İSTANBUL UNIVERSITY CERRAHPAŞA

ORIGINAL ARTICLE
Electrica 2024; 24(3): 722-732

Implementation of Generalized Model Predictive Control Algorithm for DC-DC Boost Converter on STM32

Ali Hmidene[1], Hichem Bennasr[2], Faouzi M'sahli[3]

[1]Department of Electrical Engineering, Higher Institute of Technological Studies of Sousse, Direction of ISETs, Erriadh, Tunisia
[2]Department of Electrical Engineering, Higher Institute of Technological Studies of Sfax, Direction of ISETs, Tunisia
[3]Department of Electrical Engineering, University of Monastir, National Engineering School of Monastir, Tunisia

Cite this article as: A. Hmidene, H. Bennasr and F. M'sahli, "Implementation of generalized model predictive control algorithm for DC-DC boost converter on STM32," *Electrica*, 24(3), 722-732, 2024.

ABSTRACT

This paper presents the practical implementation of a Generalized Model Predictive Control (GPC) algorithm using low-cost microcontrollers. The computational efficiency of the predictive controller is achieved through an explicit analytical solution calculated offline. The proposed approach is implemented, reviewed, and compared with academic benchmarks and complex dynamic systems. Experimental results are provided using both Black-Box and White-Box models based on a boost converter, with a comparison between the two models. The algorithm demonstrates good control performance with low computational time, making it suitable for real-time applications in systems with fast dynamics.

***Index Terms*—**Boost converter, fast dynamic systems, microcontroller, MPC controller, reduction of time calculations

Corresponding author:
Ali Hmidene

E-mail:
ali.hmidene@gmail.com

Received: June 12, 2024

Revision Requested: August 12, 2024

Last Revision Received: September 16, 2024

Accepted: September 25, 2024

Publication Date: November 8, 2024

DOI: 10.5152/electrica.2024.24059

I. INTRODUCTION

Predictive control, an evolved method under optimal control techniques, is often known as Model-Based Predictive Control (MBPC). The conceptual scheme of MBPC is given in [1]. The main idea is to find an optimal control sequence for each iteration by considering the future behavior of the process predicted by a model. This technique is mainly used in chemical and petroleum industries, distinguished by slow processes. Predictive control, which relies on solving an optimization problem online, is mostly expensive in terms of computation time and can require significant resources. Therefore, although it provides high performance over long control horizons, it comes at the cost of consuming time and resources.

In contrast, shorter control horizons reduce the computational burden but increase the risk of unfeasibility. Predictive control was initially developed for linear systems. Early formulations involve establishing analytical expressions for optimal solutions, which can be solved using optimization techniques such as quadratic programming to handle constraints. Among these formulations, we may mention generalized predictive control [2-4].

The computational time required to solve quadratic programming problems can be critical for short sampling times [5]. Moreover, presenting significant challenges [6] and [7]. However, such techniques remain a largely open problem for fast systems due to the computational time needed to solve the quadratic problem. As a result, model predictive control is limited to a slow process with a sampling time of seconds or minutes [8, 9]. With fast digital signal controllers, it is now possible to develop model predictive control algorithms for fast dynamic systems requiring short sampling times, less than a second. There are examples of fast model predictive control applications given in [10, 11].

Significant results and advances in the implementation of embedded applications have been applied to relatively slow plants [12, 13]. The online optimization procedure is used both in slow industrial process control systems and in many embedded applications [14]. Many works propose a method to speed up the model predictive control algorithms without reducing the computation load, as cited in [15-18].

In many publications, for selected parameters such as prediction and control horizon, model predictive control algorithms work very well [19, 20], but the influence of computation time and the necessary memory allocation for the algorithm is not studied [21-23].

In our work, we focus on developing a predictive control algorithm that considers the minimization of calculation time on the one hand and the allocated memory space on the other, making it easy and possible to use in fast systems.

It should be noted that the setting of predictive control parameters has an influence on the computational load. So, indeed, when the sampling periods become too short, while the computation time is too long, the generalized predictive control algorithm does not work. On the other hand, the execution time requires only tens of microseconds when it comes to fast algorithms [24, 25].

Various devices have been introduced to enable digital control, with the dSPACE platform standing out as one of the most widely adopted due to its flexibility in controlling power electronic converters [26-28]. The increasing accessibility of high-performance microcontrollers has also facilitated the adoption of advanced control techniques, such as Model Predictive Control (MPC), in power electronics [29-31].

In this work, an implementation of Generalized Predictive Control (GPC) on a boost converter has been presented, effectively avoiding the need for an optimization procedure.

The organization of the paper is structured into several sections. Section 2 describes the model predictive control algorithms. In Section 3, we present the software and hardware implementation algorithm using the microcontroller. An experimental case is studied in the lab; it concerns a boost converter. Some results are presented and discussed to illustrate the performance of the proposed new approach in time computing. This part concerns Section 4. Finally, a conclusion is presented to close the paper.

II. MODEL PREDICTIVE CONTROL ALGORITHM DESCRIPTION

In our study, we focus on a single input/single output (SISO) system described by a CARIMA model:

$$A(z^{-1})y(k) = B(z^{-1})u(k) + \frac{C(z^{-1})\xi(k)}{\Delta(z^{-1})} \tag{1}$$

where $A(z^{-1})$, $B(z_{-1})$ and $C(z^{-1})$ are the polynomials:

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a} \tag{2}$$

$$B(z^{-1}) = b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b} \tag{3}$$

$$C(z^{-1}) = 1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + c_{n_c} z^{-n_c} \tag{4}$$

and $\Delta(z^{-1}) = 1 - z^{-1}$

By using the incremental form:

$$A(z^{-1})\Delta y(k) = B(z^{-1})\Delta u(k) + C(z^{-1})\xi(k) \tag{5}$$

$\xi(k)$ is the white noise with zero-mean which we ignore for the best prediction:

$$A(z^{-1})\Delta y(k) = B(z^{-1})\Delta u(k) \tag{6}$$

where:

$$A(z^{-1})\Delta = 1 + A_1 z^{-1} + A_2 z^{-2} + \cdots + A_{n_A} z^{-n_A} \tag{7}$$

and $n_A = n_a + 1$

Let H_p denote the prediction horizon. The difference equations for H_p ahead prediction are obtained in recursive form as follows:

$$\begin{aligned} &y(k+H_p) + A_1 y(k+H_p-1) + A_1 y(k+H_p-1) \\ &+ \cdots + A_{n_A} y(k+H_p-n_A+2) = b_1 \Delta u(k+H_p-1) \\ &+ b_2 \Delta u(k+H_p-2) + \cdots + b_{n_b} \Delta u(k+H_p-n_b) \end{aligned} \tag{8}$$

where H_c is the control horizon, there is no control action after H_p steps, i.e. $\Delta u(k+l \mid k) = 0$ for $l \geq H_c$. It is easier to use the compact matrix/vector form:

