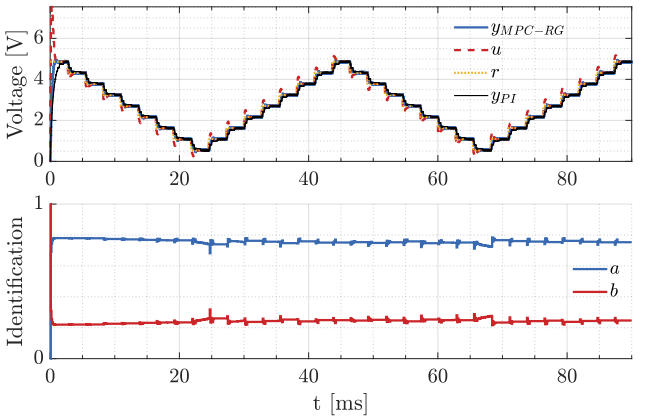
The algorithm is tested on a C2000 Delfino MCU LAUNCHXL-F28379D [39], paired with a Buck Converter BoosterPack BOOSTXL-BUCKCONV [40] by Texas Instruments™. Fig. 7 shows the experimental setup. The F28379D interfaces with the DC-DC synchronous buck converter through the BoosterPack headers. The converter is designed to regulate the nominal output voltage at 2 V, while the nominal input bus voltage is 9 V. The load consists of a static 7 Ω resistance and, possibly, an additional software-controlled 2 Ω switched load. The buck converter parameters



(a) Buck-boost converter.



(b) Boost converter.



(c) Synchronous buck converter.

Fig. 6: Power converters: step responses (simulation).

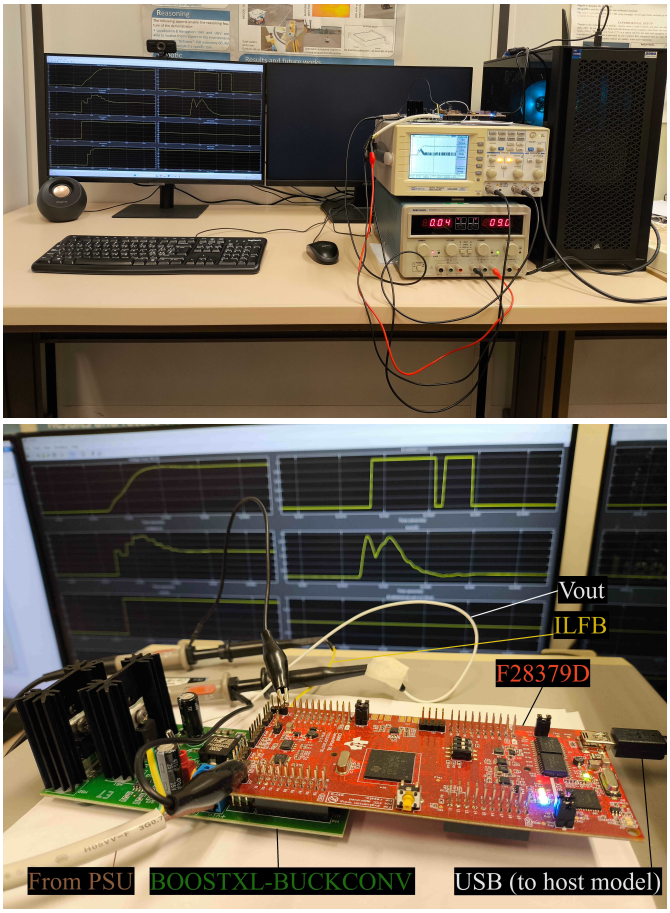


Fig. 7: Overview of the experimental setup.

are the same as in Section V-C (see Fig. 4c). A host computer is connected to the MCU via USB and is running a host model in Simulink®. The host computer is employed to continuously log and visualize signals (reference, modified reference, parameter identification results) at 66.6 kHz, as well as to sporadically send commands (enabling PWM, modifying reference voltage). The probes of a two channel digital oscilloscope (GW Instek GDS-820S) are connected to the "Vout" (measured output voltage) and "ILFB" (measured inductor current) pins on the BOOSTXL-BUCKCONV. The pinout is available in the technical documentation [40]. The output voltage signal is also set as the trigger signal.

