Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

Online model predictive control of industrial processes using low level control hardware: a pilot-scale distillation column case study

Bart Huyck[a,b], Jos De Brabanter[b], Bart De Moor[b], Jan F. Van Impe[a], Filip Logist[a,*]

[a]*Department of Chemical Engineering (CIT - BioTeC), KU Leuven, W. de Croylaan 46, B-3001 Leuven, Belgium*
[b]*Department of Electrical Engineering (ESAT - STADIUS), KU Leuven, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium*

Abstract

Throughout the years, the computing power of industrial controllers has steadily increased. Together with the development of efficient quadratic program (QP) solvers, this raises the question whether these devices can host an *online* model predictive controller (MPC). The applicability of online MPC is investigated using a programmable automation controller (PAC) and a programmable logic controller (PLC) for the control of an industrially relevant process, i.e., a pilot scale distillation column. It is demonstrated that both devices are capable of hosting MPC, however the limitations of the PLC are reached for the investigated set-up. Finally, guidelines and pitfalls for use in practice are highlighted.

Keywords: online MPC, linear MPC, PLC, PAC, pilot-scale distillation column

*Corresponding author
Email addresses: bart.huyck@kuleuven.be (Bart Huyck), jos.debrabanter@esat.kuleuven.be (Jos De Brabanter), bart.demoor@esat.kuleuven.be (Bart De Moor), jan.vanimpe@cit.kuleuven.be (Jan F. Van Impe), filip.logist@cit.kuleuven.be (Filip Logist)

Postprint version of manuscript accepted for Control Engineering Practice April 13, 2014

1. Introduction

One of the most successful advanced control methodologies in industry is *Model Predictive Control* (MPC). It has been a widely applied technique in process industry over forty years (Qin and Badgwell, 2003; Bauer and Craig, 2008; Mathur et al., 2008). The advantages of MPC over classic control methodologies, e.g., PID control, are its ability (i) to steer the process in an optimal way while taking proactively desired future behavior into account, (ii) to tackle multiple inputs and outputs simultaneously and (iii) to incorporate constraints (Maciejowski, 2002; Darby and Nikolaou, 2012).

Although model predictive control originates from the petrochemical industry (Richalet et al., 1978; Cutler and Ramaker., 1980), it has been successfully transferred to many other application areas, e.g., automotive industry (Hrovat et al., 2012), paper and pulp industry (VanAntwerp and Braatz, 2000), power converters (Kouro et al., 2009) and traffic control (Bellemans et al., 2006).

Current research on MPC focuses on, e.g., fast(er) algorithms (Bemporad et al., 2002; Ferreau et al., 2008; Wang and Boyd, 2010; Mattingley and Boyd, 2012) to encourage and simplify the use of MPC. Moreover, many of these algorithms were developed with *embedded* applications in mind. These are applications where the control scheme runs on an autonomous platform, often integrated in a machine. Typical examples are microcontrollers and field-programmable gate arrays (FPGAs). Due to the technical improvements, these devices have become readily available at relatively low prices. Nevertheless, as robustness, industry-proven reliability, and long-term support are important features for the process industry, less powerful programmable logic controllers (PLCs) are still omnipresent in process plants. These devices are robust and typically last the lifetime of an installation which eliminates expensive hardware replacements. However, evolving legislation may force companies to meet new standards that cannot always be reached by traditional control schemes. In such cases, an upgrade to, e.g., an MPC running on installed hardware can be considered. Consequently, it is interesting to evaluate these devices for the implementation of MPC algorithms.

Bemporad et al. (2002) suggested *explicit MPC* to solve the MPC problem. Here, the underlying quadratic program is solved offline and piecewise

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

linear solutions are stored in a look-up table. For the online application, the right piecewise linear solution is rapidly selected based on the measurements of the current state of the system. This development has been successfully exploited on PLC hardware by, e.g., Kvasnica et al. (2010); Valencia-Palomo and Rossiter (2011a,b, 2012). However, this approach is mainly feasible for small-scale systems and short time horizons as the storage complexity strongly increases with the size of the MPC problem.

In contrast, results for *online MPC* hosted by PLCs are scarce (e.g., Necoara and Clipici (2013)). Here, the underlying QP has to be solved online in each step. Also in Huyck et al. (2012) a successful illustration on a small-scale air-heating set-up has been presented. Although the knowledge gained in this study was substantial, the employed small-scale set-up had only a limited relevance for (process) industry. To explore the constraints that will be encountered when using MPC on a PLC in such environments, an example more relevant to industry, i.e., a pilot-scale distillation column, will be used in the current paper. As a first step, a programmable automation controller (PAC) is tested, while in a second step a transfer to a PLC is made. For the employed PLC hardware, not only memory and speed constraints will be evaluated, but also ease of implementation in practice (e.g., through the use of ready-to-use code) will be investigated.

The structure of the paper is as follows. Section 2 presents the pilot-scale distillation column. Section 3 discusses the general approach towards an implementation of MPC on a PLC. First, a linear model is identified, which exhibits an acceptable trade-off between model complexity and model accuracy. Second, the MPC algorithm is implemented and tuned. Section 4 describes the specific hard- and software that are employed, while Section 5 focuses on practical features related to the implementations on the employed control hardware. Section 6 elaborates on the memory consumption of the algorithms on the different devices. Next, Section 7 illustrates the identification results, while Section 8 provides and discusses the actual model predictive control results. To enable a systematic performance assessment, a more powerful PAC is first evaluated before the PLC is tested. Section 9 discusses the results obtained in view of practical applications and finally, Section 10 summarizes the main conclusions.

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

2. Pilot-scale distillation column set-up

The pilot-scale experimental set-up involves a packed distillation column (see Figures 1 and 2 as well as references Logist et al. (2009); Bonilla et al. (2012)). The column is about 4.5 m high and has an internal diameter of 6 cm. The column works under atmospheric conditions and contains three sections of about 0.95 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 approximately 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 power up to 250 W. At the bottom of the column a reboiler is present containing two electric heaters, each of 3000 W. These can be manipulated to deliver heat to the reboiler from 0 up to 6 kW. In the reboiler, a part of the liquid is vaporized while the rest is extracted as the 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.

In this set-up the following four variables can be manipulated: the reboiler power Qr (W), the feed rate Fv (g/min), the power 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 15 temperatures. The temperature probes are placed at the top of the column Tt and at three places on every packing section ($Ts1$ to $Ts9$). Temperature measurements $Tv1$ and $Tv2$ are available between section 1 and 2 and between section 2 and 3 respectively. Finally, the temperature measurement Tb in the reboiler of the column, and the temperatures of the feed before and after heating ($Tv0$ and $Tv2$, respectively) complete the list of measured temperatures. All temperatures are measured in degrees Celsius with a resolution of 0.01^oC.

The pilot-scale distillation column is equipped with a Compact FieldPoint (National Instruments (NI), Austin). This device consists of a NI cFP-2020 Real-Time Controller, a NI cFP-AIO-610 8-Channel Combination Analog

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Input/Output Module, NI cFP-RTD-124 8-Channel 4-Wire RTD Temperature Module, NI cFP-AI-110 8-Channel Analog Voltage and Current Input Module and NI cFP-AIO-600 8-Channel Combination Analog Input/Analog Output Module. This device is only used as a service-hatch to pass the measured outputs to the controller and desired inputs to the actuators of the column (Chambel et al., 2011).

There is no on-line measurement of the concentrations in the distillate and bottom streams although these are very important for the chemical process. The concentrations can be measured off-line, e.g., based on a their refractive index using a refractometer. However, since the concentrations for the system under study can be easily inferred from the temperatures when temperature and pressure are known, only a controller for the temperatures is implemented.

3. Approach

To systematically evaluate the possibilities for running MPC on PLCs, the following approach has been adopted. First a model is identified. Afterwards an MPC controller is implemented and tested experimentally. In order to gradually decrease the computational and memory available in the industrial control device, a PAC has been studied before moving to a PLC. In the view of safety, the developed controllers have always been evaluated through a hardware-in-the-loop (HIL) experiment before the actual test with the pilot-scale column itself. More detailed information is provided in the following subsections.

3.1. Identification of a model for control

Distillation columns can be described by linear low order systems (Skogestad, 1997). An ideal model for control has to combine an accurate tracking of the main column behavior with a low model complexity. Different linear time-invariant models exist. In this paper, identification based on low order transfer functions, often also called process models, is employed to obtain a model for control. This model is identified around a well-known and safe operating point. The identification of the model is performed using the Matlab System Identification Toolbox (Ljung, 2009). Only identification based on the continuous first and second order transfer functions with delay is considered. A Multiple-Input Multiple-Output (MIMO) process model is

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

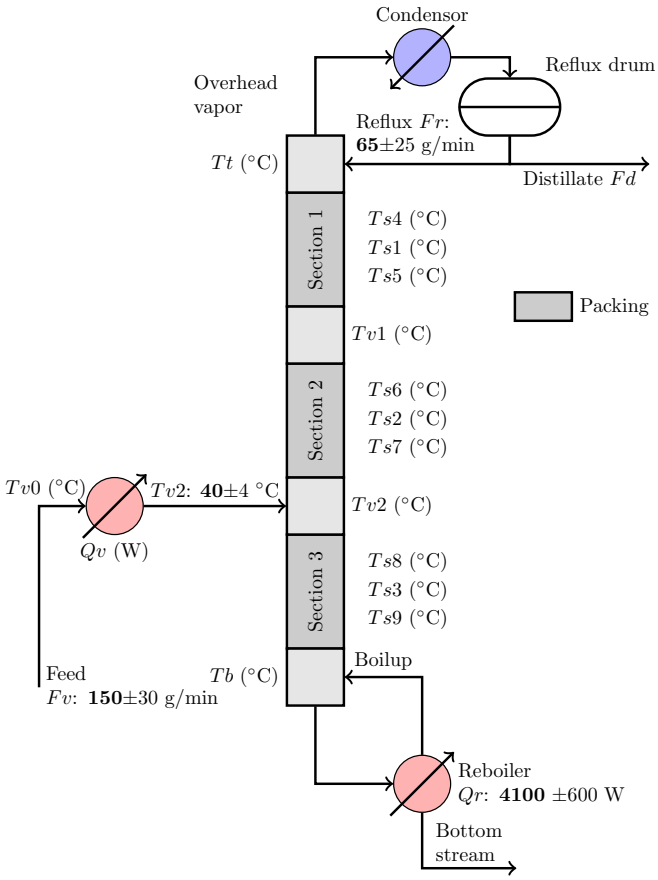

Figure 1: Diagram of the pilot-scale distillation column. Nominal set-points for identification experiments are printed in bold and are followed by the maximum employed deviations during identification experiments.

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

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

constructed based on two identified Multiple-Input Single-Output (MISO) process models. This model is converted to a state-space formulation for use in the model predictive controller.

3.2. Model predictive control formulation

In this subsection, the model predictive control formulation is described. Linear Model Predictive Control is well known in literature (Maciejowski, 2002; Camacho and Bordons, 2003; Rossiter, 2003; Wang, 2009). The basic formulation for linear MPC used for the practical implementation in this work is summarized below.

A linear, time-invariant discrete-time model is used:

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

with $\mathbf{A} \in \mathbb{R}^{p\times p}$, $\mathbf{B} \in \mathbb{R}^{p\times m}$ and $\mathbf{C} \in \mathbb{R}^{q\times p}$. Here, m, p and q are the number of inputs $\mathbf{u}(k) \in \mathbb{R}^m$, states $\mathbf{x}(k) \in \mathbb{R}^p$ and outputs $\mathbf{y}(k) \in \mathbb{R}^q$, respectively.

The cost function to be solved in each MPC step is given in this work:

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

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

with $\mathbf{y}_{ref}$ the output reference and $\hat{\mathbf{y}}$ the predicted output. $\mathbf{u}_{\text{ref}}$ relates to the corresponding control reference profile. H_p and H_c (with $H_c \leq H_p$) are the prediction and the control horizon of the controller, respectively. $\mathbf{W}_y \in \mathbb{R}^{q\times q}$ and $\mathbf{W}_u \in \mathbb{R}^{m\times m}$ are the positive definite weighting matrices. A terminal constraint ensuring stability can be omitted as long as the prediction horizon is long enough and the plant is stable (Maciejowski, 2002), which is the case in this work.

To remove the steady state error, an augmented model is constructed:

$$\underbrace{\begin{bmatrix}\Delta\mathbf{x}(k+1)\\ \mathbf{y}(k+1)\end{bmatrix}}_{\boldsymbol{\xi}(k+1)} = \underbrace{\begin{bmatrix}\mathbf{A} & \mathbf{0}\\ \mathbf{CA} & \mathbf{I}\end{bmatrix}}_{\tilde{\mathbf{A}}}\underbrace{\begin{bmatrix}\Delta\mathbf{x}(k)\\ \mathbf{y}(k)\end{bmatrix}}_{\boldsymbol{\xi}(k)} + \underbrace{\begin{bmatrix}\mathbf{B}\\ \mathbf{CB}\end{bmatrix}}_{\tilde{\mathbf{B}}}\Delta\mathbf{u}(k) \tag{3}$$

$$\mathbf{y}(k) = \underbrace{\begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}}_{\tilde{\mathbf{C}}}\begin{bmatrix}\Delta\mathbf{x}(k)\\ \mathbf{y}(k)\end{bmatrix}$$

where $\Delta\mathbf{u}(k) = \mathbf{u}(k) - \mathbf{u}(k-1)$ is the difference between two successive inputs, $\Delta\mathbf{x}(k) = \mathbf{x}(k) - \mathbf{x}(k-1)$ is the difference between two successive states and $\boldsymbol{\xi}(k)$ is the augmented state:

$$\boldsymbol{\xi}(\mathrm{k}) = \begin{bmatrix}\Delta\mathbf{x}(k)\\ \mathbf{y}(k)\end{bmatrix} \tag{4}$$

The calculation of the future predictions are based on Eq. (3). In a matrix formulation the future predictions can be written as:

$$\hat{\mathbf{y}}_{1\ldots H_p} = \mathbf{F}\boldsymbol{\xi} + \boldsymbol{\Phi}\Delta\mathbf{u}_{1\ldots H_c}. \tag{5}$$

The matrices $\mathbf{F}$ and $\boldsymbol{\Phi}$ can be found in Maciejowski (2002); Wang (2009). $\hat{\mathbf{y}}_{1\ldots H_p} \in \mathbb{R}^{H_p\cdot q}$ and $\Delta\mathbf{u}_{1\ldots H_c} \in \mathbb{R}^{H_c\cdot m}$ are the arrays into which the different vectors $\hat{\mathbf{y}}(k)$ and $\Delta\mathbf{u}(k)$ haven been stacked over the prediction and control horizon, respectively.

In the current work only input constraints are taken into account for the MPC problem:

$$\mathbf{u}_{\text{Min}}(k+j) \leq \mathbf{u}(k+j) \quad \text{for } j = 1\ldots H_c \tag{6}$$

$$\mathbf{u}_{\text{Max}}(k+j) \geq \mathbf{u}(k+j) \quad \text{for } j = 1\ldots H_c. \tag{7}$$

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

At present, no state constraints are included, but the methodology can be extended.

The optimization problem can be formulated as the minimization of cost function (2), subject to model equations (3) and input constraints (6) and (7). To solve the optimization problem, the procedure described in Wang (2009) is followed. By elimination of the states and using the following recursion for the inputs:

$$\mathbf{u}(k+j) = \mathbf{u}(k-1) + \sum_{j=0}^{H_c-1} \Delta\mathbf{u}(k+j), \tag{8}$$

a Quadratic Program (QP) is obtained:

$$\min_{\Delta\mathbf{u}} \frac{1}{2}\Delta\mathbf{u}_{1\ldots H_c}^T \mathbf{H} \Delta\mathbf{u}_{1\ldots H_c} + \mathbf{g}\Delta\mathbf{u}_{1\ldots H_c} \tag{9}$$

$$\text{subject to : } \begin{bmatrix} -\mathbf{C}_1 \\ \mathbf{C}_1 \end{bmatrix} \Delta\mathbf{u}_{1\ldots H_c} \leq \begin{bmatrix} -\mathbf{u}'_{\text{Min}} \\ \mathbf{u}'_{\text{Max}} \end{bmatrix} \tag{10}$$

with $\Delta\mathbf{u}_{1\ldots H_c}$ a vector containing the decision variables $\Delta\mathbf{u}(k+j)$ with $j = 1\ldots H_c$. To solve this QP problem, the Hildreth algorithm (Hildreth, 1957; Wang, 2009) and qpOASES (Ferreau et al., 2008) will be used.

The Hessian matrix $\mathbf{H}$ in Eq. (9) is a constant matrix as soon as all MPC parameters have been selected. It can be calculated with:

$$\mathbf{H} = \mathbf{\Phi}^T \mathbf{W}_{y_{bd}} \mathbf{\Phi} + \mathbf{W}_{u_{bd}}, \tag{11}$$

with $\mathbf{W}_{y_{bd}}$ and $\mathbf{W}_{u_{bd}}$ block diagonal matrices of $\mathbf{W}_y$ and $\mathbf{W}_u$ respectively. The Hessian can be computed offline as no online information is required. The aim in this work is to minimize the online calculation burden. To this en, matrices that can be computed in advance, are calculated offline and stored in the memory.

The gradient vector $\mathbf{g}$ in Eq. (9) has to be calculated online. It contains three parts depending on the current state $\boldsymbol{\xi}$(k), the reference of the inputs $\mathbf{u}_{\text{ref}}$ and the reference of the outputs $\mathbf{y}_{\text{ref}}$. The reference for the inputs $\mathbf{u}_{\text{ref}}$ is employed in such a way that the calculated inputs will not deviate too much from a well-known and safe nominal working point of the employed set-up. So, $\mathbf{g}$ has to be calculated according to

$$\mathbf{g} = \mathbf{G}_1\boldsymbol{\xi} - \mathbf{G}_2\mathbf{y}_{\text{ref},1\ldots H_p} - \mathbf{G}_3\mathbf{u}_{\text{ref},1\ldots H_c} \tag{12}$$

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

where $\mathbf{G}_1$, $\mathbf{G}_2$ and $\mathbf{G}_3$ are gradient matrices to be calculated according to Wang (2009); Maciejowski (2002). The gradient matrices $\mathbf{G}_1$, $\mathbf{G}_2$ and $\mathbf{G}_3$ are constant and are computed offline according to:

$$\mathbf{G}_1 = \mathbf{\Phi}^T W_{y_{bd}}^T F \quad (13)$$

$$\mathbf{G}_2 = \mathbf{\Phi}^T W_{y_{bd}}^T \quad (14)$$

$$\mathbf{G}_3 = W_{u_{bd}}^T. \quad (15)$$

The constraints in Eq. (10), i.e., the minimum and maximum admissible values $\mathbf{u}_{\mathrm{Max}}^{'}$ and $\mathbf{u}_{\mathrm{Min}}^{'}$ require online information and, hence, are calculated online. $\mathbf{u}_{\mathrm{Max}}^{'}$ is a column matrix with $\mathbf{u}_{\mathrm{Max}}^{'}(k+j) = \mathbf{u}_{\mathrm{Max}}(k+j) - \mathbf{u}(k-1)$, with $j = 1 \ldots H_c$ and similarly is $\mathbf{u}_{\mathrm{Min}}^{'}$ a column matrix $\mathbf{u}_{\mathrm{Min}}^{'}(k+j) = \mathbf{u}_{\mathrm{Min}}(k+j) - \mathbf{u}(k-1)$, with $j = 1 \ldots H_c$. The matrix $\mathbf{C}_1$ is a lower triangular matrix build up from identity matrices.

4. Hard- and software

The current section provides details about the control hardware and software used. To gradually move towards a PLC device, a PAC has been tested in an intermediate step. The software includes different algorithms for the online solution of the QP in each MPC step.

4.1. Programmable Automation Controller

As a PAC, a National Instruments CompactRIO has been used. This PAC controller consists of a NI cRio–9024 Real-Time Controller (800 MHz CPU, 512 MB of memory) (National Instruments Corporation, 2010), a cRIO–9114 Reconfigurable Chassis, a NI 9265 Analogue Current Output module and a NI 9217 RTD 24-Bit Analogue Input Module. The real-time controller is programmed with LabVIEW and is able to run a software library compiled from C/C++ code via a *call library function*. The library is compiled for the VxWorks 6.3 operating system with the GCC-compiler version 3.4.4.

The LabVIEW programming environment has been used to program the PAC. The LabVIEW programming concept is based on a Virtual Instrument (VI). A graphical user interface is composed of a front panel on a block diagram. The employed QP solvers are programmed in C and compiled with a

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

GNU Compiler Collection (GCC), which is a freeware compiler to a library compatible with the VxWorks operating system. The data between the LabVIEW VIs and the QP solvers, written in C, is exchanged by means of the LabVIEW *call library function*.

4.2. Programmable Logic Controller

A Siemens CPU319-3DP/PN PLC is used as PLC. The base memory of this CPU is increased to the maximum allowed 8 MB. This CPU is the fastest Siemens S7-300 CPU currently available. It takes 40 ns for one floating-point operation (Siemens AG, 2011). 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.

4.3. QP algorithms

To solve the QP problem, several algorithms have been used.

Hildreth QP algorithm. The Hildreth algorithm has been chosen for its limited number of code lines which makes it easy to implement. It has been written based on (Wang, 2009) in C for the PAC and in S7-SCL for the PLC. This algorithm calculates the solution in two steps. First, the unconstrained solution is calculated and if no constraints are violated, this solution is adopted. If a constraint is violated, a constrained QP is solved. The solution of the QP is then passed to the inputs of the set-up. For more information about the solution, see Hildreth (1957); Luenberger (1997); Wang (2009). If a solution to the QP could not be found, the unconstrained solution is compared to the constraint. If a constraint is violated, then that entry of the unconstrained solution is limited to the constraint.

qpOASES QP algorithm. qpOASES is an open-source C++ implementation of the recently proposed online active-set strategy (Ferreau et al., 2008). It 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

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

solver and the internal matrices 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 offered 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 of the QP solution and is not fully optimized for speed. On the PAC, the plain ANSI C implementation of qpOASES has been used. Although the full version of qpOASES is perfectly suited for hot starting, this feature is not used. Furthermore, the algorithm is only used with cold starts as a solution can be found in one step when no constraints are active. This makes it possible to start the search for a solution with offline, previously computed matrices.

5. Controller implementation features

Every industrial control device has its own characteristics. This section presents how the QP solvers have been implemented and what the consequences are of the typical features of the employed PAC and PLC devices.

5.1. Model predictive controller implementation

To compute a new input for the process, the sequence of actions presented in Algorithm 1 are followed. In advance, constant matrices are precomputed and the reference trajectory for the output is selected. This sequence of steps to solve the MPC problem has been used for both controller devices and the employed QP algorithms.

5.2. Features of the implementation on the PAC

The practical limitations while using PAC and PLC controllers is different. Programmable automation controllers employ a Real-Time Operations System (RTOS). It is an extra layer between the user code and hardware. Consequently, more resources are required to run both this RTOS and the user code. On the other hand, the operating system handles all basic tasks in a more user-friendly way, e.g., file system operations, parallelization of user programs and memory management. The CompactRIO runs VxWorks

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

```
Offline: Calculate H, G1, G2 and G3.
Online: Start PLC or PAC.
Store all precomputed matrices in working memory, together with the reference for
inputs and output.
while CPU is running do
    if controller interval passed since last call then
        Scale the in- and outputs.
        Calculate current state.
        Calculate g, u'Min and u'Max.
        case Hildreth algorithm
            Calculate the unconstrained inputs of the process.
            if unconstrained inputs violate constraints then
                while maximum number of iterations is not reached and solution
                not found do
                    Solve one iteration of the QP.
                end
                if maximum number of iterations is reached then
                    Use unconstrained solution, but inputs violating a constraint
                    are limited to that constraint.
                end
            end
        end
        case qpOASES
            while maximum number of iterations is not reached and solution not
            found do
                Solve one iteration of the QP.
            end
            if maximum number of iterations is reached then
                Use last sub-optimal solution.
            end
        end
        Scale and apply the calculated inputs to the system
    end
end
```

Algorithm 1: Steps to compute the inputs on the experimental set-up for the PAC and PLC.

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

as its RTOS. A compiler exists to convert (existing) C/C++ code. All implemented QP solvers are originally written in C or C++ and are converted into a library, supported by the RTOS.

During run-time, two programs run in parallel to control the distillation column set-up. The first LabVIEW program, further called the *Human Machine Interface Virtual Instrument* or *HMI VI*, performs the data exchange between the program and the in- and output cards of the Compact Fieldpoint, the scaling of the in- and outputs, the visualization of the process variables and logging of the data. The PI-controllers are also part of this program. A second LabVIEW program, further called the *Model Predictive Control Virtual Instrument* or *MPC VI*, performs the preparative calculations for the QP, e.g., the estimation of the state and selection of the current reference. Instead of using the built-in MPC controller (Balbis et al., 2005), the currently employed QP algorithms are programmed in C. The approach is similar to Canale et al. (2012), who implemented a QP solver on a National Instruments eXtensions for Instrumentation (PXI) system. However the employed QP solvers, the target system and the employed C-compiler are different. The QP algorithm is located in a separated library, called from within the MPC VI. The HMI VI runs at a rate of 10 Hz. The MPC VI is called every minute.

5.3. Features of the implementation on the PLC

No compiler exists to transform C/C++ source code to a running binary on a Siemens PLC. Therefore, the C/C++ code has to be translated into S7-SCL (ST). For this paper, the time consuming step has been taken to translate the qpOASES and Hildreth solvers to S7-SCL, the Siemens dialect of the ST language according to IEC 61131-3. The qpOASES solver is translated without the hot starting possibilities.

To calculate the appropriate inputs of the system and solve the QP, a number of built-in *function blocks* (FB) and *organization blocks* (OB) are programmed. Organization blocks are built-in functions called as hardware interrupts. Function blocks are user-defined functions with corresponding data stored in a data block (DB) with the same number. Fig. 3 depicts the order in which these blocks are called.

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

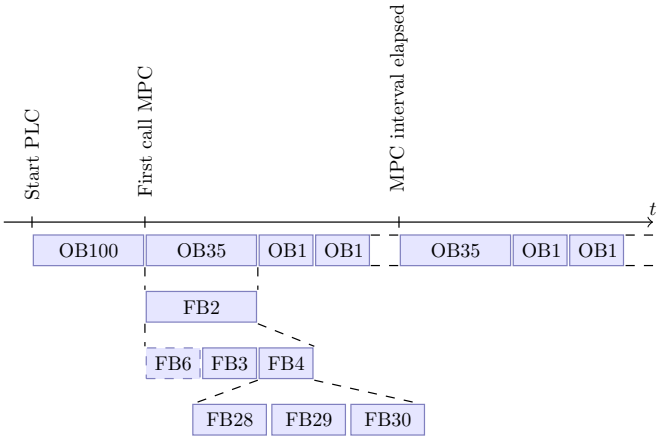

Figure 3: Schematic overview of the different employed organization and function blocks in the PLC.

Every function block has a corresponding data block. A data block cannot exceed 64 kB on the Siemens S7-319 CPU. The editor and the compiler impose some additional limitations. Initialization of a variable must be on a line not exceeding 255 characters. This is not a big issue, but only awkward when initializing matrices. More important is the limitation that an array initialized with unique elements cannot exceed 255 elements. Strangely enough, an array can be as large as desired as long as it fits into the 64 kB of available memory. To overcome this limitation, each array containing more than 255 elements has to be filled during runtime (e.g., in OB100 at the start) by arrays of maximum 255 elements. As can easily be understood, this situation is very unpractical. Linked with this situation is the need for absolute addressing if a function needs data in a data block different from its own data block. For example, to initialize matrices containing more than 255 elements, OB100 writes data to DB2. This operation requires fault-sensitive absolute addressing. The most important employed FBs and OBs to code the program are presented below. A graphical representation of the order of calling is given in Fig. 3.

OB100 - Cold Start. This block is called once when the controller is started. It is employed to initialize matrices containing more than 255 different elements. This procedure is followed to overcome the limitation an array cannot exceed 255 elements at compilation time. In the case a

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

matrix of more than 255 different elements is to be initialized, several arrays are combined in this block at runtime into a combined array. For the distillation column, the Hessian $\mathbf{H}$ and matrices $\mathbf{G}_1$, $\mathbf{G}_2$ and $\mathbf{G}_3$ are initialized in OB100.

OB1 - Main loop. This loop is started as soon as OB100 is finished. When this function finishes, it restarts. This loop is used to program standard tasks of the PLC. During the experiments, it is not used. It will run during the idle time of the CPU between two OB35 calls.

OB35 - Timed loop. This timed organization block is called every minute. It contains the necessary code to read the current inputs. This information is scaled and employed to calculate the current state (FB3). Together with the reference for the in- and outputs, the state is used to update vector $\mathbf{g}$ (FB2). Now, the QP is solved and the scaled solution will be passed to the outputs of the PLC.

FB2 - Main program. This function scales the in- and outputs, calls the estimator block, calls the QP solver itself and finally presents the calculated inputs to the system.

FB3 - State estimator. This block estimates the current state to be used to calculate the current gradient $\mathbf{g}$ in Eq. (9) for the QP solver.

FB6 - Column model. For hardware-in-the-loop experiments on the PLC, the state-space model of the column is implemented in this block.

FB4 - QP solver. This block calculates the input of the system. For the Hildreth algorithm, all code is written in this function block. For the qpOASES algorithm, a number of additional function blocks are called: FB28, FB29 and FB30. The block FB29 requires the most memory and, hence, determines the practical limits for using qpOASES on a PLC (see Section 6).

FB28 - Calculate relative distance. This function is only needed for the qpOASES algorithm and calculates the relative distance to the final solution (Ferreau et al., 2008).

FB29 - QR and Cholesky decomposition. This function is only needed for the qpOASES algorithm and calculates the QR decomposition based

on Givens rotations and the Cholesky decomposition (Golub and Van Loan, 1996).

FB30 - Matrix inversion. This function is only needed for the qpOASES algorithm. This block calculates the inverse of a lower triangular matrix by substitution.

Practical experiments on the pilot-scale column are performed in the following way. The LabVIEW *Human Machine Interface Virtual Instrument* runs on a PC which is connected to the PLC and the column by an ethernet connection. The QP solver is running on the PLC. An OPC server (NI OPC server 2012, OPC is *OLE for Process Control* where OLE is an acronym for *Object Linking and Embedding*) is responsible for the data exchange between the PC and the PLC.

6. Memory use and constraints

To get insight in the memory consumption of the applied algorithms on the PAC and PLC devices, the memory consumption is presented in this section. For the PAC, memory sizes are obtained by the built-in *performance and memory profiler*.

6.1. PAC

To get an idea of the required memory on the PAC device, two different methodologies are followed. First the built-in profiler is used to provide an idea of the memory consumption of the running VIs. Second, the total amount of used memory is verified when downloading the files to the device with LabVIEW. The latter is about 70 MB. According to the profiler, the memory requirements for the Virtual Instrument running the controller (i.e., MPC VI) are 102.7 kB for qpOASES and 98.9 kB for Hildreth.

Together with the HMI VI, which requires approximately 2795 KB and some additional general VIs, e.g. for error checking, the memory footprint is nearly 3500 kB. Compared to the total required memory on the device of approximately 70 MB, the memory needed for the VIs of the MPC controller is limited. The difference with the 70 MB used is mainly related to the overhead of the RTOS and additional general features.

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

6.2. PLC

For the PLC, the function blocks containing the QP solvers can be regarded as the bottlenecks for memory use. Hence, figures given are relate to these blocks only.

6.2.1. Hildreth Algorithm

For the Hildreth algorithm, FB4 is the function block with the highest memory consumption and is populated with the matrices to solve the QP. For a Hessian of size 40×40, the total required memory is 8 matrices of 1600 real elements, plus 7 real numbers, two vectors of 40 and two vectors of 80 real numbers. This results in 53124 bytes needed. Increasing to a Hessian of 44×44 elements exceeds the 64 kB limit of the FB. It has to be noted that based on symmetry 4 matrices can be eliminated. This makes it possible to increase the size of the Hessian to 64×64 elements which can be considered as the maximum practical limit for using the Hildreth algorithm in a PLC. Overcoming this limit is possible by storing matrices in different function blocks, but since absolute addressing is required, this is very fault sensitive and has to be avoided if possible.

6.2.2. qpOASES

For qpOASES, the QR and the Cholesky decomposition calculated in FB29 require the most memory. The calculation of these decompositions needs 11 matrices of the size of the Hessian. For the pilot-scale distillation column, a Hessian with 1600 elements is constructed. 11 matrices of 1600 elements violate the 64 kB limit. However, reusing temporary matrices employed in the QR decomposition for the Cholesky decomposition reduces the number of required matrices from 11 to 8 so that experiments can be performed. This limitation poses a strong practical constraint on using MPC based on qpOASES on a PLC. The total required memory in the employed formulation for hardware-in-the-loop experiments is 8 matrices of 1600 real elements plus 1 vector of 40 real elements and 15 real numbers which makes in total 51420 required bytes. Increasing the size of the Hessian to 44×44 results in 62188 bytes. This is the absolute maximum to fit the QR and the Cholesky decomposition into one function block. Separating both QR and Cholesky decomposition is not worthwhile, as the latter needs no extra memory to be allocated in the current implementation.

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

7. Identification results

In this section, the results are summarized to obtain a suitable model for control of the column.

7.1. Excitation signals

In order to generate estimation and validation data for the system identification, two experiments are performed. The excitation signal is built up with Pseudo Random Binary (PRB) signals for the different manipulated variables. Before the excitation signals are applied, the column is kept at a constant operating point for two hours to ensure that the column is in steady-state. The nominal steady-state values of the different manipulated variables are: a reflux flow rate Fr of 65 g/min, a feed flow rate Fv of 150 g/min, a feed heater duty Qv of 152 W to maintain a feed temperature $Tv2$ of 40°C and a reboiler power Qr of 4100 W. These nominal values are known to yield an appropriate operating point for the column (Logist et al., 2009). All manipulated variables are controlled by PI controllers except for the heating duties which are controlled directly. When the column has reached steady-state the experiment is started.

When the excitation signals are applied, all manipulated variables switch between two values. The reflux flow rate Fr fluctuates between 40 and 90 g/min, while the feed flow rate Fv changes between 120 and 180 g/min. The feed heater duty Qv is manipulated between 0 and 250 W and the reboiler power Qr switches between 3500 and 4700 W. These values are displayed in Fig. 1. The distillate flow rate Fd is manipulated in order to keep the content of the reflux drum at 40% of its maximum volume. All data are recorded with a sampling period of 100 ms.

The first PRB input signal is constructed in the following manner. The reboiler duty Qr is a repeated periodic signal of 6000 s. The clock period, i.e., the minimum time before the signal is allowed to switch, is 300 s. From previous experiments (Logist et al., 2009), it is known that the dynamics of the system are faster at the top of the column. Therefore, the clock period of the other inputs is slightly smaller. For the feed flow rate Fv and feed temperature $Tv2$ a clock period of 120 s is taken with a period of length 3720 s and for the reflux flow rate Fr, the clock period is 20 s with a period length of 5100 s. These input signals are combined into one experiment with a

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

time span of 25000 s. The second experiment is slightly slower. The periodic signal has a length of 8000 s for all manipulated variables and the clock period for Qr, Fv, $Tv2$ and Fr is 500 s, 120 s, 120 s and 60 s, respectively. The duration of the experiment is also 25000 s in total.

7.2. Data preparation

The sampling period of the three recorded data sets is reduced to 60 s. Therefore, for every 60 s a sample is taken from the original recorded data that is first filtered with zero-phase digital filter (-3dB frequency = 0.3 Hz). Next, an identification and a validation data set have been created. The first PRB data set described in the former section is selected to be the estimation data set, the second is the validation data set.

7.3. Model identification and validation

Here different quality indexes are calculated to assess the model quality, e.g., the FIT value as defined in the Matlab Identification Toolbox (Ljung, 2009) and the mean Squared errors (MSE). The model selection is based on model selection criteria as, e.g., the Akaike Information Criterion (AIC) (Akaike, 1974). However, operator experience is also taken into account.

For each output, a MISO model consisting of low order transfer functions, is fitted. After initial identifications with each of the four transfer functions of an identical structure, an attempt has been made to create a model with a lower AIC and a lower MSE/higher FIT. Therefore each of the input-output transfer functions pairs is exchanged with a different model structure. If re-identification resulted in lower AIC and MSE, and if the structure could be reasonably accepted based on operator expertise, then the new structure, called *PMIX*, will be adopted. In Table 1 the MSE, FIT and AIC are indicated for the final identified model.

From physical insight, one can assume that all subsystems have a time delay. This is not the case in the PMIX models, as several second order models appear without a time delay. In these cases, the second order pole masks the time delay. Re-estimation based on a first order transfer function with time delay resulted often in a worse AIC or FIT value. In such a case, the second-order subsystem has been preserved. The two selected PMIX models have been merged into one MIMO transfer function presented in Eq. (16) and converted to state-space.

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

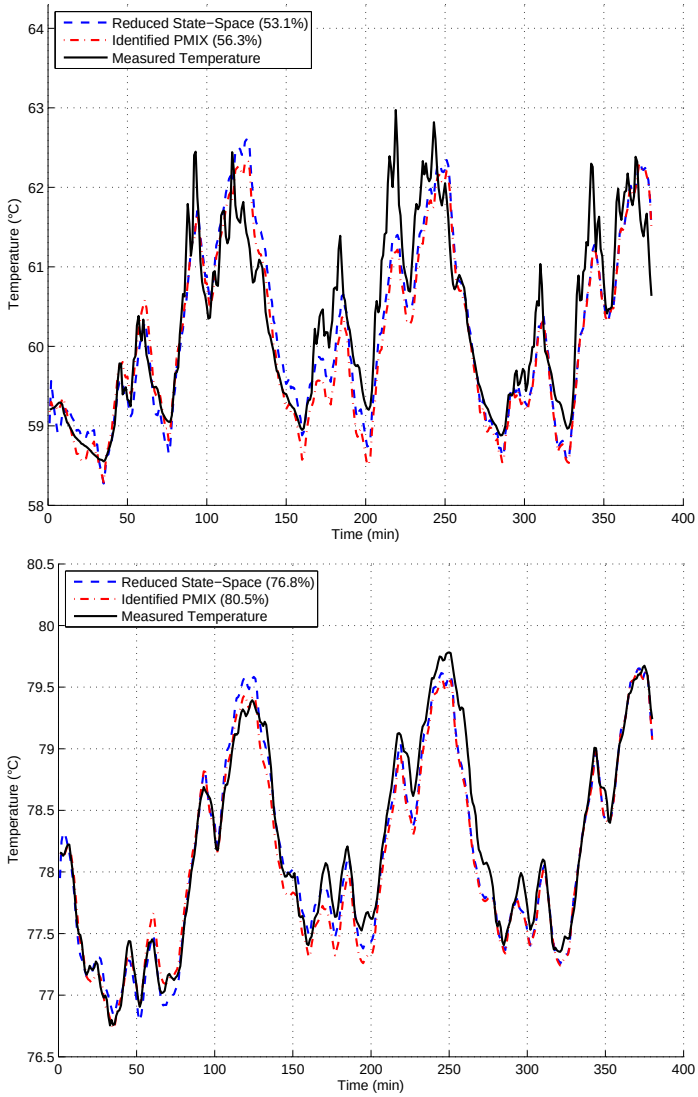

Figure 4: Validation of the top and reboiler temperature. Each figure depicts the simulation for the reduced state-space model and the identified PMIX model.

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

Table 1: The FIT and MSE values for the reduced state-space model and identified PMIX models for both the reboiler and top temperature.

Models	FIT	MSE	AIC
- Top temperature -			
Reduced State-Space	53.1%	0.2608	
PMIX transfer function	56.3%	0.2261	-3.83
- Reboiler temperature -			
Reduced State-Space	76.8%	0.0810	
PMIX transfer function	80.5%	0.0566	-1.66

$$\begin{bmatrix} Tt \\ Tb \end{bmatrix} = \begin{bmatrix} \frac{-2.26}{(1+2565s)(1+135s)} & \frac{0.53}{1+735s} & \frac{3.74}{(1+1803s)(1+78s)} & \frac{-3.45}{(1+2698s)(1+72s)}e^{-53.3s} \\ \frac{-1.87}{(1+890s)(1+169s)} & \frac{0.62}{1+1465s}e^{-307s} & \frac{5.21}{(1+2864s)}e^{-43s} & \frac{-2.36}{(1+1275s)(1+525s)} \end{bmatrix} \begin{bmatrix} \mathrm{Fv} \\ \mathrm{Qv} \\ \mathrm{Qr} \\ \mathrm{Fr} \end{bmatrix} \quad (16)$$

Conversion to discrete state-space is required for the model predictive control formulation. The selected discretization interval is one minute. This interval has been selected based on a trade-off between a sufficiently small model size on the one hand and a sufficient capability to represent the dynamics in the process on the other hand. The resulting model is of order 20. After plotting the Hankel singular values, no clear jump can be noted, but there is a significant reduction in state energy by a factor of 500 between the state with the highest energy and state 13. Therefore, the number of states to be left in the controller model, is selected to be 13. The reduced state-space model has been checked to be observable and controllable by a rank test following the standard procedure implemented in Matlab Control System Toolbox (MathWorks, 2009). In Fig. 4 two plots are depicted. The first one presents the simulation of the reduced state-space and original selected transfer function model for the top temperature, the second one does the same for the reboiler temperature. For the top temperature, both models follow the trend of the measured temperature well, although some peaks are not captured. Especially, sharp peaks and fast variations are not covered by the models. The fast varying peaks are believed not to disturb the control action of the controller due to the long time constants of the model. For the reboiler temperature, the measured temperatures are followed accurately.

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

Table 1 presents the FIT and MSE values for the plotted models. Conversion to state-space and model reduction results in only a small loss of accuracy of the model. As a result, the reduced state-space model is selected to be the controller model.

8. Model predictive control results

In this section, the observations when implementing and using an MPC on low level hardware to control the distillation column, are described. First, the results for the more powerful PAC are presented, while afterwards the observations for the PLC are provided.

8.1. Model predictive control settings

The different MPC horizons are determined in trial-and-error simulations and are set to $H_c = 10$ steps for the control horizon and $H_p = 50$ steps for the prediction horizon. Every step takes one minute. This leads to a Hessian of size 40×40 in Eq. (9). The weight matrix $\mathbf{W}_y$ 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_u has four elements on the diagonal $W_{u_{11}} = 0.8$, $W_{u_{22}} = 1$, $W_{u_{33}} = 0.8$ and $\mathbf{W}_{u_{44}} = 1$ for the feed flow rate, the feed duty, the reboiler duty and the reflux flow rate, respectively. This choice has been made to encourage the use of the flow rates, which are the fastest varying control variables. All of the off-diagonal elements are taken equal to 0.

As a reference trajectory for both the top and reboiler temperature, a sequence of steps is applied. The sizes of these steps for the top temperature are 0.88 and 0.62^{o}C. For the reboiler the large and small steps are 0.44 and 0.31^{o}C, respectively. The length of the steps is between 50 and 60 minutes for both the top and the reboiler temperature. Note, the shift of approximately 30 minutes between jumps in both reference values. As the outputs are highly correlated (Huyck et al., 2013), each change in reference trajectory for one of the two temperatures will also affect the other temperature.

8.2. Model predictive control on a PAC

8.2.1. Hardware-in-the-loop experiments

To make sure that the controlling LabVIEW VI and corresponding C-libraries for the MPC controller are implemented correctly, hardware-in-the-

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

loop experiments are performed. The VI running the MPC controller is connected to a linear plant model of the pilot-scale distillation column. The employed linear plant model is the discrete state-space representation of the MIMO transfer function in Eq. (16) obtained before model reduction. The number of states of this model is 20. The HIL simulations are plotted together with the experiments on the column in Fig. 5. Both the Hildreth and the qpOASES algorithm have been employed for HIL experiments. Given the identical result and correspondingly indistinguishable plots, only one HIL experiment is plotted. The overall shape of the reference is followed well, but it is clear that the column will never reach a steady-state situation for this experiment. After the HIL experiments, it can now be expected that the LabVIEW VI is working properly and, hence, it is employed on the pilot-scale set-up.

8.2.2. Experiments on the pilot-scale set-up

The top plot in Fig. 5 depicts the measured top and the reboiler temperature during the experiment. Both controlled variables follow the same reference trajectory as during the HIL experiment. The numerous changes yield a challenging reference path to be tracked. As mentioned earlier, the time constants of the different subsystems are longer than half an hour, causing the system to never reach steady-state. These references are selected in order to combine the safety regulations of the column with a sufficient number of set-point changes in the time slot available for experiments.

Two experiments are recorded: one experiment with the Hildreth algorithm and one with qpOASES. Unfortunately, the environmental conditions were different. This is of major importance as the pilot-scale distillation set-up is not thermally insulated from its environment. As a consequence heat loses depend largely on the environmental conditions. For the Hildreth experiment the ambient temperature was 17^{o}C at noon. At night the temperature was not lower than 14^{o}C. For the second experiment, the temperature at noon was 10^{o}C and close to zero at night.

In the first two hours of the experiment, both the top and reboiler temperatures are followed quite accurately. The plotted temperature obtained from hardware-in-the-loop experiments can be considered as the temperature to be tracked. The differences between the HIL experiments and the experiments on the set-up are limited for both QP solvers. The inputs, presented

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.

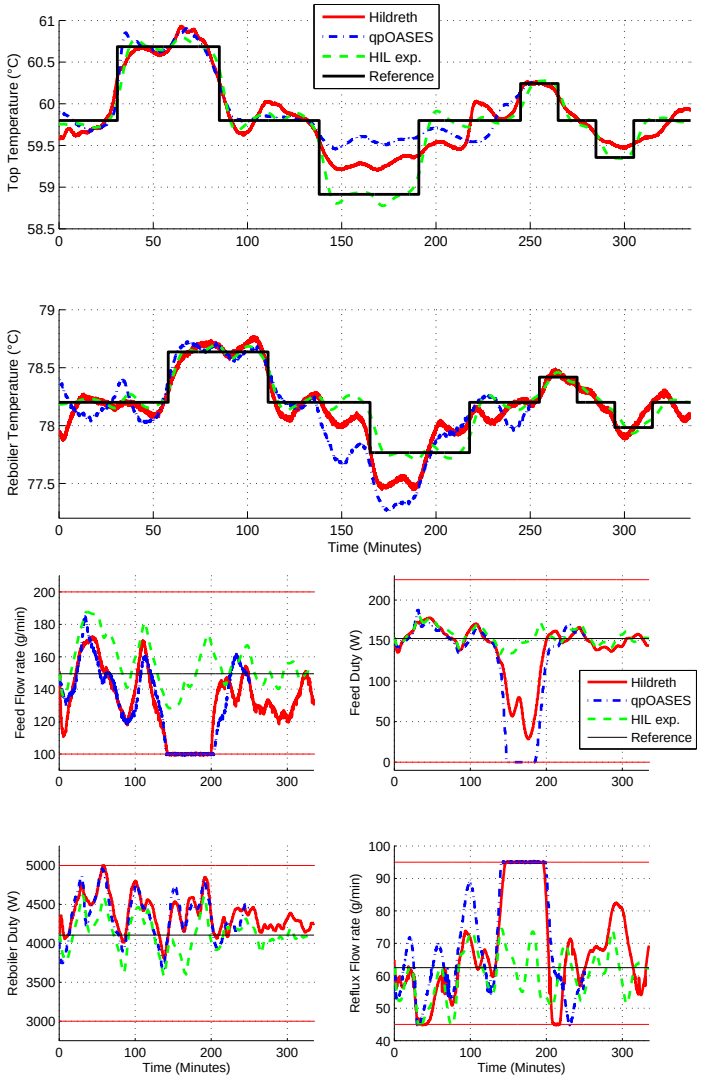

Figure 5: Measured outputs (top) and inputs (bottom) for experiments on the distillation column and HIL experiments tracking the desired reference temperature profile with an MPC controller running on the PAC.

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

in the bottom plot in Fig. 5, also differ only slightly form the HIL experiment.

However, between the second and fourth hour, both temperatures have to follow a step down in the temperature. As soon as the top temperature reference starts decreasing, both the top and the reboiler temperature decrease. For the experiments with the Hildreth solver, corresponding to an ambient temperature of 17^{o}C at noon, the top temperature only decreases half the applied temperature step for the top temperature. This temperature stays constant for half an hour and increases slightly to 3h40. This is the moment where both the top and reboiler reference are back at the nominal temperatures. The top temperature evolves now quickly to this temperature. The measured reboiler temperature is 0.25^{o}C below the reference in the interval 2h20 to 3h15. From 3h25 this temperature is close to the nominal reboiler temperature. For the experiments with the qpOASES solver, the top temperature hardly leaves the nominal temperature, the reboiler temperature is even further from the reference.

It is clear from the plot in the top plot of Fig. 5 that the top and the reboiler temperature do not follow their references accurately in this interval. From the input plots presented in the bottom plot of Fig. 5, one can see that two constraints are reached for Hildreth and three for qpOASES. This, together with an ambient temperature below the excitation experiment, causes the inaccurate control. Nevertheless, this difficult control is interesting to investigate the behavior of the different QP solvers. The latter part of the experiment from 4h on, is a repetition of the first 4 hours, but speeded up with a factor 2.

In Table 2 the mean cost $\bar{J}$, which is the cost J divided by the length of an experiment, has been adopted because not all experiments have exactly the same duration. The mean cost values corroborate the visual observations that the experiment with the Hildreth algorithm as QP solver is much closer to the reference than the experiment with qpOASES as solver.

Based on these two control experiments, it can be concluded that for this pilot-scale set-up the ambient temperature is of high importance. Although difficult to incorporate in the model, except if a (large) sequence of experiments throughout the year can be set-up, the quality of control can only be considered *good* when the ambient temperature differs only at most 3^{o}C

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

Table 2: The mean cost $\bar{J}$ calculated for hardware-in-the-loop (HIL) experiments and experiments (EXP) on the set-up using the PAC.

	$\bar{J}$
Hildreth (EXP)	2.7255
qpOASES (EXP)	5.0153
qpOASES (HIL)	0.8337

Table 3: The total run time as well as the mean, minimum and maximum run times for one MPC iteration are presented to indicate the speed of the algorithms on the PAC.

	run-time (ms)			
	total	mean	min.	max.
Hildreth (EXP)	1655	4.58	0.28	67.59
qpOASES (EXP)	333	1.33	0.84	3.11
qpOASES (HIL)	314	0.89	0.83	1.69

from from the value of 20.6°C measured during the identification experiment.

The bottom plot in Fig. 5 depicts the four inputs of the distillation column. The two heating powers do not reach the constraints for the Hildreth experiment. Both flow rates touch the constraints. For the qpOASES experiment, all controller variables hit the constraints except the reboiler duty.

The bottom plot in Fig. 6 depicts the number of iterations required to solve the QP problem for both the Hildreth and qpOASES experiment together with the HIL experiment. All algorithms require between 10 and 15 iterations to solve the MPC problem around 30 minutes after the start. One active constraint on the reflux flow rate is responsible for this peak in the number of iterations. For the HIL experiment, no more than one iteration is required to present a solution, demonstrating no constraints are hit anymore. For the two experiments on the real set-up, the number of iterations increases fast as soon as two or more constraints are hit in the interval 2h15 to 3h10. The maximum number of iterations for Hildreth is 70 and for qpOASES is 30. The number of iterations for qpOASES is lower to solve the same problem. For practical use, the required time is more critical. Based on the two plots displaying the required time, the upper left plot in Fig. 6 presents the required time for each calculation of an optimal input, the upper right is an extract of the left plot, but limited to 4 ms. As can be seen, the Hildreth

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

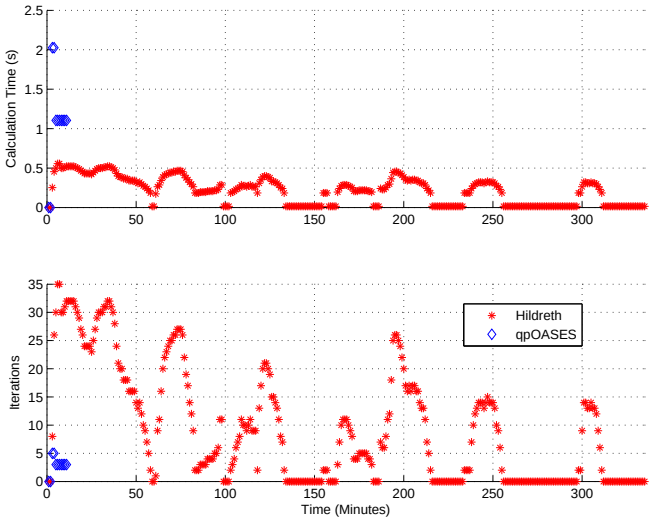

Figure 6: Calculation time and number of iterations for the employed QP algorithms on the PAC.

algorithm requires more time than the qpOASES algorithm if the number of iterations is above 20. qpOASES requires maximum 3.2 ms to solve the problem. Hildreth multiplies this value at some iterations with a factor of nearly 10. This confirms the suggestion made by Huyck et al. (2012). Large systems, and this column can be considered between small and average scales on an industrial level, can benefit from this new state-of-the-art solver on industrial devices.

Table 3 presents the mean time for one iteration and total runtime for the experiment. The mean time for one iteration for Hildreth is higher than that for the qpOASES algorithm. This demonstrates that the increased complexity for qpOASES leads to a decrease in computation time, but the system has to be *sufficiently large*. The cause for the peaks seen in the calculation time (upper left plot in Fig. 6) is not clear, but most likely a hardware interrupt for communication or module detection causes the algorithm to pause for some time. This phenomenon is not seen for the qpOASES algorithm as these modules were removed from the CompactRIO device. Table 3 also indicates the minimum and the maximum required calculation times for the QP solvers. It is clear that the Hildreth algorithm requires significantly more

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

time to solve the QP. Note that the maximum time required for Hildreth is more than 20 times higher than the maximum required time for qpOASES.

The PAC has clearly sufficient computation power to run the MPC controller and the PI controllers together with the data logging routines on this set-up. Whichever QP solver used, the computation time is below 5 ms. Together with the approximately 30 ms for the MPC VI, this is far below the one minute update interval between two MPC inputs. The employed working memory is around 70 MB. Be aware that this value contains the controller programs and the operation system memory demands. In summary, the question of whether MPC can be applied on a PAC for this pilot-scale distillation column, can be answered positively.

8.3. Model predictive control on a PLC

This section presents the results of the final aim of this work, i.e., running an MPC on a PLC to control the pilot-scale distillation column. First, hardware-in-the-loop experiments are carried out.

8.3.1. Hardware-in-the-loop experiments

In Fig. 7, the in- and outputs are plotted for the HIL experiments on PLC. The settings for the MPC are taken identical to those for the PAC device. A shift of plus 0.3 and minus 0.3^{o}C for the top and reboiler references respectively is applied to make the control a little bit more challenging such that the input constraints are touched during the HIL experiments.

For the Hildreth algorithm, the number of iterations is presented in Fig. 8. The required time to solve the QP is plotted in the upper plot of Fig. 8. The maximum required time is around 556 ms for 35 iterations (Table 5). The minimum value for the run times of 14 ms is the required time to present a solution if a QP has to be solved.

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 pilot-scale distillation column HIL experiments on PLC, this maximum is set to 250 iterations. This corresponds to a calculation time of 3970 ms. With a cycle time of 4 seconds for the organization block initiating the MPC solver, this is the absolute maximum allowed calculation time

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

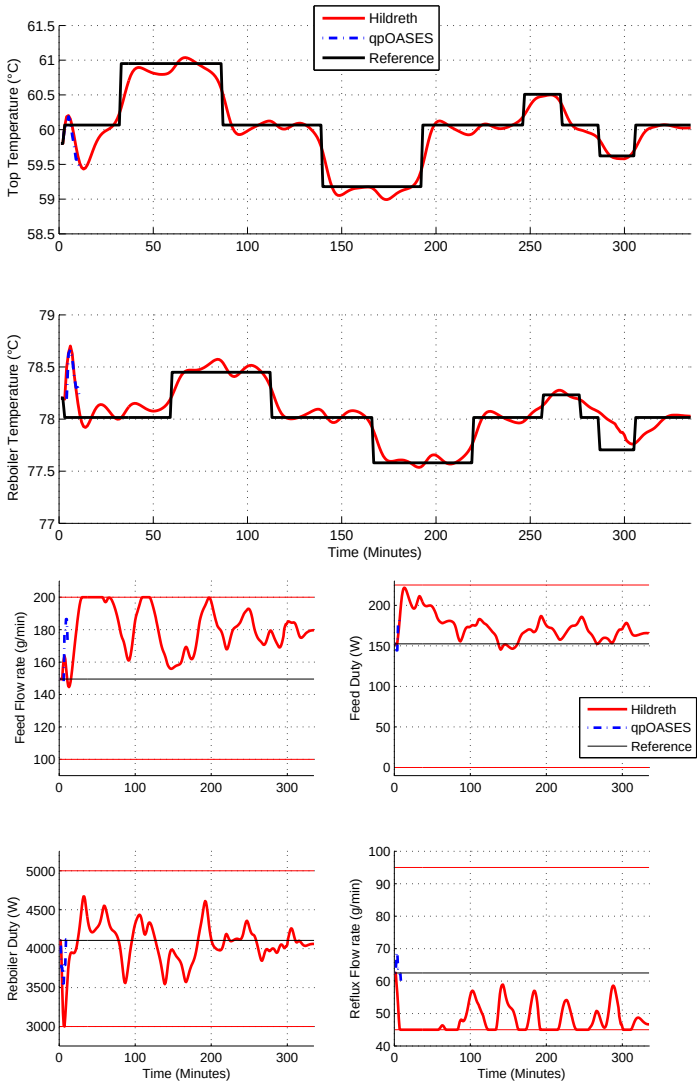

Figure 7: Measured outputs (top) and inputs (bottom) for the top- and reboiler temperature for an experiment tracking a desired reference temperature profile with a MPC controller running on the PLC.

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

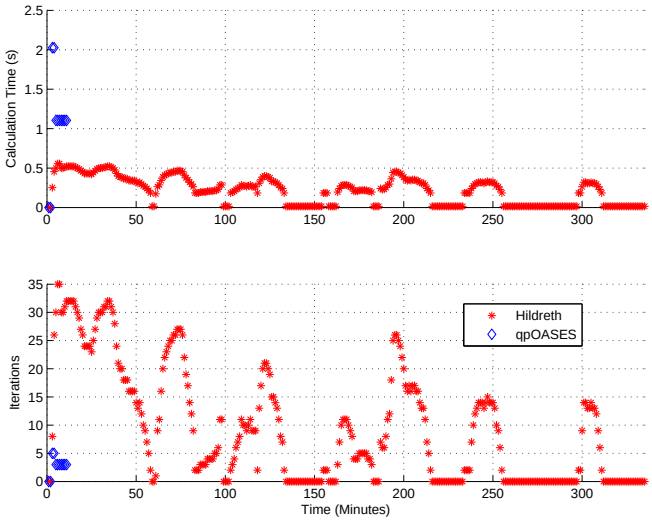

Figure 8: Calculation time and number of iterations for the used QP algorithm on the PLC.

Table 4: The mean cost $\bar{J}$ calculated for hardware-in-the-loop (HIL) experiments and experiments (EXP) on the set-up using the PLC.

	$\bar{J}$
Hildreth (HIL)	0.7668
qpOASES (HIL)	Failed
Hildreth (EXP)	1.6622

for the maximum number of iterations. On the pilot-scale set-up, an MPC update cycle of 1 minute is selected. The calculation times to solve the QP are far below this cycle time. The calculation time of the QP is therefore not considered to be a bottleneck for the implementation of MPC on a PLC for the pilot-scale set-up.

As already mentioned in Section 5.3, the other programming issues, e.g., maximum amount of 64 kB for each function block and the difficult initialization of array larger than 255 elements are much harder to take into account. Based on the latter issues, a practical limit for a QP to be solved with the Hildreth algorithm is a Hessian with 64×64 elements. It is possible to increase that size, but then you have to spread the matrix data in different

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

Table 5: The total, mean, minimum and maximum run times for one MPC iteration is presented to indicate the speed of the algorithms 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

data blocks which makes it even more difficult to manage as a programmer. The memory constraint is not yet reached. 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 and to increase the QP problem size to 64×64 elements as suggested.

For the qpOASES algorithm, the experiments 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 that the algorithm needed more than 3.95 s. Increasing the cycle time to 10 seconds, or even one minute, caused the scan cycle monitoring time to be set to its maximum setting of 5999 ms. Unfortunately, even this is not high enough for the qpOASES algorithm. 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 as employed in this HIL experiment. This is at least 8.75 s and far above the maximum allowed calculation time, the scan cycle monitoring time, of 6 s for the PLC. The additional calculations required for this algorithm, need too much time to calculate the solution online. The presented time values in Table 5 are calculated on the first 7 time steps. The time step causing an error is not added. The maximum calculation time indicated in the table corresponds to 5 iterations.

As already mentioned in Section 5.3, the practical upper memory limit for this algorithm should be a Hessian of size 44. The required calculation time, however, reduces this upper limit. An experiment with control horizon set to 5 instead of 10, corresponding to a Hessian size of 20 was successful.

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

This successful HIL experiment for qpOASES demonstrates that the code is working properly. The observation that a control horizon of 10 steps reaches the computing power limits of the employed PLC, which is the most powerful of its series, demonstrates that on a PLC does not need a sophisticated solver as, e.g., qpOASES. The speed constraint is the bottleneck for this algorithm. The Hildreth algorithm on the other hand succeeds to solve the MPC problem with control horizon 10. For the Hildreth algorithm, the *practical* memory constraint of 64 kB for one block is reached first for this experiment.

In contrast to the PAC exponents, the required time to solve the QP is higher for qpOASES than for Hildreth. Although the number of iterations is different, it is also important to repeat the S7-SCL translation which is not fully optimized for speed, which can affect the calculation time. Optimization of the algorithm is not done as the practical memory constraint limits the use of qpOASES on a PLC.

8.3.2. Experiments on the pilot-scale set-up

Nevertheless 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 in Fig. 9 are obtained. The additional shift for the reference of 0.3^oC employed in the HIL experiments on the PLC, has been removed. The ambient temperature was 14^oC at the start of the experiment and 19^oC at noon.

After a transient from the start-up temperatures above the set-point during the first 30 minutes (top plot of Fig. 9), the reference is followed clearly more accurate than for the experiments on the PAC. The different environmental conditions during the PLC experiment mainly cause the better control, which was confirmed by the mean cost values in Tables 2 and 4. In contrast to the PAC experiments, the reference trajectory is now also followed when the temperature decreased below the reference starting temperature for both the top and the reboiler temperature.

The inputs plotted in the bottom plot of Fig. 9 are also more close to 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 calculate the QP are plotted in Fig. 10. Some more

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

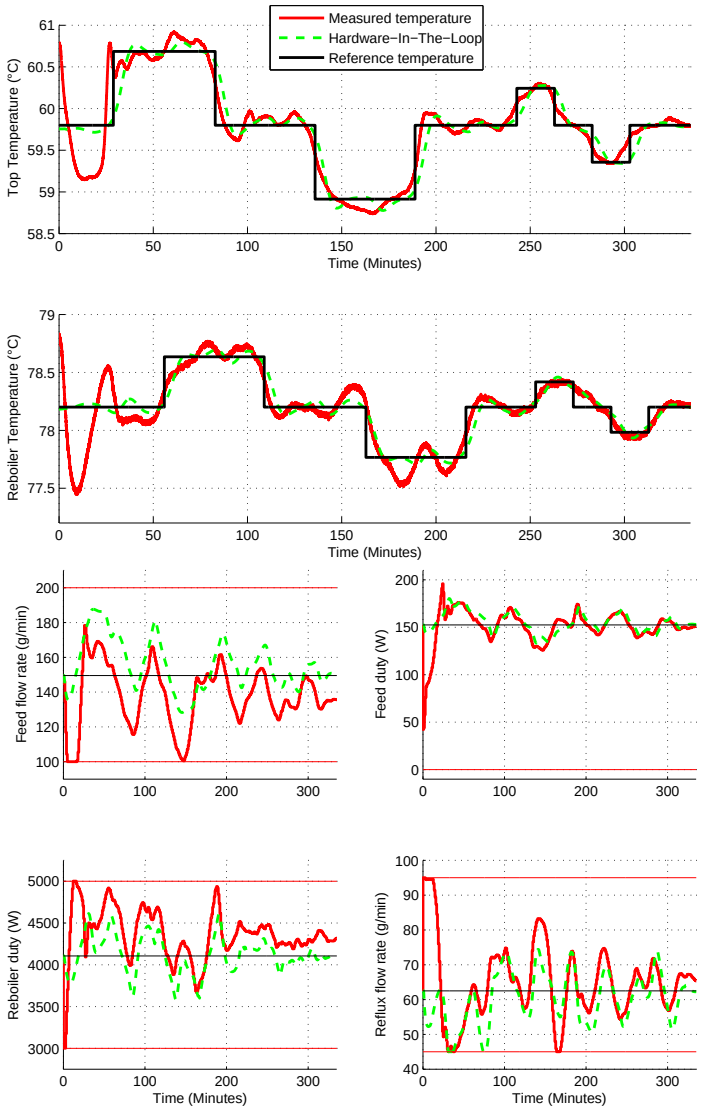

Figure 9: Measured outputs (top) and inputs (bottom) for the top- and reboiler temperature for an experiment tracking a desired reference temperature profile with a MPC controller running on the PLC.

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

Postprint version of paper published in Control Engineering Practice 2014, vol 28, pages 34 - 48.
The content is identical to the published paper, but without the final typesetting by the publisher.

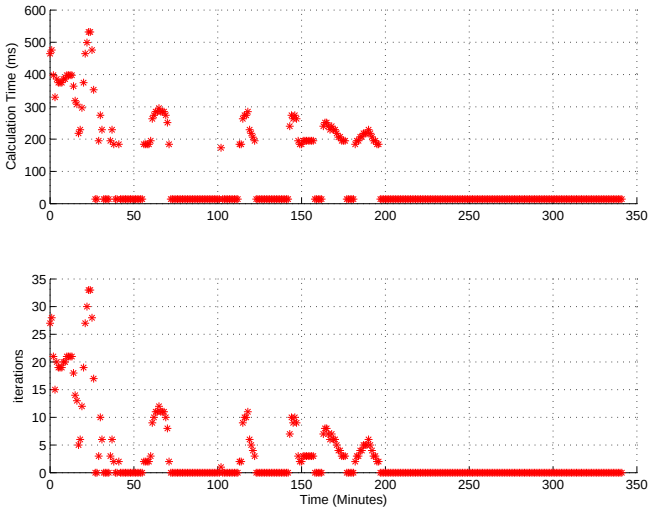

Figure 10: Calculation time and number of iterations for the used QP algorithm on the PLC.

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

9. Practical guidelines

To encourage the use of MPC in industry, the ease of implementation of model predictive controllers on commonly used devices has to be investigated. In this paper two standard industrial devices have been employed, i.e. (i) a PAC device, which can be seen as a robust computer with real-time properties and additional in- and outputs, and (ii) a PLC, a standard industrial automation device. Experiments with MPC on the PAC demonstrate that MPC can easily be implemented. Neither memory nor speed constraints are encountered in this paper and it is believed that even much larger systems can be MPC-controlled by these types of devices. However, absolute limits were not investigated. The practical implementation of MPC by an engineer is realizable in a limited amount of time. The software is well documented, is easy to use and has many features. In summary, MPC implemented on PAC devices is ready for industry.

Journal homepage: http://www.journals.elsevier.com/control-engineering-practice/
Original file available at: http://dx.doi.org/10.1016/j.conengprac.2014.02.016

The second investigated device is a PLC. Historically, this device is situated in the automation branch of the industry. The evolution in industry has been to request additional features, e.g., logging and human-machine interaction, which go together with the increase in computation power of the PLC. The research in this work to employ MPC on a PLC demonstrated that these devices, although capable, are not the ideal devices to run MPC. The limited amount of memory and especially the limited speed of the devices make it difficult to implement MPC. Based on the experience from this work, one can conclude that online MPC on a PLC is only applicable for problems resulting in a QP of limited size. For the PLC used for this study, a Hessian of size 64×64, or 4096 elements is the absolute maximum. Important to notice is that the classical Hildreth algorithm is able to successfully run a hardware-in-the loop test and experiments for the pilot-scale set-up.

The state-of-the-art qpOASES algorithm, created with MPC applications in mind, fails in this test due to its complexity. Notice, however, that it was not optimized for speed. The hardware time constraint that a function cannot run longer than 5999 ms, is reached very fast for qpOASES. For the Hildreth algorithm, the 560 ms required for the experiments in this paper is sufficiently removed from the previously mentioned time constraint. Although QP algorithms tailored for MPC exist, simple algorithms have been found to do the job in the current situation using the PLC as a host. Moreover, the uncommon programming language is an extra barrier for the easy applicability of MPC on PLCs, as the employed algorithms are complex and require thorough testing. Given the easy applicability for MPC on a PAC, PLCs are not advisable for MPC applications in industry. To make a PLC a valuable alternative for a PAC, not only speed and memory limits need to increase drastically, e.g., by a factor of 100, but also the standard programming languages employed in a PLC have to include more commonly used languages such as C or C++. However, in that situation the difference between a PAC and a PLC has become rather small.

10. Conclusion

In this paper, the use of an online model predictive control algorithm hosted by a programmable automation controller and a programmable logic controller for a pilot-scale binary distillation column has been investigated.

A linear state-space model of the pilot-scale distillation column has been obtained via system identification. This model is employed to design a model predictive controller which is used in practical experiments on the set-up. The runtimes for two online QP solvers, the Hildreth and the qpOASES algorithm, have been evaluated and compared. It has been demonstrated that it is possible to run an online MPC algorithm on a PAC which is not only able to run the supervisory algorithm, but is also responsible for data logging, human-machine interfacing and low level control. It has also been experimentally verified that the pilot-scale set-up can be MPC controlled by a PLC using the Hildreth algorithm. Unfortunately, the current implementation for the qpOASES algorithm surpassed the limits of the current state-of-the-art PLC devices such that these experiments were not successful. Possible future work is the integration of output (state) constraints.

Acknowledgements

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, IAP VII/19 (DYSCO); 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.

References

Akaike, H., 1974. A new look at the statistical model identification. IEEE Transactions on Automatic Control 19, 716 – 723.

Balbis, L., Katebi, R., Dunia, R., 2005. Graphical based predictive control design. In: Proceedings of the IEEE Conference on Control Applications. pp. 297 – 302.

Bauer, M., Craig, I. K., 2008. Economic assessment of advanced process control - a survey and framework. Journal of Process Control, 2 – 18.

Bellemans, T., De Schutter, B., De Moor, B., 2006. Model predictive control for ramp metering of motorway traffic: A case study. Control Engineering Practice 14, 757 – 767.

Bemporad, A., Morari, M., Dua, V., Pistikopoulos, E. N., 2002. The explicit linear quadratic regulator for constrained systems. