Implementation of online Model Predictive Control on a Programmable Logic Controller for controlling a pilot-scale distillation column

Bart Huyck, Jos De Brabanter, Bart De Moor, Jan Van Impe and Filip Logist

***Abstract*— As Programmable Logic Controllers (PLCs) have become more powerful, their use as a host for an online Model Predictive Controller (MPC) can be considered. In this paper, an online MPC controller has been implemented on a PLC. The resulting controller has been successfully tested for the tracking of reference temperatures in a pilot-scale binary distillation column. To this end, hardware-in-the-loop tests have been performed, before performing real experiments with the actual column. Two algorithms, i.e., the Hildreth and the qpOASES algorithm, have been used to solve online the Quadratic Programs (QPs) appearing in the MPC. It turns out that both algorithms can be hosted on a PLC. However, only the classic Hildreth algorithm has been found to be able to handle sufficiently large QPs (as the ones needed for the distillation column control), while allowing a relatively easy translation to a standard PLC programming language.**

I. INTRODUCTION

The increasing computing power of industrial controllers and embedded devices as well as algorithmic developments to increase computation speed, open new perspectives for the use of Advanced Process Control strategies in industrial applications. These strategies can be used, e.g., to reduce process and/or product variations or to improve the operational profit. Model Predictive Control (MPC) is an example of such an Advanced Process Control strategy. To successfully employ an MPC, a number of requirements have to be fulfilled: (i) there has to be a need for such algorithms due to, e.g., economic or ecological demands, (ii) knowledge about the employed techniques has to be present, (iii) a model of the process has to be easily obtainable, and (iv) a suitable control device that is capable of hosting the advanced control algorithm is available.

This paper focusses on the device used for running the MPC algorithm and it is assumed that all other requirements are fulfilled. For a new installation, a powerful device can be selected that has sufficient computing power, whereas in an existing installation that has to be upgraded, control devices are not always replaced. Moreover, requirements with respect to robustness and reliability in harsh industrial environments render the use of the latest and fastest standard PCs impossible and favor the exploitation of robust but slower industry standard control devices.

Bart Huyck, Filip Logist and Jan Van Impe are with BioTeC, Department of Chemical Engineering, Katholieke Universiteit Leuven, W. de Croylaan 46, B-3001 Leuven, Belgium. jan.vanimpe@cit.kuleuven.be

Bart Huyck, Jos De Brabanter and Bart De Moor are with ESAT-STADIUS, Department of Electrical Engineering, Katholieke Universiteit Leuven, kasteelpark Arenberg, B-3001 Leuven, Belgium

The main job in any MPC controller relates to the solution of Quadratic Programs (QPs). To quickly solve these QPs, two directions have been followed. One way is to solve the problem offline and employ a look-up table to retrieve the desired solution online [1], [9], [14], [15]. An alternative way involves the use of fast QP solvers online. For low-level controllers and embedded devices, the latter approach is rather recent (e.g. [8], [13]). However, the development of dedicated software and code generators (e.g., [3], [11]) lately is intended to facilitate and stimulate the use of online MPC algorithms on these devices.

Based on successful online MPC implementations using Programmable Automation Controllers (PACs) [5], [7], the current paper investigates the use of the widely spread, robust Programmable Logic Controllers (PLCs), which typically have less computing power than PACs for online MPC. To explore the limits of performance of these devices a large-scale real-life case study is adopted, i.e., the temperature control of a pilot-scale distillation column. In view of safety all control algorithms and hardware are first tested offline through hardware-in-the-loop tests. When these tests have been found to be successful, the transfer to the real distillation column is made in view of real life experiments. In brief, the paper's main contribution is the successful implementation of online MPC on a PLC for the control of a real-life large-scale experimental set-up.

The paper is organized as follows. Section II describes the employed MPC formulation and the selected online QP solvers. Section III discusses the implementation of the MPC controller on the PLC. In Section IV the pilot-scale binary distillation column is described together with its model. The obtained experimental results are presented in Section V. Finally, Section VI summarizes the main conclusions.

II. MODEL PREDICTIVE CONTROL

Linear MPC is well known in literature [2], [10], [16] and the reader is invited to read these works for a detailed description. The basic formulation used to control the pilot-scale distillation column is briefly described below.