TABLE IV: LAUNCHXL-F28379D specifications.

Parameter	Value
CPU	2x C28x, 2x CLA
Frequency	200 MHz
Flash memory	1024 kByte
RAM	204 kByte
ADC resolution	12 Bps, 16 Bps
Total processing	800 MIPS

Table IV reports the main MCU specifications. The MPC_{FO} parameters are the same as in Table I. The baseline controller is a PI controller and its default (pre-tuned) parameters are kept (see Section V-C), including the sample time for the inner control loop $T_{sc} = 5\,\mu\text{s}$ (200 kHz) and the previous duty cycle saturation constraint in $[0, 0.6]$. The sample times for both RLS_{FO} and MPC_{FO} are set to $100\,\mu\text{s}$ (10 kHz). The entire control scheme runs on a single C28x core, and the Control Law Accelerator (CLA) is not utilized.

In the following subsections, three transients are analyzed in detail. The voltage and current measurements are acquired using the digital oscilloscope (1 MHz sampling rate), while the references and the identification parameters are communicated by the device to the host PC via the USB serial port (66.6 kHz sampling rate). The average of the current measurements is also shown for clarity. The average is calculated using a 10-sample mean filter with forward-backward filtering to achieve zero-phase digital filtering. We stress that all the hardware and control software parameters just introduced, as well as the converter topology and the low-level control structure, are assumed to be unknown by the adaptive reference governor.

A. Switching on

In this first transient, the device is turned on: the required voltage is 2 V, while the load is 7 Ω. In Fig. 8, the solid blue line represents the output voltage, the yellow dashed line is the original reference, and the orange dashed line is the reference modified by MPC_{FO}. For comparison, the output voltage without MPC_{FO} is represented by a solid black line. In Fig. 8, $t = 0\,\text{s}$ is the instant when the device is switched on. Fig. 8 also reports the outcome of the RLS_{FO}, where the parameters a and b are represented by a solid blue line and a dashed red line, respectively, while the shaded area highlights the activation of parameter identification. For $t < 0\,\text{s}$, the initial conditions $a = 0$, $b = 1$ describe a model with no dynamics, hence the MPC_{FO} does not alter the reference when it triggers for the first time. The RLS_{FO} returns its first estimation of a and b after approximately 0.135 ms. The initial estimation is inaccurate, however, is immediately exploited by MPC_{FO}, which alters the modified reference. The estimation is refined in the subsequent steps and, consequently, the reference is increased to speed up the transient. After 1.23 ms, the system is in steady state and the identification is turned off because (12) does not hold. The improvement in terms of rise time employing MPC_{FO} is even more effective than in simulation (see Fig. 5c for comparison). Table V compares the performance indices of MPC_{FO} and the default PI controller. Both the rise time and the settling time benefit from the introduction of MPC_{FO}: the rise time is reduced by 58.8% and the IAE is reduced by 24.8%. However, the overshoot increases from 4.0% to 6.0%. Fig. 8 also shows the currents using the MPC_{FO}. Please note that they are comparable in magnitude to the baseline controller, and the peak current is increased by 6.8% using MPC_{FO}.

B. Reference increase

In the second transient, the reference rises from 2 V to 3 V. In Fig. 9, $t = 0\,\text{s}$ is the instant when the original reference is modified. At $t = 0.015$ ms, the MPC_{FO} reacts to the voltage drop by increasing the modified reference. At $t = 0.15$ ms, the RLS_{FO} algorithm detects the transient phase and the parameter update lasts for 0.885 ms, with minor parameter changes. The

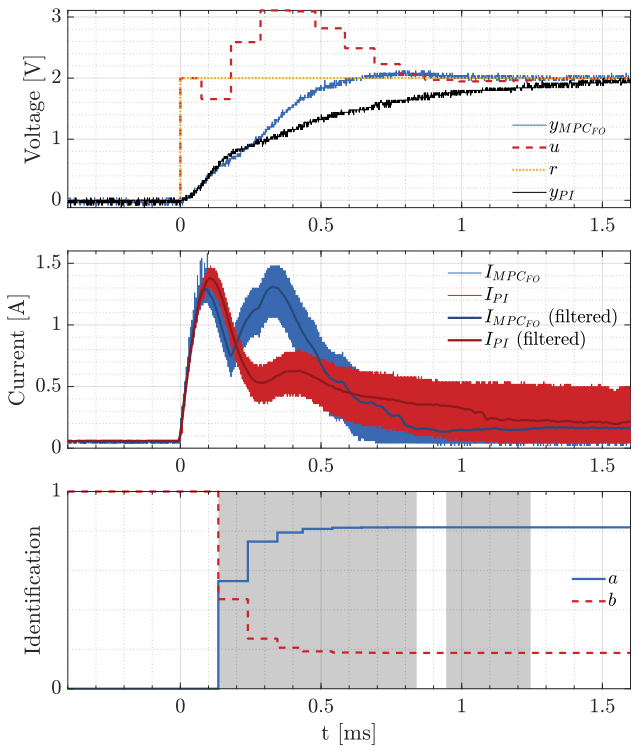


Fig. 8: Buck converter: switching on.

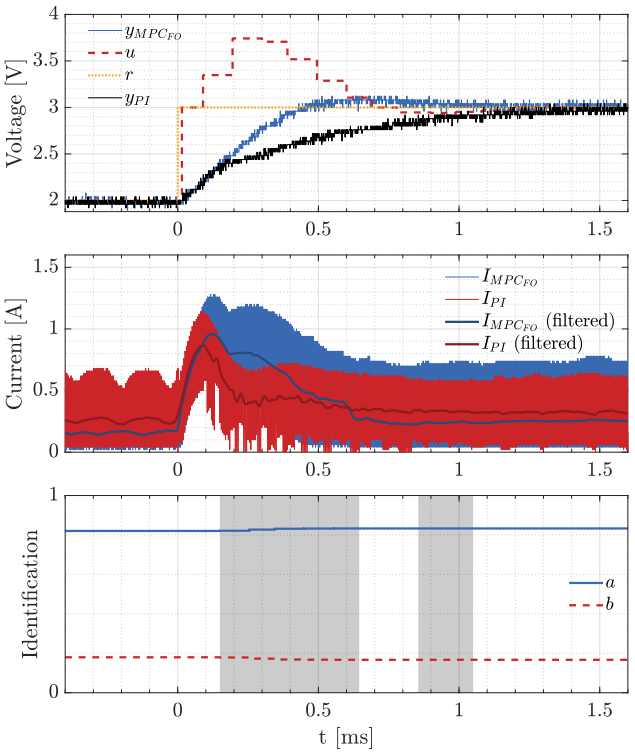


Fig. 9: Buck converter: reference increase.

TABLE V: Experimental Results for LAUNCHXL-F28379D on different scenarios

	IAE [V·μs]	**Peak current** [A]	**Rise time** [ms]	**Overshoot** [%]
Switching on				
PI	1112.72	1.48	1.010	4.0
MPC_{FO}	836.96	1.58	0.416	6.0
Ref. Increase				
PI	695.4	1.14	0.776	2.7
MPC_{FO}	450.76	1.28	0.335	4.0
Ref. Decrease				
PI	673.52	0.74	0.796	6.0
MPC_{FO}	485.08	0.80	0.321	10.0

estimated parameters are consistent, validating the reliability of the time-varying SISO model for the experimental setup. Even in this case, the rise time and the settle time benefit from the introduction of the MPC_{FO}. Table V reports the performance indices of MPC_{FO} and the default PI controller. The rise time with MPC_{FO} is reduced by 56.8% and the IAE is reduced by 35.2%. The improvement is larger than in the previous case because the prediction model better reflects the dynamics of the device. The overshoot increases from 2.7% to 4.0%. Fig. 8 also shows the currents using the MPC_{FO}. Please note that they are comparable in magnitude to the baseline controller, and the peak current is increased by 6.8% using MPC_{FO}.

C. Reference decrease

In the third transient, the original reference drops from 3 V to 2 V. In Fig. 10, $t = 0$ s is the instant when the original reference is modified, and the MPC_{FO} reacts at $t = 0.09$ ms by decreasing the modified reference. Table V reports the performance indices. The rise time with MPC_{FO} is reduced by 59.7% and the IAE is reduced by 28.0%. The improvement is lower in this case because the current is non-negative by construction of the buck converter. The undershoot increases from 6.0% to 10.0% and the peak current is increased by 8.1% using MPC_{FO}.

D. Discussion

The experimental results show that a few steps (up to 4 steps, i.e., 0.4 ms, see Fig. 8) are sufficient to obtain a reliable parameter identification for DC-DC converters. Then, the parameters exhibit minimal changes, even if the additional software-controlled load is activated. As the RLS_{FO} identifies the closed-loop control system, including both the DC-DC converter and its PI controller, we infer that the identified parameters do not change because the internal feedback compensates for the load variation, so that the time constant of the first order model is essentially unaffected by load changes. Also please note that the MPC_{FO} delay due to execution time, together with variable quantization, can generate a small ripple in the output voltage in steady state (± 0.01 V in the experimental setup). The issue is practically solved by overwriting the modified reference with the original reference, whenever their difference is less than a threshold (e.g., 0.02 V).

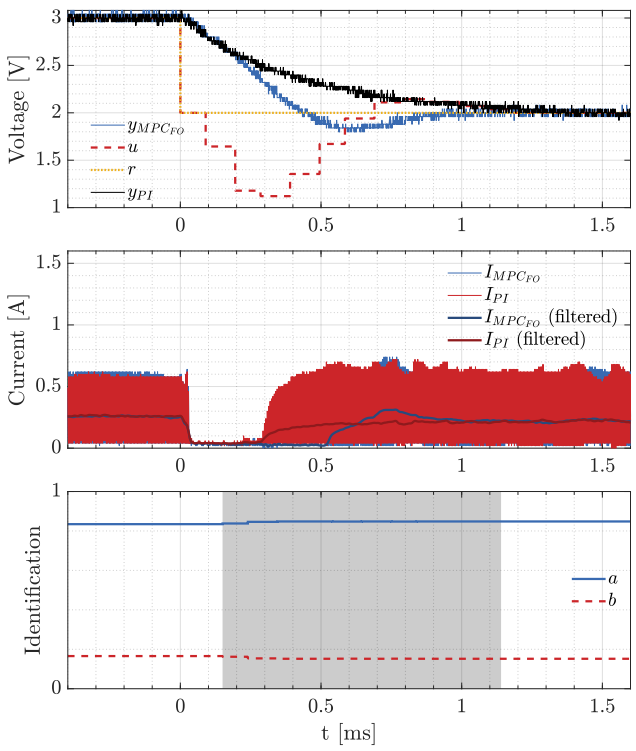


Fig. 10: Buck converter: reference decrease.

E. Comparison against other MPC software packages

Section IV has shown that QR_{FO} massively outperforms popular linear algebra packages, and that MPC_{FO} is faster than a tailored condensing routine (see steps (40)). The improvement is larger than an order of magnitude. Here we want to show the computational performance of our algorithm on the board selected for the experiments, and compare them to a popular commercial choice for implementing MPC controllers: the Model Predictive Control Toolbox™ from MATLAB®. It is a widely adopted solution both in research and industry to design and deploy MPC schemes into embedded control boards by means of the *Embedded Coder*. We design an adaptive reference governor with the *adaptive MPC* functionality, mathematically equivalent to the one implemented by MPC_{FO}, and test it on the F28379D control board. The numerical solution is obviously equivalent, thus not reported, but what is striking is the difference in throughput and memory occupancy. Fig. 11 shows the execution time on the board of the MPC_{FO} routine. The MPC_{FO} execution time increases linearly with the prediction horizon from 4 to 40, whereas the execution time of the Model Predictive Control Toolbox™ increases superlinearly, leading to a control loop that is faster by a factor starting from x3.6 (for prediction horizon 4) and increasing significantly with larger prediction horizons (x184 for prediction horizon 40). The difference is substantial also in memory occupancy, where, with a prediction horizon equal to 6, we have 0.32 kB for the adaptive reference governor implemented with the tools presented in this paper, and 0.82 kB for the one produced by MATLAB® tools.

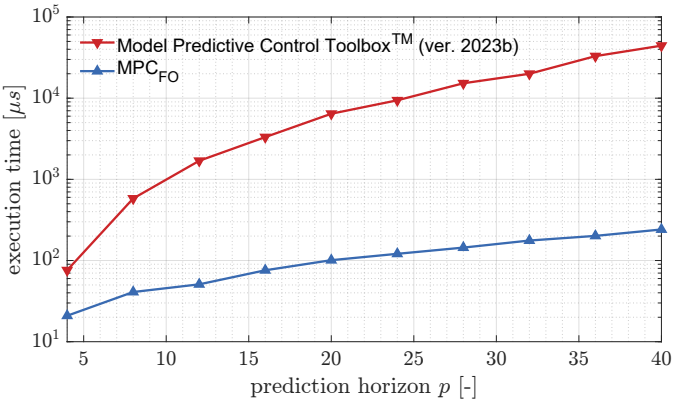


Fig. 11: Execution time comparison on the F28379D.

VII. Conclusion

In this paper, we have proposed an adaptive reference governor, based on MPC, to improve the transient response of various DC-DC converters for which the low-level controller is not modifiable. That is a typical scenario in production programs where either the power electronics is provided by a supplier, or the industrial certifications for robustness and safety preclude a potential change of the low-level controller. The topic is of recent interest in the industry, and here we have introduced the novelty of *adaptivity*, meaning that the mathematical model of the closed-loop system is time-varying. This substantial improvement led to a plug-and-play solution that is suitable to different converter topologies, widens the operating range, and deals more effectively with the nonlinearities of the closed-loop system, if compared to a more standard time-invariant approach. Nevertheless, the adaptive MPC based on a model learnt from data avoids the limitation of needing a white-box mathematical representation of the closed-loop. The various factors influencing the converters' dynamics and the manufacturer not disclosing the low-level controller details are indeed a serious obstacle for other known governor approaches.

The cornerstones of the control algorithm are an RLS routine for system identification that deals with the persistence of excitation problem and a novel efficient real-time optimizer for an equality constrained QP problem. The estimated parameters proved to be consistent over time, confirming that the main dynamics of the DC-DC converter were captured despite the simplicity of the model. Moreover, the MPC demonstrated tolerance to this type of model uncertainty, thanks to the receding horizon framework, which inherently acts as a form of feedback. We developed algebra routines tailored to the specific structure arising in first-order MPC that are shown to outperform well-known computationally efficient libraries by more than an order of magnitude. Throughput and memory occupancy of the proposed controller are also analyzed analytically, and the execution in the experimental setup configuration requires 219 FLOPs and a memory allocation of 33 floats, allowing the adaptive MPC to seamlessly run at 20 kHz on a conventional board for power conversion control.

Although the algorithm is presented for low-power DC-DC converters, it could also be applied to medium-power

applications, such as microgrids and industrial motors, as these are typically controlled with similar PI control schemes [9], [10]. However, for these applications, different specifications may apply (e.g., thermal requirements and EMIs); addressing these considerations requires specific investigation. Moreover, it is expected to effectively handle any system that satisfies Assumptions 1-4, as well as different low-level control algorithms beyond PI. Hence, several thermal systems, mechanical systems, and chemical processes could benefit from the proposed control scheme, as they often show a stable, first order closed-loop behaviour and piecewise constant references. Additionally, it can also be used as the low-level controller in those applications that satisfy such assumptions. On the other hand, the proposed scheme is expected to struggle in presence of a higher-order (or strongly nonlinear) closed-loop behaviour. Also, parameter identification may fail to detect parameter drifts until the persistence of excitation is restored.

REFERENCES

[1] Z. Cheng, Z. Li, S. Li, J. Gao, J. Si, H. S. Das, and W. Dong, "A novel cascaded control to improve stability and inertia of parallel buck-boost converters in dc microgrid," *International Journal of Electrical Power & Energy Systems*, vol. 119, p. 105950, 2020.

[2] V. Bist and B. Singh, "An adjustable-speed pfc bridgeless buck–boost converter-fed bldc motor drive," *IEEE transactions on industrial electronics*, vol. 61, no. 6, pp. 2665–2677, 2013.

[3] J. Moon, J. Lee, K. Javed, J. Hong, and J. Roh, "Concurrent current and voltage regulated buck–boost converter for automotive led matrix headlights," *IEEE Transactions on Power Electronics*, vol. 38, no. 5, pp. 6015–6023, 2023.

[4] W. Lee, Y. Wang, D. Shin, N. Chang, and M. Pedram, "Optimizing the power delivery network in a smartphone platform," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 1, pp. 36–49, 2013.

[5] G. R. Walker and P. C. Sernia, "Cascaded DC-DC converter connection of photovoltaic modules," *IEEE transactions on power electronics*, vol. 19, no. 4, pp. 1130–1139, 2004.

[6] H. Komurcugil, S. Biricik, S. Bayhan, and Z. Zhang, "Sliding mode control: Overview of its applications in power converters," *IEEE Industrial Electronics Magazine*, vol. 15, no. 1, pp. 40–49, 2021.

[7] C. A. Soriano-Rangel, W. He, F. Mancilla-David, and R. Ortega, "Voltage regulation in buck–boost converters feeding an unknown constant power load: An adaptive passivity-based control," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 1, pp. 395–402, 2021.

[8] P. Karamanakos, E. Liegmann, T. Geyer, and R. Kennel, "Model predictive control of power electronic systems: Methods, results, and challenges," *IEEE Open Journal of Industry Applications*, vol. 1, pp. 95–114, 2020.

[9] Q. Xu, N. Vafamand, L. Chen, T. Dragičević, L. Xie, and F. Blaabjerg, "Review on advanced control technologies for bidirectional dc/dc converters in dc microgrids," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 9, no. 2, pp. 1205–1221, 2021.

[10] G. Mirzaeva and D. Carter, "High performance control of power converters in industrial ac drive applications," *IEEE Transactions on Industry Applications*, pp. 1–10, 2024.

[11] E. Garone, S. Di Cairano, and I. Kolmanovsky, "Reference and command governors for systems with constraints: A survey on theory and applications," *Automatica*, vol. 75, pp. 306–328, 2017.

[12] L. Cavanini, G. Cimini, and G. Ippoliti, "Model predictive control for pre-compensated power converters: Application to current mode control," *Journal of the Franklin Institute*, vol. 356, no. 4, pp. 2015–2030, 2019.

[13] L. Cavanini, G. Cimini, G. Ippoliti, and A. Bemporad, "Model predictive control for pre-compensated voltage mode controlled DC–DC converters," *IET Control Theory & Applications*, vol. 11, no. 15, pp. 2514–2520, 2017.

[14] C. Yfoulis, "An MPC Reference Governor Approach for Enhancing the Performance of Precompensated Boost DC–DC Converters," *Energies*, vol. 12, no. 3, 2019.

[15] C. Yfoulis, S. Papadopoulou, and S. Voutetakis, "Enhanced control of a buck-boost DC-DC converter via a closed-form MPC reference governor scheme," in *IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society*, vol. 1. IEEE, 2019, pp. 365–370.

[16] L. Cavanini, G. Cimini, F. Ferracuti, and G. Ippoliti, "Reference governor for switching converters with power factor correction," in *2021 European Control Conference (ECC)*. IEEE, 2021, pp. 2549–2554.

[17] G. Seenumani, H. Peng, and J. Sun, "A reference governor-based hierarchical control for failure mode power management of hybrid power systems for all-electric ships," *Journal of Power Sources*, vol. 196, no. 3, pp. 1599–1607, 2011.

[18] A. Hajizadeh, A. H. Shahirinia, N. Namjoo, and D. C. Yu, "Self-tuning indirect adaptive control of non-inverting buck–boost converter," *IET Power Electronics*, vol. 8, no. 11, pp. 2299–2306, 2015.

[19] N. Namjoo and A. Hajizadeh, "Adaptive control of non-inverting buck-boost converter," in *The 5th Annual International Power Electronics, Drive Systems and Technologies Conference (PEDSTC 2014)*. IEEE, 2014, pp. 211–216.

[20] H. Mollaee, S. M. Ghamari, S. A. Saadat, and P. Wheeler, "A novel adaptive cascade controller design on a buck–boost dc–dc converter with a fractional-order pid voltage controller and a self-tuning regulator adaptive current controller," *IET Power Electronics*, vol. 14, no. 11, pp. 1920–1935, 2021.

[21] S. M. Ghamari, F. Khavari, H. Molaee, and P. Wheeler, "Generalised model predictive controller design for a dc–dc non-inverting buck–boost converter optimised with a novel identification technique," *IET Power Electronics*, vol. 15, no. 13, pp. 1350–1364, 2022.

[22] G. Rojas-Duenas, J.-R. Riba, and M. Moreno-Eguilaz, "Nonlinear least squares optimization for parametric identification of dc–dc converters," *IEEE Transactions on Power Electronics*, vol. 36, no. 1, pp. 654–661, 2020.

[23] B. D. Anderson, "Exponential convergence and persistent excitation," in *1982 21st IEEE Conference on Decision and Control*. IEEE, 1982, pp. 12–17.

[24] N. R. Sripada and D. G. Fisher, "Improved least squares identification for adaptive controllers," in *1987 American Control Conference*. IEEE, 1987, pp. 2027–2037.

[25] G. Chowdhary and E. Johnson, "Concurrent learning for convergence in adaptive control without persistency of excitation," in *49th IEEE Conference on Decision and Control (CDC)*. IEEE, 2010, pp. 3674–3679.

[26] S. Kouro, M. Perez, J. Rodriguez, A. Llor, and H. Young, "Model predictive control: MPC's role in the evolution of power electronics," *Industrial Electronics Magazine, IEEE*, vol. 9, no. 4, pp. 8–21, Dec 2015.

[27] I. Harbi, J. Rodriguez, E. Liegmann, H. Makhamreh, M. L. Heldwein, M. Novak, M. Rossi, M. Abdelrahem, M. Trabelsi, M. Ahmed, P. Karamanakos, S. Xu, T. Dragičević, and R. Kennel, "Model-predictive control of multilevel inverters: Challenges, recent advances, and trends," *IEEE Transactions on Power Electronics*, vol. 38, no. 9, pp. 10 845–10 868, 2023.

[28] S. Vazquez, J. Rodriguez, M. Rivera, L. G. Franquelo, and M. Norambuena, "Model predictive control for power converters and drives: Advances and trends," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 2, pp. 935–947, 2017.

[29] G. Cimini, G. Ippoliti, G. Orlando, and M. Pirro, "Explicit sensorless model predictive control of synchronous buck converter," in *2013 International Conference on Renewable Energy Research and Applications (ICRERA)*, 2013, pp. 1200–1205.

[30] L. Ljung and T. Söderström, *Theory and practice of recursive identification*. MIT press, 1983.

[31] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 03 1960.

[32] N. Guler, S. Biricik, S. Bayhan, and H. Komurcugil, "Model predictive control of dc–dc sepic converters with autotuning weighting factor," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 10, pp. 9433–9443, 2020.

[33] O. Malik, G. Hope, and S. Cheng, "Some issues on the practical use of recursive least squares identification in self-tuning control," *International Journal of Control*, vol. 53, no. 5, pp. 1021–1033, 1991.

[34] R. Scattolini, "Architectures for distributed and hierarchical model predictive control – a review," *Journal of Process Control*, vol. 19, no. 5, pp. 723–731, 2009.

[35] C. Ciurans, N. Bazmohammadi, L. Poughon, J. C. Vasquez, C. G. Dussap, F. Gòdia, and J. M. Guerrero, "Hierarchically controlled ecological

life support systems," *Computers & Chemical Engineering*, vol. 157, p. 107625, 2022.
[36] M. T. da Silva, G. A. Júnior, and P. R. Barros, "An iterative closed-loop identification method for continuous-time fopdt model using equality constraints," *Journal of Control, Automation and Electrical Systems*, vol. 33, no. 2, pp. 664–676, Apr 2022.
[37] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU press, 2013.
[38] J. H. Wilkinson, *The algebraic eigenvalue problem*. Oxford University Press, Inc., 1988.
[39] Texas Instruments, "F28379D documentation," last accessed 17 June 2024. [Online]. Available: https://www.ti.com/product/TMS320F28379D
[40] ——, "TIDM-DC-DC-BUCK documentation," last accessed 17 June 2024. [Online]. Available: https://www.ti.com/lit/ug/tidu986a/tidu986a.pdf

Gionata Cimini received the master's degree in Computer and Automation Engineering and the Ph.D. degree in Information Engineering from Università Politecnica delle Marche, Ancona, Italy, in 2012 and 2017, respectively. He has been a Guest Ph.D. Scholar with the IMT School for Advanced Studies Lucca, and he was the recipient of the Best Ph.D. Thesis in Systems and Control Engineering, Italy, 2017. He held Research Assistant positions with the Information Department, Università Politecnica delle Marche and with the Automotive Research Center, University of Michigan, MI, USA. He worked as a control engineer for General Motors Company, Detroit, MI, USA. Since 2017 he works at ODYS S.r.l., Milano, Italy, where he is currently the Technical Lead for Numerical Optimization. He is the coinventor of 4 patents, and the author or co-author of commercial optimization and model predictive control software packages. His research interests include embedded optimization, model predictive control, and their application to problems in the automotive, aerospace, and energy domains.

Riccardo Felicetti received his Master's Degree cum laude in Computer and Automation Engineering and his Ph.D. degree cum laude in Information Engineering from Università Politecnica delle Marche, in 2016 and 2021, respectively. Since 2019, he has been a postdoctoral research fellow with Università Politecnica delle Marche. His main research interests are fault detection and diagnosis, fault-tolerant control, and optimization with applications to unmanned vehicles and energy management systems.

Francesco Ferracuti received his Ph.D. degree in automation, information and management engineering from Università Politecnica delle Marche, Ancona, Italy, in 2014. He is a Researcher at Università Politecnica delle Marche. His research interests include model-based and data-driven fault diagnosis, signal processing, statistical pattern recognition, system identification, and their applications in industry.

Luca Cavanini received his Ph.D. degree in automation, information and management engineering from Università Politecnica delle Marche, Ancona, Italy, in 2018. He works as a Technical Consultant at Industrial Systems and Control Ltd. His activity includes model predictive control, autonomous mobile robotics and artificial intelligence, and machine learning techniques for control systems.

Andrea Monteriù (S'04-M'06) received his M.Sc. degree in Electronic Engineering and Ph.D. degree in Artificial Intelligence Systems from Università Politecnica delle Marche, Italy in 2003 and 2006, respectively. Currently, he is Associate Professor at the Department of Information Engineering, Università Politecnica delle Marche where he leads the Laboratory of Artificially Intelligent Robotics (former Robotics Laboratory). His research interests primarily include fault diagnosis, fault tolerant control, nonlinear dynamics and control, periodic and stochastic system control, applied in different fields including aerospace, marine, robotic and unmanned artificial intelligent systems.