$$\begin{aligned} &\underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ A_1 & 0 & \cdots & 0 \\ A_2 & A_1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}}_{C_A} \underbrace{\begin{bmatrix} y(k+1) \\ y(k+2) \\ \vdots \\ y(k+H_p) \end{bmatrix}}_{\mathbf{y}} + \\ &\underbrace{\begin{bmatrix} A_1 & A_2 & \cdots & A_{n_A} \\ A_2 & A_3 & \cdots & 0 \\ A_3 & A_4 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}}_{\mathbf{H_A}} \underbrace{\begin{bmatrix} y(k) \\ y(k-1) \\ \vdots \\ y(k-n_a) \end{bmatrix}}_{\mathbf{y_{past}}} = \\ &\underbrace{\begin{bmatrix} b_1 & 0 & \cdots & 0 \\ b_2 & b_1 & \cdots & 0 \\ b_3 & b_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}}_{\mathbf{C_{zb}}} \underbrace{\begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+H_c-1) \end{bmatrix}}_{\Delta \mathbf{u}} + \\ &\underbrace{\begin{bmatrix} b_2 & b_3 & \cdots & b_{n_b} \\ b_3 & b_4 & \cdots & 0 \\ b_4 & b_5 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}}_{\mathbf{H_{zb}}} \underbrace{\begin{bmatrix} \Delta u(k-1) \\ \Delta u(k-2) \\ \vdots \\ \Delta u(k-n_b+1) \end{bmatrix}}_{\Delta \mathbf{u_{past}}} \end{aligned} \tag{9}$$

Therefore:

$$\mathbf{C_A y + H_A y_{past} = C_{zb} \Delta u + H_{zb} \Delta u_{past}} \tag{10}$$

The output prediction is:

$$\mathbf{y = H \Delta u + P \Delta u_{past} + Q y_{past}} \tag{11}$$

where:

$$\mathbf{H = C_A^{-1} C_{zb}\ ,\quad P = C_A^{-1} H_{zb}\ \ and\ \ Q = -C_A^{-1} H_A} \tag{12}$$

A. Matrix Coefficients of H, P and Q

These coefficients will be determined recursively. The first step in prediction is:

$$\begin{aligned} y(k+1) &= b_1\Delta u(k) + b_2\Delta u(k-1) + \cdots + \\ & b_{n_n}\Delta u(k-n_b+1) + A_1 y(k) + \\ & A_2 y(k-1) + \cdots + A_{n_A} y(k-n_A+1) \end{aligned} \tag{13}$$

Introducing the notion of the prediction horizon $(\cdot)^{[i]}$ designed to denote the i-step ahead prediction, such that in general:

$$y(k+i) = \mathbf{H^{[i]}}\Delta\mathbf{u} + \mathbf{P^{[i]}}\Delta\mathbf{u_{past}} + \mathbf{Q^{[i]}}\mathbf{y_{past}} \tag{14}$$

Initially, for $i = 1$:

$$\begin{cases} \mathbf{H^{[i]}} = [b_1, 0, 0, 0, \cdots] \\ \mathbf{P^{[i]}} = [b_1, b_2, b_3, \cdots, b_{n_b}] \\ \mathbf{Q^{[i]}} = [-A_1, -A_2, -A_3, \cdots, -A_{n_A}] \end{cases} \tag{15}$$

The other rows of the matrices *H*, *P,* and *Q* are obtained recursively:

$$\begin{cases} \mathbf{H^{[i+1]}} = \left[\mathbf{P_1^{[i]}} + \mathbf{Q_1^{[i]}}\mathbf{H_1^{[1]}}, \mathbf{H_1^{[i]}}\right] \\ \mathbf{P^{[i+1]}} = \left[\mathbf{P_2^{[i]}}, \cdots, + \mathbf{P_{nb\text{-}1}^{[i]}}, 0\right] + \mathbf{Q_1^{[i]}}\mathbf{P^{[1]}} \\ \mathbf{Q^{[i+1]}} = \left[\mathbf{Q_2^{[i]}}, \cdots, + \mathbf{P_{nA\text{-}1}^{[i]}}, 0\right] + \mathbf{Q_1^{[i]}}\mathbf{Q^{[1]}} \end{cases} \tag{16}$$

B. Synthesis of the Controller

The future increments are calculated online by the MPC algorithm.

$$\Delta\mathbf{u} = [\Delta u(k), \Delta u(k+1), \cdots, \Delta u(k+H_c-1)]^T \tag{17}$$

In order to be minimized, the cost function can be written as:

$$J = \|\mathbf{r} - \mathbf{y}\|_2^2 + \lambda\|\Delta\mathbf{u}\|_2^2 \tag{18}$$

By substituting the general expression of y given by equation (11), the cost function can be rewritten as:

$$J = \|\mathbf{r} - \mathbf{H}\Delta\mathbf{u} - \mathbf{P}\Delta\mathbf{u_{past}} - \mathbf{Q}\mathbf{y_{past}}\|_2^2 + \lambda\|\Delta\mathbf{u}\|_2^2 \tag{19}$$

and

$$\begin{aligned} \min_{\Delta u} J = {} & \Delta\mathbf{u}^{\mathbf{T}}\left(\mathbf{H^T}\mathbf{H} + \lambda\mathbf{I}\right)\Delta\mathbf{u} + 2\Delta\mathbf{u}^{\mathbf{T}}\mathbf{H}^{\mathbf{T}}[\mathbf{P}\Delta\mathbf{u_{past}} \\ & + \mathbf{Q}\mathbf{y_{past}} - \mathbf{r}] + \|\mathbf{r} - \mathbf{P}\Delta\mathbf{u_{past}} - \mathbf{Q}\mathbf{y_{past}}\|_2^2 \end{aligned} \tag{20}$$

where

$$\Delta\mathbf{u} = \left(\mathbf{H^T}\mathbf{H} + \lambda\mathbf{I}\right)^{-1}\mathbf{H}^{\mathbf{T}}[\mathbf{r} - \mathbf{P}\Delta\mathbf{u_{past}} - \mathbf{Q}\mathbf{y_{past}}] \tag{21}$$

The control action is defined by the first element of the increment: $\Delta u_k = \mathbf{e_1^T}\Delta\mathbf{u}$ with $\mathbf{e_1^T} = [1, 0, 0, \cdots, 0]$.

$$\Delta u_k = \mathbf{P_r}\mathbf{r} - \mathbf{N_k}\Delta\mathbf{u_{past}} - \mathbf{D_k}\mathbf{y_{past}} \tag{22}$$

and

$$\begin{cases} \mathbf{P_r} = \mathbf{e_1^T}\left(\mathbf{H^T}\mathbf{H} + \lambda\mathbf{I}\right)^{-1}\mathbf{H}^{\mathbf{T}} \\ \mathbf{N_k} = \mathbf{e_1^T}\left(\mathbf{H^T}\mathbf{H} + \lambda\mathbf{I}\right)^{-1}\mathbf{H}^{\mathbf{T}}\mathbf{P} \\ \mathbf{D_k} = \mathbf{e_1^T}\left(\mathbf{H^T}\mathbf{H} + \lambda\mathbf{I}\right)^{-1}\mathbf{H}^{\mathbf{T}}\mathbf{Q} \end{cases} \tag{23}$$

with:

$$\begin{cases} \mathbf{P_r} = [p_1, p_2, p_3, \cdots, p_{H_p}] \\ \mathbf{N_k} = [N_1, N_2, N_3, \cdots, N_{n_b-1}] \\ \mathbf{D_k} = [D_1, D_2, D_3, \cdots, D_{n_A}] \end{cases} \tag{24}$$

$$\Delta u_k = \sum_{i=1}^{H_p} p_i r_i - \sum_{i=1}^{n_b-1} N_i\Delta u(k-i) - \sum_{i=1}^{n_A} D_i y(k-i+1) \tag{25}$$

The manipulated variable $u(k)$ to be applied to the process is defined by: $u(k) = \Delta u(k) + u(k-1)$.

Without anticipation, the r_i coefficients are determined offline by: $C_p = \sum_{i=1}^{H_p} p_i$

So, Δu_k is reduced to the following expression:

$$\Delta u_k = C_p r_k - \sum_{i=1}^{n_b-1} N_i\Delta u(k-i) - \sum_{i=1}^{n_A} D_i y(k-i+1) \tag{26}$$

Δu_k is independent of H_p and H_c.

III. PLATFORM OF MPC ALGORITHM IMPLEMENTATION

The MPC algorithms have been successfully implemented on the low-cost STM32F746 Discovery board. The STM32F746, clocked at 216 MHz, is a digital signal controller (DSC) based on an Arm RISC Cortex-M7 32-bit core that supports single-cycle SIMD and DSP instructions and integrates a hardware floating point unit (FPU). These characteristics make it a great candidate for real-time embedded applications, requiring good performance and low power consumption.

The digital signal controller is programmed using the Keil µvision MDK-ARM toolchain. The integrated development environment platform (IDE) includes necessary tools for developing embedded applications, such as a macro assembler, C/C++ compiler, linker, and programmer. Moreover, it generates HEX files and AXF files for program debugging.

We configured the analog-to-digital converter ADC1 in scan mode in conjunction with the Direct Memory Access (DMA) for transferring the converted data of regular group channels to SRAM after each conversion operation. A timer (TIM10) is used to generate a Pulse Width Modulation (PWM) signal to control the boost converter transistor. This signal has a frequency of 80 kHz and a resolution of 0.1% on the duty cycle.

In terms of programming techniques, using the hardware floating point unit (FPU) and the DSP_lib library developed by ARM for

processing matrix operations considerably improves the calculation time. The program embedded in the microcontroller has two phases: offline and online. Thus, the MPC algorithm is divided into two parts: Initialization and Control.

A. Initialization Part

This part consists of determining the necessary vectors to calculate the control action, which is greedy in memory and execution time. Generally, these vectors are automatically calculated offline by software such as Matlab. In this work, we opted to implement the whole algorithm in the microcontroller and consequently estimate the execution time of each phase. This phase includes the following stages:

Define the prediction horizon H_p, the control horizon H_c, the λ factor, and the different coefficients a_i and b_i of the Boost model. For example:

```
#defineHp 15 //prediction horizon
#defineHc5 //control horizon
#definenB2 //numberofcoef. in B
#definenA3 //number of coef. in A
float32_t B[nB] ={0.7935, 0.00833};
float32_t A[nA] ={1, -0.9411, 0.3266};
```

- Calculate the polynomial coefficients $A(z^{-1})\Delta$,
- Calculate the matrix coefficients H, P, and Q,
- Calculate the vectors P_k, N_k and D_k.

B. Control Part

The sampling period is set by a timer SysTick at 1 ms. The target samples are pre-programmed in a table. At the start of each sampling period, the following sequences are carried out:

1. Offset of past samples from the system output.
2. Measurement of the output signal. This value is recorded at the start of the past vector of the outputs.
3. Calculation of the new value of the manipulated variable.
4. Shifting the previous samples of the used variable and saving the new command at the head of the table.
5. Application of the new command to the system.

We introduced a one-period delay in the model to compensate for the execution time of the GPC algorithm. At a given time k, the incremental control variable is obtained in a very compact way with a few lines of code.

```
for(i= 0; i<nh ; i++)
        du[k]= du[k] + Pr[i]*r[k+i];
for(i= 0; i<nA ; i++)
        du[k]= du[k] + Dk[i]*yp[i];
for(i= 0; i< nB-1 ; i++)
        du[k]= du[k] + Nk[i]*dup[i];
u[k]= u[k-1] + du[k];
```

Our system is controlled by a pulse width modulation (PWM) signal whose duty cycle is limited between 0 and 0.8. Optimal control can generate a command that exceeds the physical constraints of the system; these constraints are generally addressed in the optimization problem. In order to avoid overloading the control action, we have opted to make a slight modification to the control law by limiting the control variable between two saturation thresholds, U_{min} and U_{max}. The saturation of control variables is given by:

```
if(u[k]>Umax)
        {       u[k] = Umax;
                du[k] = Umax - u[k-1];
        }
else
        if(u[k] <Umin)
        {
        u[k] = Umin;
                du[k] = Umin - u[k-1];
        }
```

IV. EXPERIMENT RESULTS

A. Emulated SISO process

In order to compare the proposed strategy, we consider the following example taken from [32]. It is defined by the continuous-time first-order transfer function given by:

$$G(s) = \frac{1}{0.04s + 1} \tag{27}$$

The implementation of the proposed algorithm in the unconstrained and constrained cases is shown in Fig. 1. The algorithm is able to control the process so that where the set-point is changed, the output follows the set-point without any steady-state error. In the constrained case, the process control input, delimited by $0 \leq u < 3$, is also respected. The time calculation is evaluated to be about 1.74 µs compared to 2.17 µs given in [26], which is typically small. Thus, this approach improves a strong point by solving computation for fast dynamics systems.

To verify the ability of the proposed concept to reject disturbances, an output disturbance of the values –0.4 and 0.4 is introduced, respectively, at 250 ms and 720 ms (Fig. 1). The disturbance is eliminated, revealing a satisfactory ability to reject disturbances. As part of a simulation-based comparison, it is proved that our system control algorithm is able to deliver a significantly improved performance compared to conventional MPC. Therefore, the difficulty of minimizing a performance function for MPC is avoided. Thus, the conventional MPC usually uses at each sampling time a programming technique whose time consumption cannot be neglected.

B. Boost Process

The control signal is supplied by a PWM module. This signal controls the switching of the power transistor via the MOSFET driver. The output of the boost converter is connected to the input of the analog-to-digital converter (ADC) of the microcontroller. The system to be identified has the following characteristics:

Supply voltage: $V_g = 12$V,

Output voltage : $V_o = 24$ V,

Switching frequency: 80 kHz,

Load resistance: $R = 50\Omega$,

Inductance: $L = 430$ µH

Capacitance: C = 440 µF

ESR of capacitor: $R_c = 80$ mΩ

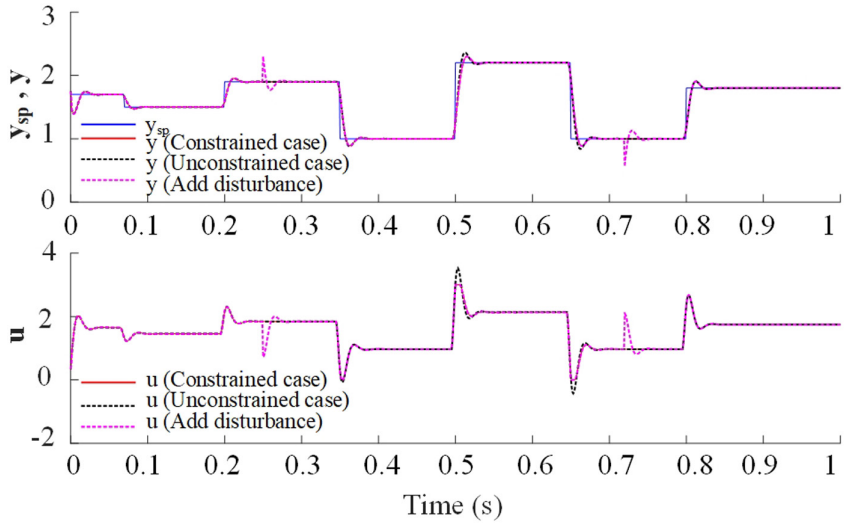

Fig. 1. Emulated SISO process: results of the experiments using STM32.

The boost converter bloc diagram is given in Fig. 2.

Several methods are used to identify system parameters. In this work, we use two approaches for modeling the boost converter: the white-box and black-box methods, and compare the two approaches.

C. White-Box Model

Figure 3 illustrates the topology of a boost converter operating in Continuous Conduction Mode (CCM). In this model, both the transistor and the diode are assumed to be ideal.

1) Switched model

The modeling of the switched system involves establishing differential equations for each of the converter's operating states [33, 34]. Two distinct states are considered: one where the switch is in the closed position and the diode is non-conductive, and another where the switch is open and the diode allows current flow. The first state, corresponding to the closed switch, is characterized by the state-space equations (28) and (29).

$$\frac{d}{dt}\begin{bmatrix} i_L(t) \\ v_c(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & -\dfrac{1}{(R+R_c)C} \end{bmatrix}}_{\mathbf{A_1}} \begin{bmatrix} i_L(t) \\ v_c(t) \end{bmatrix} + \underbrace{\begin{bmatrix} 1/L \\ 0 \end{bmatrix}}_{\mathbf{B_1}} V_g \tag{28}$$

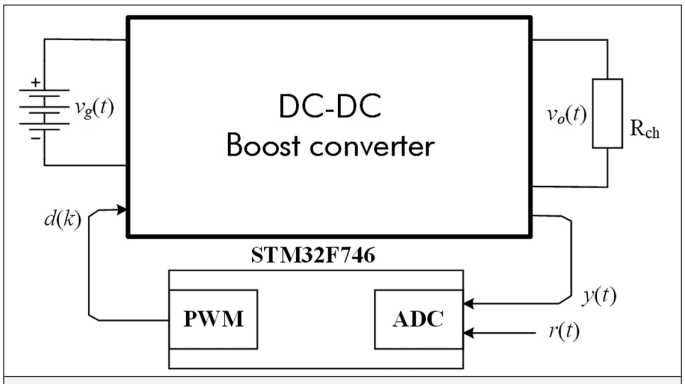

Fig. 2. Boost converter bloc diagram.

$$v_o(t) = \underbrace{\begin{bmatrix} 0 & \dfrac{R}{R+R_c} \end{bmatrix}}_{\mathbf{C_1}} \begin{bmatrix} i_L(t) \\ v_c(t) \end{bmatrix}_g \tag{29}$$

These equations can be written as:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A_1}\mathbf{x}(t) + \mathbf{B_1}u(t) \\ y(t) &= \mathbf{C_1}\mathbf{x}(t) \end{aligned} \tag{30}$$

where $\mathbf{x}(t) = \left[i_L(t)\ \ v_c(t)\right]^T$, $u(t) = V_g$ and $y(t) = v_o(t)$.

Similarly, the second state (switch off) is represented by:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A_2}\mathbf{x}(t) + \mathbf{B_2}u(t) \\ y(t) &= \mathbf{C_2}\mathbf{x}(t) \end{aligned} \tag{31}$$

where:

$$\mathrm{A}_2 = \begin{bmatrix} -\dfrac{RR_c}{(R+R_c)L} & -\dfrac{R}{(R+R_c)L} \\ \dfrac{R}{(R+R_c)C} & -\dfrac{1}{(R+R_c)C} \end{bmatrix}, \mathrm{B}_1 = \mathrm{B}_2 \text{ and } \mathrm{C}_2 = \begin{bmatrix} \dfrac{RR_c}{R+R_c} & \dfrac{R}{R+R_c} \end{bmatrix}$$

2) Averaged state-space model

Equations (30) and (31) can be combined in an averaged form to yield a single continuous-time representation of the state-space, as shown below:

$$\begin{aligned} \dot{\bar{\mathrm{x}}}(t) &= [d(t)\mathrm{A}_1 + d'(t)\mathrm{A}_2]\bar{\mathrm{x}}(t) + [d(t)\mathrm{B}_1 + d'(t)\mathrm{B}_2]\bar{u}(t) \\ \bar{\mathrm{y}}(t) &= [d(t)\mathrm{C}_1 + d'(t)\mathrm{C}_2]\bar{\mathrm{x}}(t) \end{aligned} \tag{32}$$

Alternatively, in a compact representation

$$\begin{aligned} \dot{\bar{\mathrm{x}}}(t) &= \mathrm{f}(\bar{\mathrm{x}}(t), d(t), \bar{u}(t)) \\ \bar{\mathrm{y}}(t) &= \mathrm{g}(\bar{\mathrm{x}}(t), d(t), \bar{u}(t)) \end{aligned} \tag{33}$$

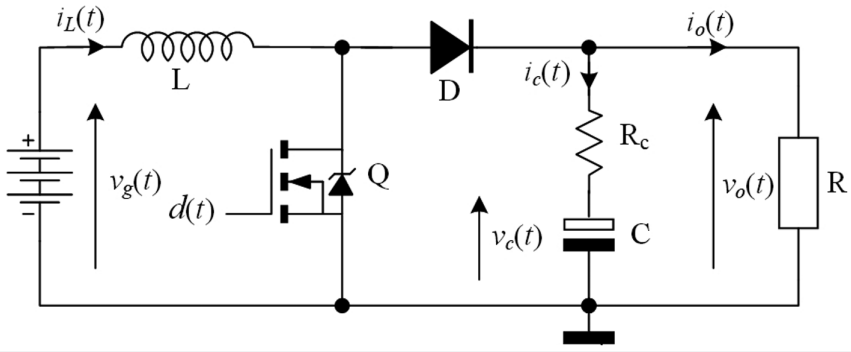

Fig. 3. Circuit of the boost converter.

where $d(t)$ represents the duty cycle, considered as the averaged control input, and $d'(t) = 1 - d(t)$.

The averaging process smooths out the high-frequency variations of the system, resulting in a time-invariant nonlinear model where the nonlinearity arises from the multiplication of the control signal $d(t)$ by the state variables.

3) Small signal linear model

The ultimate goal is to develop an equivalent linear model to analyze the system and design a GPC controller. The process begins by identifying the equilibrium or steady-state operating point, followed by linearizing the system around this operating condition.

At the equilibrium point (X, D, U), the state-space averaged model that describes the converter is:

$$\begin{aligned} 0 &= \mathrm{A}X + \mathrm{B}U \\ Y &= \mathrm{C}X \end{aligned} \tag{34}$$

where:

$$\begin{aligned} \mathrm{A} &= D\mathrm{A}_1 + D'\mathrm{A}_2 \\ \mathrm{B} &= D\mathrm{B}_1 + D'\mathrm{B}_2 \\ \mathrm{C} &= D\mathrm{C}_1 + D'\mathrm{C}_2 \end{aligned} \tag{35}$$

The uppercase letters $X = [I_L, V_c]^T$, D, $Y = V_o$ and $U = V_g$ denote the variables at the operating point. The dc state equations (34) therefore become

$$\begin{aligned} 0 &= -\frac{RR_cD'}{(R+R_c)L}I_L - \frac{RD'}{(R+R_c)L}V_C + \frac{1}{L}V_g \\ 0 &= \frac{RD'}{(R+R_c)C}I_L - \frac{1}{(R+R_c)C}V_C \\ V_o &= \frac{RR_cD'}{R+R_c}I_L + \frac{R}{R+R_c}V_C \end{aligned} \tag{36}$$

Given the values of V_g and V_o, (36) can be analytically solved to derive the values of I_L and D.

$$V_c = V_o = RD'I_L \Rightarrow I_L = \frac{V_o}{RD'} \tag{37}$$

$$V_0 = \frac{V_g}{1 - \frac{R}{R+R_c}D} \tag{38}$$

From which

$$D = \frac{R+R_c}{R} \cdot \frac{V_o - V_g}{V_o} \tag{39}$$

The numerical application based on the parameters of the boost converter gives us $I_L = 0.96A$ and $D = 0.502$. To facilitate the computation of the transfer function, we take $D = 0.5$; consequently, the value of the state vector X will be $[0.956A\ 23.9V]^T$.

The primary objective, however, is to create a corresponding linear model. To achieve this, we introduce perturbations to the averaged equation around the operating point.

$$\begin{aligned} \bar{\mathbf{x}}(t) &= \mathbf{X} + \check{\mathbf{x}}(t) \\ \bar{y}(t) &= Y + \check{y}(t) \\ d(t) &= D + \check{d}(t) \Rightarrow d'(t) = D' - \check{d}(t) \end{aligned} \tag{40}$$

We assume that the variation in the supply voltage is zero and $\check{d}(t)$ represents a small AC perturbation of the duty cycle. The vectors $\check{\mathbf{x}}(t)$ and $\check{y}(t)$ represent the resulting small AC perturbations in the state and output, respectively. These AC perturbations are assumed to be much smaller than their corresponding steady-state values.

The linearized (ac, small-signal) system can now be obtained from (33):

$$\begin{aligned} \dot{\check{\mathbf{x}}}(t) &= \mathbf{A_c}\check{\mathbf{x}}(t) + \mathbf{B_c}\check{d}(t) \\ \check{y}(t) &= \mathbf{C_c}\check{\mathbf{x}}(t) + \mathbf{E_c}\check{d}(t) \end{aligned} \tag{41}$$

where

$$\mathbf{A_c} = \left[\frac{\partial \mathbf{f}}{\partial \bar{\mathbf{x}}}\right]_{\substack{\bar{\mathbf{x}}(t)=\mathbf{X} \\ \bar{d}(t)=D}} = \mathbf{A} \tag{42}$$

$$\mathbf{B_c} = \left[\frac{\partial \mathbf{f}}{\partial \bar{d}}\right]_{\substack{\bar{\mathbf{x}}(t)=\mathbf{X} \\ \bar{d}(t)=D}} = \left(\mathbf{A}_1 - \mathbf{A}_2\right)\mathbf{X} + \underbrace{\left(\mathbf{B}_1 - \mathbf{B}_2\right)}_{0}U \tag{43}$$

$$\mathbf{C_c} = \left[\frac{\partial \mathbf{g}}{\partial \bar{\mathbf{x}}}\right]_{\substack{\bar{\mathbf{x}}(t)=\mathbf{x} \\ \bar{d}(t)=D}} = \mathbf{C} \tag{44}$$

$$\mathbf{E_c} = \left[\frac{\partial \mathbf{g}}{\partial \bar{d}}\right]_{\substack{\bar{\mathbf{x}}(t)=\mathbf{x} \\ \bar{d}(t)=D}} = \left(\mathbf{C}_1 - \mathbf{C}_2\right)\mathbf{X} \tag{45}$$

Assuming zero initial conditions, the Laplace transform of (41) is:

$$\begin{aligned} s\check{\mathbf{x}}(s) &= \mathbf{A_c}\check{\mathbf{x}}(s) + \mathbf{B_c}\check{d}(s) \\ \check{y}(s) &= \mathbf{C_c}\check{\mathbf{x}}(s) + \mathbf{E_c}\check{d}(s) \end{aligned} \tag{46}$$

The state vector is:

$$\begin{aligned} \check{\mathbf{x}}(s) &= (s\mathbf{I} - \mathbf{A_c})^{-1}\mathbf{B_c}\check{d}(s) \\ \check{y}(s) &= \mathbf{C_c}\check{\mathbf{x}}(s) + \mathbf{E_c}\check{d}(s) \end{aligned} \tag{47}$$

Given that $\check{y}(t) = \check{v}_o(t)$, the transfer function of the converter's output as a function of the duty cycle is:

$$H(s) = \frac{\check{v}_o(s)}{\check{d}(s)} = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B_c} + \mathbf{E_c} \tag{48}$$

Based on the parameters of the boost converter, the transfer function $H(s)$ is given by (49).

$$H(s) = 47.7\frac{\left(1 + \frac{s}{28409}\right)\left(1 - \frac{s}{29023}\right)}{0.7567 \cdot 10^{-6}s^2 + 0.1046 \cdot 10^{-3}s + 1} \tag{49}$$

The presence of a Right-Half Plane (RHP) zero in the model of a boost converter significantly affects its behavior in closed-loop control. The RHP zero introduces a non-minimum phase characteristic, which complicates the design of stable closed-loop controllers. This effect restricts the achievable bandwidth and necessitates precise tuning to prevent instability and excessive overshoot.

In digital predictive control, such as on a microcontroller, it is essential to incorporate a delay in the model to accurately capture the system's dynamics. This delay represents the influence of previous control inputs on the future state of the system. To determine the transfer function in the z-domain, MATLAB's toolbox was utilized, enabling the effective integration of this delay into the model.

$$H_W(z) = \frac{26.37z^{-1} + 26.37z^{-2}}{1 - 0.7668z^{-1} + 0.8709z^{-2}} \tag{50}$$

The subscript "*W*" refers to the white-box model.

D. Black-Box Model

Black-box modeling is an experimental approach that uses data to determine the system's transfer function. The identification test consists of superimposing a low-amplitude pseudo-random sequence (Pseudo Random Binary Sequence: PRBS) on the system's input. This sequence approximates zero-mean white noise without altering the system's operating point.

The experimental data collected from the identification test is processed by a recursive identification algorithm, thus providing good precision of the system parameters [35]. The amplitude of the pseudo-random sequence $\Delta PRBS[k]\Delta PRBS[k] = \pm 0.025$. The command entry: $d[k] = D_0 + \Delta PRBS[k]$.

Using the CARIMA model, the discrete transfer function for the boost converter can be described as:

$$H(z^{-1}) = \frac{B(z^{-1})}{A(z^{-1})} \tag{51}$$

with,

$$\begin{aligned} A(z^{-1}) &= 1 - 0.9411z^{-1} + 0.3266z^{-2} \\ B(z^{-1}) &= 0.7935z^{-1} + 0.00833z^{-2} \end{aligned} \tag{52}$$

where

$$H_B(z^{-1}) = \frac{0.7935z^{-1} + 0.00833z^{-2}}{1 - 0.9411z^{-1} + 0.3266z^{-2}} \tag{53}$$

The subscript "*B*" refers to the black-box model.

There are two objectives to study from these experiences. The first is to validate the implementation of the generalized predictive control (GPC) algorithm on a microcontroller. The second is to study the influence of certain parameters of predictive control, such as the need for memory consumption and the maximal response time of the generalized predictive control (GPC) algorithms.

E. Hardware Simulation

The simulation in MATLAB does not allow for an accurate estimation of the computation time of the Generalized Predictive Control (GPC) algorithm; even a Processor-In-the-Loop (PIL) simulation does not enable a precise estimation of the loop's computation time. Therefore, we opted for a purely hardware-based simulation; the code for the GPC control algorithm and the boost converter model is loaded onto the microcontroller. This on-chip simulation phase allows for a precise determination of the response time of the GPC algorithm for different values of the prediction and control horizons. The CoreSight debugger's capabilities in the ARM architecture, including the Instrumentation Trace Macrocell (ITM) module, facilitate the collection of system information for subsequent offline analysis (Data Logger). In all tests, the sampling period is set to 1 ms. The computational complexity of the control increment depends on the order of the boost model and the prediction horizon, while the control horizon only affects the offline part. The GPC algorithm performs correctly; each time the setpoint changes, the corresponding response reaches it without error, even in the presence of disturbances injected into the output at 0.08s and 0.22s. The dynamic response is shown in Fig. 4(a); the overshoots observed in the white-box model are due to the Right-Half Plane (RHP) zero.

For a comparative study, we implemented a PID controller on both models. The responses of the two models, as shown in Fig. 4(b), clearly illustrate the effect of the RHP zero on the dynamics of the white-box model.

The time required for the manipulated variable is studied. Fig. 4 represents the computation time of the control sequence as a function of the prediction horizon. This time is relatively short, i.e., 2.95µs for $H_p = 50$. Without anticipation, this computation time is reduced to

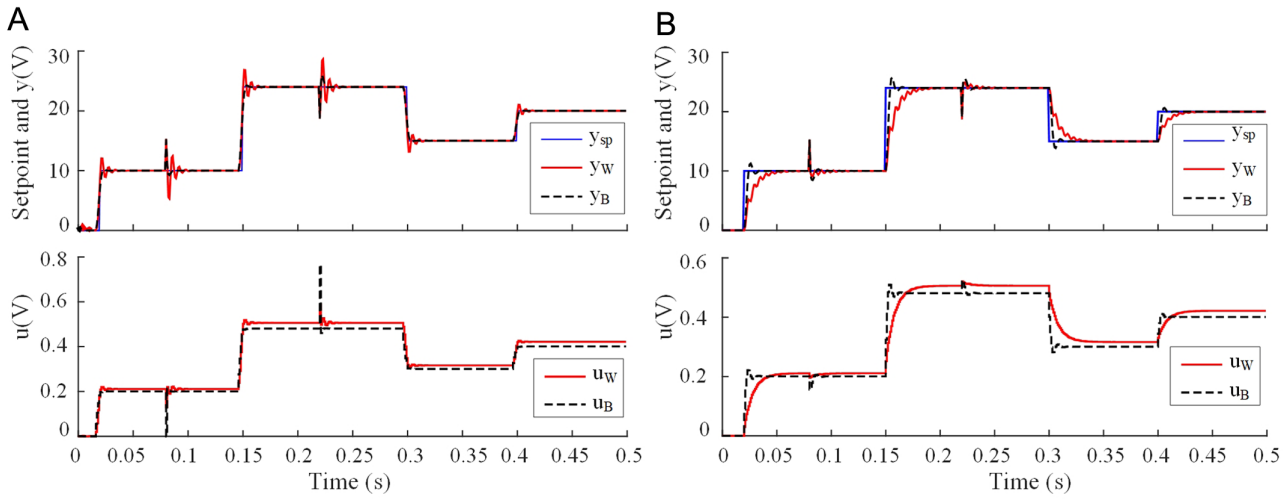

Fig. 4. Control performance with disturbances for both models: (a) GPC responses, (b) PID responses.

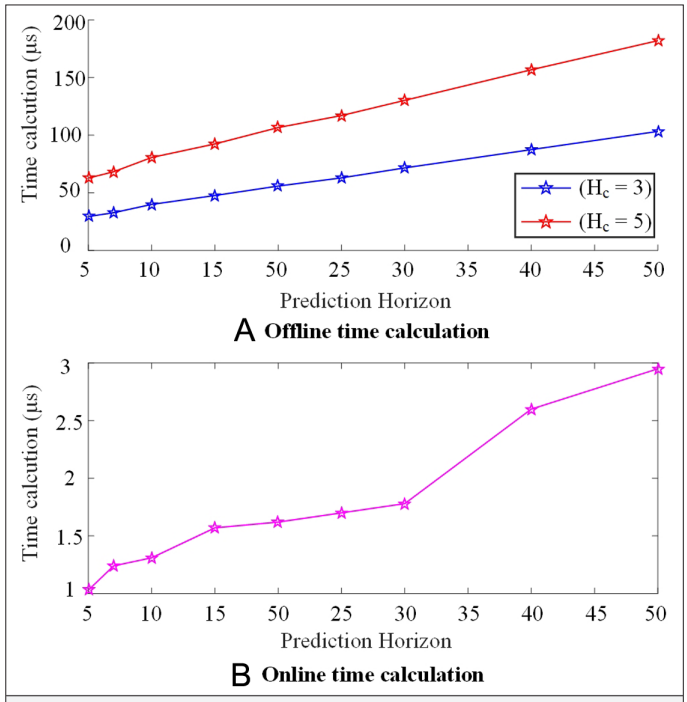

Fig. 5. Online and offline computational time requirement.

0.54µs independently of H_p. It is also important to study the memory occupation and the offline time calculation.

The results presented in Table 1 are a function of H_p and H_c. For example, for a long (H_p=5 and H_p=50) the memory allocation does not exceed 4Ko.

The calculation time does not exceed 20% of the sampling period. In fact, for example, in the case of H_p=50, the offline time calculation is evaluated as 180 µs, which is about 20% of the sampling period (200 µs). The remaining 80% can be useful for implementing more complex algorithms when the constraints on the control are considered. Fig. 5 illustrates the execution time of the offline part and the online part as a function of H_p for both cases of H_c=3 and H_c=5.

F. Hardware Real-Time Application

Hardware real-time application was carried out on the STM32 microcontroller for the boost converter across both models. The GPC algorithm performs correctly; each time the setpoint changes, the system response reaches the setpoint without error, even in the presence of disturbances injected into the output at 0.175 s and 0.325 s. The results for the two cases are illustrated in Fig. 6(a), while the performance of the PID regulator is shown in Fig. 6(b).

The evaluation of computation times yielded results of 4.64 µs for H_p=5 and 5.6 µs. These findings demonstrate the efficiency and effectiveness of the implemented control strategies in managing the boost converter's performance while maintaining low computational overhead.

Considering the study previously presented, we can arrive at the following remarks:

- It is like an optimization procedure; the new execution algorithm is very favorable.
- The computational load necessary for the process control is minimized by taking into account the analytical solution.
- The creation of the control is easily researched without recourse to any optimization.

TABLE 1. MEMORY ALLOCATION ALGORITHMS IN BYTE

HC/HP	3	5	7	10	15	20	25	30	40	50
1	188	252	316	412	572	732	892	1052	1372	1692
3	316	412	507	652	892	1132	1372	1612	2092	2572
5	-	636	468	956	1276	1596	1916	2236	2876	3516

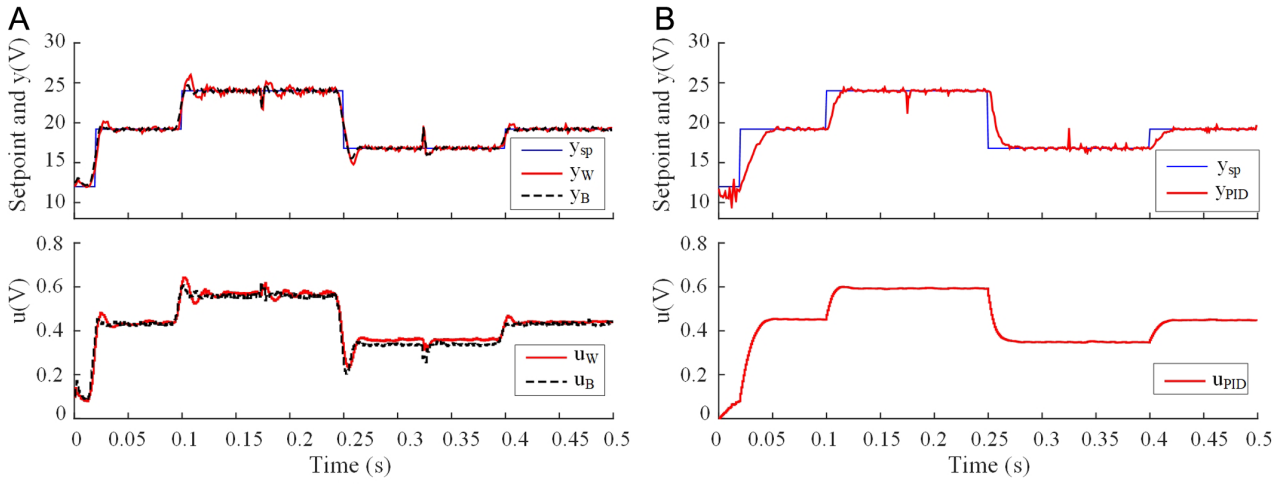

Fig. 6. Boost converter experimental control performance with disturbances (a) GPC response for $H_c = 1$, $Hp = 7$; (b) PID responses.

- The implementation time required for the solver and the process command are minimized.

V. CONCLUSION

This paper presented the implementation of a Model Predictive Control (MPC) strategy on a low-cost microcontroller, specifically the STM32F746, for a boost converter. Both white-box and black-box modeling approaches were utilized to represent the converter's dynamics. The white-box model, developed using an average state-space approach, provided an analytical description of the system, while the black-box model was identified using a Pseudo-Random Binary Sequence (PRBS).

On-chip simulations were conducted with both the GPC and a PID controller applied to the two models. The results indicated that the black-box model exhibited greater stability in both control scenarios, while the predictive control anticipated system behavior more effectively, leading to a faster response. The overshoots observed in the averaged model were due to the presence of a Right-Half Plane (RHP) zero, which restricted the tuning options for the PID controller and resulted in slower dynamics compared to the black-box model.

Furthermore, the computational requirements for the MPC algorithm were analyzed. The results demonstrated that the algorithm's execution time remained well below 20% of the sampling period, and the memory footprint occupied less than 2% of the total available data memory. These findings suggest that it is feasible to implement more complex control algorithms, such as nonlinear predictive control, on the same platform.

Experimental validation confirmed the simulation results, demonstrating the effectiveness of the proposed MPC strategy.

Availability of Data and Materials: The data that support the findings of this study are available on request from the corresponding author.

Peer-review: Externally peer-reviewed.

Author Contributions: Concept – A.H., H.N.; Design – A.H., H.N.; Supervision – F.N.; Resources – A.H.; Materials – A.H.; Data Collection and/or Processing – A.H. Analysis and/or Interpretation – H.N., A.H.; Literature Search – H.N., A.H.; Writing – H.N., A.H.; Critical Review – H.N.

Declaration of Interests: The authors have no conflict of interest to declare.

Funding: The authors declare that this study has received no financial support.

REFERENCES

1. S. Tahir, J. Wang, M. H. Baloch, et G. S. Kaloi, "Digital control techniques based on voltage source inverters in renewable energy applications: A review," *Electronics*, vol. 7, no. 2, Art. no. 2, févr. 2018. [CrossRef]
2. J. M. Maciejowski, *Predictive Control: With Constraints*. Englewood Cliffs, United States of America: Prentice Hall, 2002.
3. P. Tatjewski, *Advanced Control of Industrial Processes*. London: Springer, 2007. [CrossRef]
4. D. W. Clarke, C. Mohtadi, et P. S. Tuffs, "Generalized predictive control—Part I. The basic algorithm," *Automatica*, vol. 23, no. 2, pp. 137–148, 1987. [CrossRef]
5. H. J. Ferreau, H. G. Bock, et M. Diehl, "An online active set strategy for fast parametric quadratic programming in MPC applications," in *NMPC-FS'06-IFAC Workshop on Nonlinear Model Predictive Control for Fast Systems, Date*, France: Grenoble, 2006, pp. 13–22.
6. G. Banjac, B. Stellato, N. Moehle, P. Goulart, A. Bemporad, et S. Boyd, "Embedded code generation using the OSQP solver," in 56th Annual Conference on Decision and Control. (CDC). IEEE PUBLICATIONS. Melbourne: VIC. Australia, 2017, pp. 1906–1911. [CrossRef]
7. J. Mattingley, and S. Boyd, "CVXGEN: A code generator for embedded convex optimization," *Optim. Eng.*, vol. 13, no. 1, 1–27, 2012. [CrossRef]
8. S. J. Qin, AND T. A. Badgwell, "A survey of industrial model predictive control technology" *Control Eng. Pract.*, vol. 11, no. 7, pp. 733–764, 2003. [CrossRef]
9. S. Plamowski, "Model predictive control of a dynamic system with fast and slow dynamics: Implementation Using PLC," in *Advanced*. Cham, 2020, pp. 1104–1115. [CrossRef]
10. L. Cheng *et al.*, "Model predictive control for DC–DC boost converters with reduced-prediction horizon and constant switching frequency," *IEEE Trans. Power Electron.*, vol. 33, no. 10, pp. 9064–9075, 2018. [CrossRef]
11. M. Gulan, G. Takács, N. A. Nguyen, S. Olaru, P. Rodriguez-Ayerbe, et B. Rohaľ-Ilkiv, "Embedded linear model predictive control for 8-bit microcontrollers via convex lifting," *IFAC PapersOnLine*, vol. 50, no. 1, pp. 10697–10704, 2017. [CrossRef]
12. M. B. Kane, J. Scruggs, et J. P. Lynch, "Model-predictive control techniques for hydronic systems implemented on wireless sensor and actuator networks," in American Control. Conference, Portland, OR, USA, 2014, 2014, pp. 3542–3547. [CrossRef]
13. F. Xu, H. Chen, X. Gong, et Q. Mei, "Fast nonlinear model predictive control on FPGA using particle swarm optimization," *IEEE Trans. Ind. Electron.*, vol. 63, no. 1, pp. 310–321, 2016. [CrossRef]

14. G. Takács, P. Zometa, R. Findeisen, and B. Rohal'-Ilkiv, in European Control. Conference (ECC), Aalborg, Denmark, 2016, pp. 1334–1340. [CrossRef]
15. M. Schwenzer, M. Ay, T. Bergs, et D. Abel, "Review on model predictive control: An engineering perspective," *Int. J. Adv. Manuf. Technol.*, vol. 117, no. 5–6, pp. 1327–1349, 2021. [CrossRef]
16. C. Jugade, D. Ingole, D. Sonawane, M. Kvasnica, J. Gustafson, "A memory-efficient explicit model predictive control using posits," in Sixth Indian Control. Conference (ICC), 2019, pp. 188–193. [CrossRef]
17. D. Ingole, M. Kvasnica, H. De Silva, et J. Gustafson, "Reducing memory footprints in explicit model predictive control using universal numbers," *IFAC PapersOnLine*, vol. 50, no. 1, pp. 11595–11600, 2017. [CrossRef]
18. M. Kvasnica, P. Bakaráč, et M. Klaučo, "Complexity reduction in explicit MPC: A reachability approach," *Syst. Control Lett.*, vol. 124, 19–26, 2019. [CrossRef]
19. A. A. Kheriji, F. Bouani, M. Ksouri, et M. B. Ahmed, "A microcontroller implementation of model predictive control," *Int. J. Electr. Inf. Eng.*, vol. 5, no. 5, pp. 600–606, 2011. [CrossRef]
20. R. Kouki, H. Salhi, et F. Bouani, "Application of model predictive control for a thermal process using STM32 microcontroller," in International Conference on Control., Automation and Diagnosis (ICCAD), Hammamet, Tunisia, 2017, 146–151. [CrossRef]
21. X. Jiao, X. Zhu, M. Xu, and Liming Di, "A Simplified Dynamic Matrix Control for application of embedded real-time MPC," in International Conference on Information Science and Technology, Nanjing, 2011, pp. 1062–1065. [CrossRef]
22. Z. Chen, L. Yu, et E. Dong, "A fast generalized predictive control algorithm based on Toeplitz Matrix," in International Conference on Computational Intelligence and Software Engineering, vol. 1–4, 2010. [CrossRef]
23. C. Schreppel, and J. Brembeck, "A QP Solver Implementation for Embedded Systems Applied to Control Allocation" *Computation*, vol. 8, no. 4, 2020. [CrossRef]
24. A. Wojtulewicz, and M. Lawrynczuk, in 23rd International Conference on Methods & Models in Automation & Robotics (MMAR), 2018, pp. 579–584. [CrossRef]
25. A. Wojtulewicz, and M. Ławryńczuk, "Implementation of Multiple-Input Multiple-Output Dynamic Matrix Control Algorithm for Fast Processes Using Field Programmable Gate Array" *IFAC PapersOnLine*, vol. 51, no. 6, pp. 324–329, 2018. [CrossRef]
26. Z. Karami, Q. Shafiee, S. Sahoo, M. Yaribeygi, H. Bevrani, et T. Dragicevic, "Hybrid model predictive control of DC–DC boost converters with constant power load," *IEEE Trans. Energy Convers.*, vol. 36, no. 2, pp. 1347–1356, 2021. [CrossRef]
27. Q. Guo, I. Bahri, D. Diallo, et E. Berthelot, "Model predictive control and linear control of DC–DC boost converter in low voltage DC microgrid: An experimental comparative study," *Control Eng. Pract.*, vol. 131, p. 105387, 2023. [CrossRef]
28. R. Niu, H. Zhang, et J. Song, "Model predictive control of DC–DC boost converter based on generalized proportional integral observer," *Energies*, vol. 16, no. 3, 2023. [CrossRef]
29. M. Ehsani, M. Saeidi, H. Radmanesh, et A. Abrishamifar, "Comparisons between generalized predictive control and linear controllers in multi-input DC-DC boost converter," *Int. J. Ind. Electron. Control. Optim.*, vol. 3, no. 1, pp. 27–34, 2020. [CrossRef]
30. J. A. González-Castro *et al.*, "Low-cost platform implementation of discrete controllers for DC-DC boost converter," *Energies*, vol. 17, no. 16, 2024. [CrossRef]
31. A. Marahatta, Y. Rajbhandari, A. Shrestha, S. Phuyal, A. Thapa, et P. Korba, "Model predictive control of DC/DC boost converter with reinforcement learning," *Heliyon*, vol. 8, no. 11, p. e11416, 2022. [CrossRef]
32. P. Chaber, and M. Lawrynczuk, "Fast Analytical Model Predictive Controllers and Their Implementation for STM32 ARM Microcontroller", *IEEE Trans. Ind. Inf.*, vol. 15, no. 8, pp. 4580–4590, 2019. [CrossRef]
33. P. Almeida, K. M. Monteiro, A. A. Ferreira, V. F. Montagner, et P. G. Barbosa, "Revisiting boost converter modeling and analysis to integrate power electronics and control courses," *Tech.Rxiv*, 2024. [CrossRef]
34. A. Choubey, P. K. Padhy, et S. K. Jain, "Model Predictive control for DC-DC Boost converter," in IEEE Conference on Energy Conversion (CENCON), Johor Bahru, Malaysia, 2021, pp. 58–63. [CrossRef]
35. A. J. Forsyth, AND S. V. Mollov, "Modelling and control of DC-DC converters" *Power Eng. J.*, vol. 12, no. 5, pp. 229–236, 1998. [CrossRef]

Ali Hmidene received his Master of Science and DEA from ENSET, Tunis, in 1987 and 1992, respectively. He also obtained his CESS and Agrégation degrees in Electrical Engineering from ESSTT, Tunisia, in 1997 and 2000. Currently, he is working at the Higher Institute of Technological Studies (ISET) in Sousse, Tunisia. His main research interests include embedded control systems and fast model predictive control.

Ben Nasr Hichem received his Master of Science and CESS from ENSET, Tunis, Tunisia, in 1994 and 1996, respectively. In 1998, he obtained his Agregation Degree in Electrical Engineering from ESSTT, Tunis, Tunisia. In 2009, he obtained his Doctorate Degree in Electrical Engineering from ENIS, Tunisia. His current research interests include Modeling, Identification, and Predictive Control of Linear and Nonlinear Fast Dynamics Systems. He has published over 15 technical papers and is a co-author of a book 'Predictive Control' ISBN 978-953-307-168-8, June 2011.

Faouzi M'Sahli received his Master of Science and DEA from ENSET, Tunis, Tunisia, in 1987 and 1989, respectively. In 1995, he obtained his Doctorate Degree in Electrical Engineering from ENIT, Tunisia. He is currently a Professor of Electrical Engineering at the National School of Engineers, Monastir, Tunisia. His research interests include Modeling, Identification, Predictive, and Adaptive Control Systems. He has published over 100 technical papers and is a co-author of a book "Identification et commande numerique des procédés industriels," Technip editions, Paris, France.