A. Model predictive control formulation

A linear, time invariant discrete-time system is given by:

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

with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ and $C \in \mathbb{R}^{p\times n}$. Here m, n and p are the number of inputs $\mathbf{u}$, states $\mathbf{x}$ and outputs $\mathbf{y}$, respectively. The objective of the controller is to find the optimal input for this system by means of minimizing a cost function:

$$\begin{aligned} J &= \sum_{i=1}^{H_p} \|\hat{\mathbf{y}}(k+i|k) - \mathbf{y}_{\text{ref}}(k+i|k)\|^2_{W_y} \\ &+ \sum_{j=0}^{H_c-1} \|\Delta\mathbf{u}(k+j|k)\|^2_{W_u}, \end{aligned} \tag{2}$$

with $\mathbf{y}_{ref}$ the output reference and $\hat{\mathbf{y}}$ the predicted output. The formulation $\mathbf{y}(k+i|k)$ represents the vector $\mathbf{y}$ on sample time $k+i$ at calculation time k. The change of the input is $\Delta\mathbf{u}(k+j|k) = \mathbf{u}(k+j|k) - \mathbf{u}(k+j-1|k)$. H_p and H_c (with $H_c \leq H_p$) are the prediction and control horizon of the controller, respectively. $W_y \in \mathbb{R}^{p\times p}$ and $W_u \in \mathbb{R}^{m\times m}$ are positive definite weight matrices.

One of the key elements of MPC is the possibility to handle constraints. For this paper only input constraints are taken into account:

$$\begin{aligned} \mathbf{u}(j) &\leq \mathbf{u}_{\text{Max}} \\ \mathbf{u}(j) &\geq \mathbf{u}_{\text{Min}}. \end{aligned} \tag{3}$$

Output constraints are omitted in the current study, but can be introduced. The optimization problem can be formulated as the minimization of Eq. (2), subject to Eq. (1) and (3). In order to solve this problem, the optimization problem is reformulated by elimination of the states in the form of a QP:

$$\min_{\theta} J = \frac{1}{2}\theta^T H\theta + g^T\theta \tag{4}$$

$$\text{subject to :} \quad P\theta \leq \alpha \tag{5}$$

with Eq. (4) the quadratic objective function, Eq. (5) the linear inequality constraints and θ the decision variables. To reformulate the problem, the following steps are taken. First, the state-space model Eq. (1) is rewritten in terms of the augmented state:

$$\xi = \begin{bmatrix} \Delta\mathbf{x} \\ \mathbf{y} \end{bmatrix} \tag{6}$$

with $\Delta\mathbf{x}(k) = \mathbf{x}(k) - \mathbf{x}(k-1)$ and $\mathbf{y}$ the currently measured output [16]. The prediction over the prediction horizon is written in matrix formulation and is formulated as:

$$\hat{\mathbf{Y}} = F\xi + Q\Delta\mathbf{U}, \tag{7}$$

with $\hat{\mathbf{Y}} \in \mathbb{R}^{pH_p\times 1}$ and $\Delta\mathbf{U} \in \mathbb{R}^{mH_c\times 1}$ column vectors of predicted outputs and delta inputs, respectively. $\Delta\mathbf{U}$ is composed of $\Delta\mathbf{u}(k|k)$ to $\Delta\mathbf{u}(k+H_c-1|k)$. The matrices F and Q can be found in many works on MPC [10], [16]. The matrix Q is post-processed by including the weight matrices resulting in the hessian $H \in \mathbb{R}^{mH_c\times mH_c}$ of the objective function in Eq. (4):

$$H = Q^T W_{y_{bd}} Q + W_{u_{bd}} \tag{8}$$

with $W_{y_{bd}}$ and $W_{u_{bd}}$ block diagonal matrices containing H_p and H_c times the matrices W_y and W_u, respectively. To minimize the online calculations, matrices that can be computed in advance, are calculated offline. The hessian matrix is one of the precomputed matrices as it does not change during runtime. The gradient vector g from Eq. (4) has to be calculated online. It contains three parts depending on the current state and the reference of the in- and outputs. As a result, g is calculated as:

$$g = G_1\xi - G_2\mathbf{Y}_{\text{ref}} - G_3\mathbf{U}_{\mathbf{ref}} \tag{9}$$

where $G_1 \in \mathbb{R}^{mH_c\times n}$, $G_2 \in \mathbb{R}^{mH_c\times pH_p}$ and $G_3 \in \mathbb{R}^{mH_c\times mH_c}$ are helper matrices to calculate the gradient online. $\mathbf{Y}_{\text{ref}}$ is an $\mathbb{R}^{pH_p\times 1}$ matrix of the references $\mathbf{y}_{\text{ref}}(k|k)$ to $\mathbf{y}_{\text{ref}}(k+H_p|k)$. The matrices G_1, G_2 and G_3 are constant and are computed offline:

$$G_1 = H^T W_{y_{bd}}^T F \tag{10}$$

$$G_2 = H^T W_{y_{bd}}^T \tag{11}$$

$$G_3 = W_{u_{bd}}^T \tag{12}$$

The constraints, i.e., the minimum and maximum admissible values for $\Delta\mathbf{U}$, are calculated online. $\Delta\mathbf{U}_{\text{Max}}$ is a column matrix of $\Delta\mathbf{u}(k+i|k)_{\text{Max}} = \mathbf{u}_{\text{Max}} - \mathbf{u}(k-1), i = 0\ldots H_c-1$. Similar, $\Delta\mathbf{U}_{\text{Min}}$ is a column matrix of $\Delta\mathbf{u}(k+1|k)_{\text{Min}} = \mathbf{u}_{\text{Min}} - \mathbf{u}(k-1), i = 0\ldots H_c-1$. Finally, being C_1 a lower triangular matrix of the appropriate identity matrices to convert $\mathbf{u}$ to $\Delta\mathbf{u}$, the QP problem to be solved yields:

$$\min_{\Delta\mathbf{U}} \quad \tfrac{1}{2}\Delta\mathbf{U}^T H\Delta\mathbf{U} + g^T\Delta\mathbf{U} \tag{13}$$

$$\text{subject to :} \quad \begin{bmatrix} C_1 \\ -C_1 \end{bmatrix}\Delta\mathbf{U} \leq \begin{bmatrix} \Delta\mathbf{U}_{\text{Max}} \\ -\Delta\mathbf{U}_{\text{Min}} \end{bmatrix} \tag{14}$$

To solve this QP problem, the Hildreth algorithm [4] and qpOASES [3] are used.

B. The Hildreth QP algorithm

This classic algorithm has been selected for its easy implementation. The implementation as presented in [16] has been employed. One the advantages of this approach is that the solution of the QP is based on an element-by-element search, which can be easily implemented if no extended mathematical support is available on a device. In case of conflicting constraints, the algorithm will give a compromised, near-optimal solution. The maximum number of allowed iterations is 250 and the stop criterium is set to 10^{-8}. As the source code has to be translated to the S7-SCL language, which is used for programming Siemens PLCs following the IEC 61131-3 standard, an easy to implement algorithm is indispensable.

C. qpOASES QP algorithm

qpOASES is an open-source C++ implementation of a recently proposed online active-set strategy [3]. This QP solver is intended to deliver the solutions of the QP fast, which makes it perfectly suited for fast MPC. qpOASES builds on the idea that the optimal sets of active constraints do not differ much from one QP to the next. At each

sampling instant, it starts from the optimal solution of the previous QP and follows a homotopy path towards the solution of the current QP. Along this path, constraints may become active or inactive as in any active-set QP solver and the internal matrix factorizations are adapted accordingly. While moving along the homotopy path, the online active-set strategy delivers sub-optimal solutions in a transparent way. Therefore, such sub-optimal feedback can be reasonably passed to the process in case the maximum number of iterations is reached.

A simplified version of qpOASES has been translated to S7-SCL. Note that the simplified implementation does not allow for hot starting the QP solution and is not fully optimized for computational speed. However, the current implementation enables starting the search for a solution with offline computed matrices. Furthermore, based on the knowledge that a solution is found in one step if no constraints are active, which is expected at many time instances, cold starting of the algorithm is not an issue.

III. Controller implementation

The experimental set-up employed in this paper is a pilot-scale distillation column. The actuators and sensors are connected to a Compact Fieldpoint (National Instruments, Austin) with a cFP-2020 controller interface and I/O modules cFP-RTD-124, cFP-AIO-610, cFP-AIO-600 and cFP-AI-110. Due to the lack of computing power, this Compact Fieldpoint is connected to a PC to run a supervisory program. This LabVIEW program contains the PI-controllers, the human-machine interface (HMI), the data logging and the visualization. This program runs at a rate of 10 Hz.

As PLC, a Siemens CPU319-3DP/PN is used to host the MPC controller and state estimator. The Siemens CPU is programmed using the Step 7 Professional 2010 software. To code the problem, the *Structured Control Language* (S7-SCL) is used. This programming language corresponds to Structured Text (ST) in the standard IEC 61131-3. The information exchange between the PLC and the PC is performed by means of an OPC server (NI OPC server 2012, National Instruments).

The base memory of this CPU is increased to the maximum allowed 8 MB. This CPU is the fastest Siemens S7-300 CPU and it takes 40 ns for one floating-point operation [12]. The size of functions (in this case called *function blocks*) programmed in the PLC are maximum 64 kB large. Hence, this strongly limits the number and the size of the variables that are used in one function. Moreover, arrays and matrices cannot be initialized with more than 255 unique elements at compilation time. To overcome this limitation, these arrays and matrices can be filled during runtime by, for example at start-up, by copying the elements of different small matrices of at maximum 255 elements into an array/matrix containing more than 255 elements. However, this approach is cumbersome and error-prone due to the need of absolute addressing. In summary, it can be expected that not the total available memory will be the bottleneck, but the memory that can be allocated to a single function. One function is allowed to run for a pre-specified *scan cycle monitoring time*, which cannot be set to larger values than 5999 ms. Exceeding this time results in a hardware interrupt and a restart of the device.

IV. Distillation Column Set-Up

A. Description

The experimental set-up involves a packed distillation column (see Fig. 1 and 2). The column is about 6 m high and has an internal diameter of 6 cm. The column works under atmospheric conditions and contains three sections of about 1.5 m with Sulzer CY packing (Sulzer, Winterthur) responsible for the separation. This packing has a contact surface of 700 m^2/m^3 and each meter packing is equivalent to 3 theoretical trays. The feed stream containing a mixture of methanol and isopropanol is fed into the column between packed sections 2 and 3. The temperature of the feed can be adjusted by an electric heater which can deliver heat up to a maximum of 250 W. At the bottom of the column a reboiler is present containing two electric heaters, each of maximum 3000 W. In the reboiler, a part of the liquid is vaporized while the rest is extracted as bottom stream. At the top of the column, a total condenser allows the condensing of the entire overhead vapor stream, which is then collected in a reflux drum. A part of the condensed liquid is fed back to the column as reflux, while the remainder leaves the column as the distillate stream.

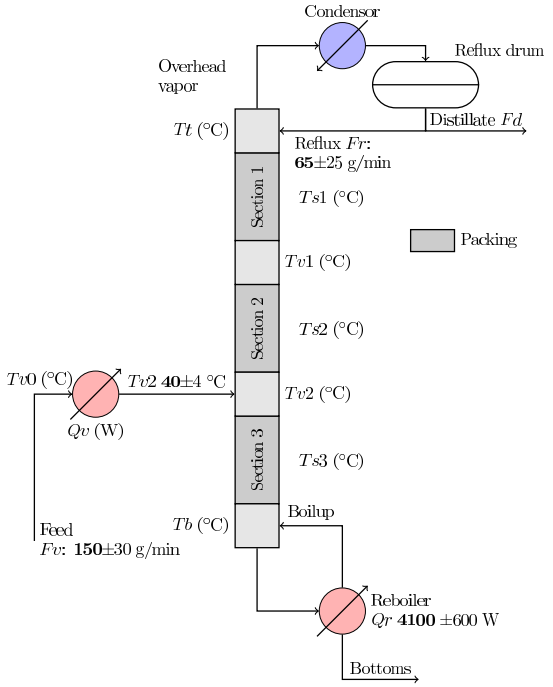

Fig. 1. Diagram of the pilot-scale distillation column. Nominal set-points are printed in bold and are followed by the maximum admissible deviations.

In this set-up the following four variables can be manipulated: the reboiler duty Qr (W), the feed rate Fv (g/min), the duty of the feed heater Qv (W) and the reflux flow rate Fr (g/min). The distillate flow Fd (g/min) is adjusted to maintain a constant reflux drum level. Measurements are available for the reflux flow rate Fr, the distillate flow rate Fd, the feed flow rate Fv and nine temperatures: the temperature at the top of the column Tt, the temperatures in the center of every packing section (i.e., $Ts1$, $Ts2$ and $Ts3$, respectively), the temperature $Tv1$ between section 1 and 2, the temperature $Tv2$ between section 2 and 3, the temperature Tb in the reboiler of the column, and the temperatures of the feed before and after heating (i.e., $Tv0$ and $Tv2$, respectively). All temperatures are measured in degrees Celsius.

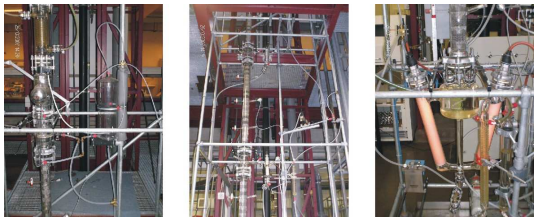

Fig. 2. Pictures of the pilot-scale distillation column: condenser (left), packed section and feed introduction (center), and reboiler (right).

There is no online measurement of the concentrations in the distillate and bottom stream. These can be measured off-line based on a their refractive index using a refractometer. However, since the concentrations for the system under study can easily be inferred from the temperatures when temperature and pressure are known, only a controller for the temperatures has to be implemented.

B. Model Identification

For linear MPC, a linear process model is needed. To describe the behavior of the column, a model has to be created that links the four manipulated variables, i.e., the feed flow rate (Fv), feed duty ($Tv2$), the reboiler duty (Qr) and the reflux flow rate (Fr) with two temperatures, i.e., the top temperature (Tt) and the reboiler temperature (Tb). Although the concentration of the end product is actually a desired controller variable, this can easily be inferred from the temperatures when the pressure is known. Hence, a *Multiple Input, Multiple Output* (MIMO) model that controls the reboiler and top temperature will be implemented. More information on the excitation experiments and the obtained model can be found in [5] and [6].

The obtained discrete state-space model has 13 states and is controllable, observable and stable. The validation results presented in [6] have demonstrated that the model for the top temperature is less accurate than the model for the reboiler temperature. Nevertheless, both models are believed to be of sufficient quality to be used in a model based controller.

C. Model predictive control parameters

In the above formulation, the control horizon is set to 10 and the prediction horizon is 50. This leads to a hessian of size 40×40 in Eq. (13). The matrices G_1, G_2 and G_3 are of size 40×15, 100×40 and 40×40, respectively. Multiplying the number of elements by four, gives an idea of the memory consumption of these stored matrices in the device in bytes. The weight matrix W_y is a diagonal matrix with elements $W_{y_{11}} = 1$ and $W_{y_{22}} = 0.9$. This punishes each deviation from the top temperature reference slightly more than a deviation from the reboiler temperature. The weight matrix W_y has four elements on the diagonal $W_{u_{11}} = 0.8$, $W_{u_{22}} = 1$, $W_{u_{33}} = 0.8$ and $W_{u_{44}} = 1$ for the feed flow rate, the feed duty, reboiler duty and reflux flow rate, respectively. All non-diagonal elements are zero.

V. Model predictive control on a PLC

This section presents the results for MPC experiments hosted by a PLC to control the pilot-scale distillation column. First, hardware-in-the-loop (HIL) experiments are carried out. Next, real experiments on a pilot-scale experimental set-up (EXP) are performed. The focus in these experiments is the employed memory and time required to solve QP problem.

A. Hardware-in-the-loop experiments

Hardware-in-the-loop experiments consist of simulation experiments with the hard- and software that will be used for experiments on the set-up. The real set-up itself, however, is replaced by a linear model. A shift of +0.3 and -0.3^{o}C for the top and reboiler temperature references, respectively, is applied to make the control more challenging. This shift has been applied to make sure the input constraints are touched during the HIL experiments.

TABLE I
THE TOTAL, MEAN, MINIMUM AND MAXIMUM RUN TIMES FOR ONE MPC ITERATION ON THE PLC.

	run-time (ms)			
	total	mean	min.	max.
Hildreth (HIL)	70956	212.4	15	556
qpOASES (HIL)	11780	1308.8	1104	2026
Hildreth (EXP)	31123	91.0	14	532

The results of these HIL experiments are presented in Table I. For the Hildreth algorithm, the number of iterations is at maximum 35. The maximum required time to solve the QP is around 556 ms for this number of iterations. The minimum run time, corresponding to 1 iteration is 14 ms. Worthwhile to mention is that in trials previous to this experiment, small mistakes caused the QP solver to reach its maximum number of iterations. For the HIL experiments on PLC, this maximum is set to 250 iterations. This corresponds to a calculation time of 3970 ms. The scan cycle monitoring time was set to 4000 ms, which is higher than the maximum required calculation time. On the pilot-scale set-up, an MPC update cycle of 1 minute will be used. The calculation times to solve the QP are far

below this MPC cycle time. The calculation time of the QP is therefore not a bottleneck for the implementation of MPC on a PLC for the pilot-scale set-up with the Hildreth algorithm.

Programming issues on the employed PLC, e.g., the maximum amount of 64 kB for each function block and the difficult initialization of arrays larger than 255 elements are much harder to take into account. While programming the QP solver, the code is split up in functions. For the Hildreth algorithm the solution of the QP is fit into one function block. This function requires to store five times a matrix of the size of the hessian. Based on the programming issues and the required memory, a practical limit for a QP to be solved with the current implementation of the Hildreth algorithm, is a hessian with at most 64×64 elements. As mentioned, it is possible to increase that size by spreading the matrix data in different function blocks, but such an approach is cumbersome. However, the total available memory is far from filled. According to the online memory status information, the current implementation needs only 4% of the 8 MB available Random Access Memory, 16% of the 1.4 MB working memory and 26% of the 0.7 MB retentive data memory. Hence, there is enough space to add additional control programs.

A similar experiment with the qpOASES algorithm ended abruptly after 8 time steps with a system failure. According to the diagnostic registers, the error was caused by a *scan cycle monitoring time* violation, which means the function block needed more than 5999 ms. When taking a margin of about 10 percent, one iteration needs between 350 and 450 ms. According to a Matlab simulation, up to 25 iterations are to be expected for the reference trajectory and settings as employed in this HIL experiment. This is at least 8.75 s and thus above the scan cycle monitoring time. The presented time values in Table I are calculated on the first 7 time steps. The time step causing the error is not added. The maximum calculation time indicated in the table corresponds to 5 iterations.

As a test, an experiment with control horizon set to 5 instead of 10, corresponding to a hessian size of 20×20 was executed. This HIL experiment was successful, which demonstrates the code is working properly. The maximum size of a QP that can be solved based on memory constraints is for qpOASES determined by the (sub)function block that calculates the QR-decomposition. This function needs 11 times a matrix of the size of the hessian to be stored. Based on the latter, a QP with a hessian of 44×44 can in principle be solved. Unfortunately, the required calculation time is too high to solve such problems.

B. Experiments on the pilot-scale set-up

Despite the unexpected failure of the qpOASES algorithm during the HIL experiments, experiments on the set-up are performed. Using the Hildreth algorithm, the results plotted

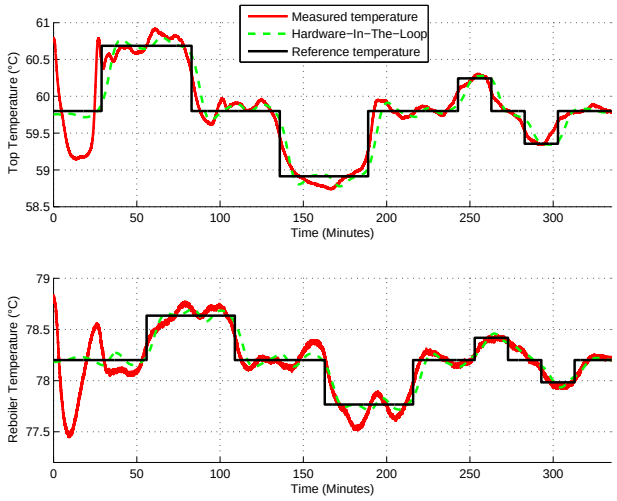

Fig. 3. Measured outputs for the top- and reboiler temperature for an experiment tracking a desired reference temperature profile with a MPC controller running on the PLC.

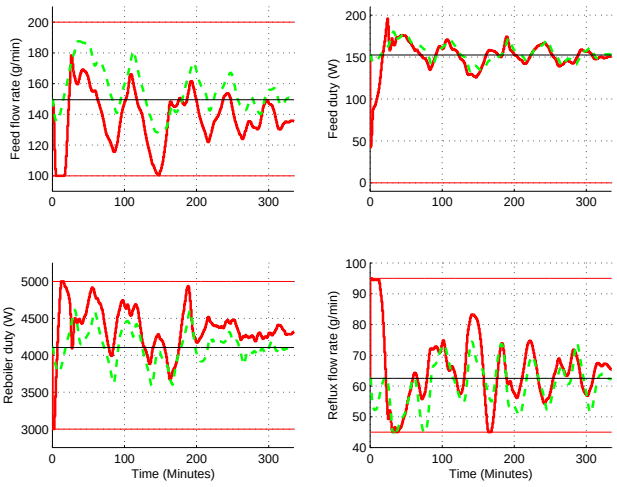

Fig. 4. Measured inputs for an experiment tracking a desired reference temperature profile with a MPC controller running on the PLC.

in Fig. 3 and Fig. 4 are obtained. The additional shift for the reference of 0.3^{o}C employed in the HIL experiments on the PLC, has been removed. The plot also contains a HIL experiment (dashed line). This can be considered as the best possible control that can be obtained with these MPC settings.

After a transient from the start-up temperatures above the set-point during the first 30 minutes (Fig. 3), the reference is followed accurately. In Table II the mean squared error (MSE) is presented. The table indicates the MSE nearly doubles for the experiment on the set-up compared to the HIL experiment. The inputs are plotted in Fig. 4. They are similar to these of the HIL experiments. The constraints are hit only during a few minutes for the flow rates.

The calculation time to present a solution to the column and the number of iterations required to solve the QP

TABLE II

THE MSE VALUE CALCULATED FOR THE PLC EXPERIMENTS.

	MSE
- Top temperature -	
Hildreth (HIL)	0.0253
qpOASES (HIL)	Failed
Hildreth (EXP)	0.0445
- Reboiler temperature -	
Hildreth (HIL)	0.0139
qpOASES (HIL)	Failed
Hildreth (EXP)	0.0255

are plotted in Fig. 5. The maximum number of iterations is 33 and at maximum 532 ms are required during this experiment. Some more run-times are presented in Table I.

Based on these experiments, online MPC can successfully be implemented on a PLC. However, the speed constraint and the memory constraint of 64 kB for a function, are important elements to take into account. These limit the size of the QP problems, and thus MPC problems, that can be solved. Due to this limitation of QP size, classic algorithms such as Hildreth are powerful enough for MPC on a PLC. The added complexity of state-of-the-art algorithms, e.g., qpOASES is for this type of problems not necessary.

VI. CONCLUSIONS

This paper has investigated the use of an online Model Predictive Control (MPC) algorithm on a Programmable Logic Controller for a pilot-scale binary distillation column to track reference temperatures. It has been experimentally demonstrated that it is possible to run an online MPC algorithm on this low-level industry standard device. Two QP algorithms, i.e., the Hildreth and qpOASES algorithm, have first been implemented and tested in hardware-in-the-loop experiments. The use of a PLC imposes both memory and computation time limitations. Based on the memory restrictions, the maximum size of the hessian in the QP is 64×64 for Hildreth and 44×44 for qpAOSES. In the HIL experiments a control horizon of 10 has been used yielding a hessian of 40×40. However, only the Hildreth algorithm has been found to always remain within the maximum allowed scan cycle monitoring time. Consequently, only the Hildreth algorithm has been evaluated in real-life experiments on the practical set-up. In addition, practical issues related to the implementation on the PLC have been highlighted.

ACKNOWLEDGMENT

Work supported in part by Katholieke Universiteit Leuven: OT/10/035, OPTEC Center-of-Excellence Optimization in Engineering (PFV/10/002), SCORES4CHEM (KP/09/005); by the Belgian Federal Science Policy Office: Belgian Program on Interuniversity Poles of Attraction; by the European Commission: Interreg IVa 2 Seas 07_022_BE_i-MOCCA. J.F. Van Impe holds the chair Safety Engineering sponsored by the Belgian chemistry and life sciences federation essenscia.

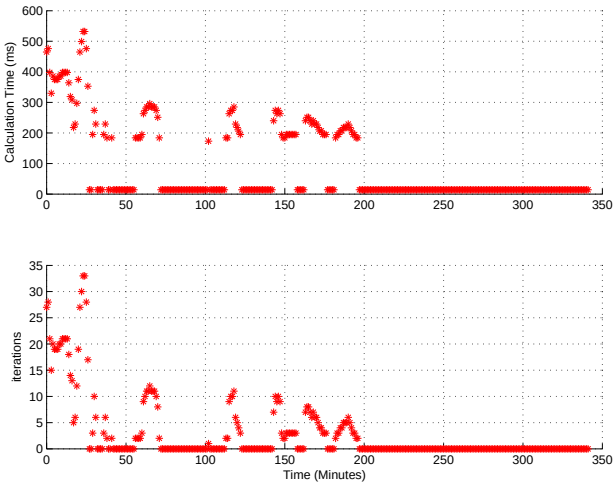

Fig. 5. Calculation time and number of iterations for the Hildreth QP algorithm running on the PLC for the experiment on the set-up.

REFERENCES

[1] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3 – 20, 2002.

[2] E. F. Camacho and C. Bordons. *Model Predictive Control*. Springer, 2003.

[3] H. J. Ferreau, H. G. Bock, and M. Diehl. An online active set strategy to overcome the limitations of explicit MPC. *International Journal of Robust and Nonlinear Control*, 18(8):816 – 830, 2008.

[4] C. Hildreth. A quadratic programming procedure. *Naval Research Logistics Quarterly*, 4:79 – 85, 1957.

[5] B. Huyck, J. De Brabanter, B. De Moor, J. Van Impe, and F. Logist. Model predictive control of a pilot-scale distillation column using a programmable automation controller. In *Proceedings of The European Control Conference 2013*, pages 1053 – 1058, 2013.

[6] B. Huyck, K. De Brabanter, F. Logist, J. De Brabanter, J. Van Impe, and B. De Moor. Identification of a pilot scale distillation column: A kernel based approach. In *18th World Congress of the International Federation of Automatic Control*, pages 471–476, 2011.

[7] B. Huyck, F. Logist, H. Ferreau, M. Diehl, J. De Brabanter, J. Van Impe, and B. De Moor. Towards online model predictive control on a programmable logic controller: practical considerations. *Mathematical Problems in Engineering*, 2012.

[8] J. L. Jerez, P. Goulart, S. Richter, G. Constantinides, E. Kerrigan, and M. Morari. Embedded Predictive Control on an FPGA using the Fast Gradient Method. In *European Control Conference 2013*, pages 3614 – 3620, 2013.

[9] M. Kvasnica, I. Rauova, and M. Fikar. Automatic code generation for real-time implementation of model predictive control. In *Computer-Aided Control System Design (CACSD), 2010 IEEE International Symposium on*, pages 993 –998, 2010.

[10] J. Maciejowski. *Predictive Control With Constraints*. Pearson Education Limited, 2002.

[11] J. Mattingley and S. Boyd. Cvxgen: a code generator for embedded convex optimization. *Optimization and Engineering*, 13:1 – 27, 2012.

[12] Siemens AG. CPU 31xC and CPU 31x: Technical specifications manual, March 2011. A5E00105475-12.

[13] I. Necoara and D. Clipici. Efficient parallel coordinate descent algorithm for convex optimization problems with separable constraints: Application to distributed MPC. *Journal of Process Control*, 23(3):243 – 253, 2013.

[14] E. N. Pistikopoulos, V. Dua, N. A. Bozinis, A. Bemporad, and M. Morari. On-line optimization via off-line parametric optimization tools. *Computers & Chemical Engineering*, 26:175 – 185, 2002.

[15] G. Valencia-Palomo and J. Rossiter. Efficient suboptimal parametric solutions to predictive control for PLC applications. *Control Engineering Practice*, 19:732–743, 2011.

[16] L. Wang. *Model Predictive Control System Design and Implementation Using MATLAB*. Springer-Verlag London Limited, 2009.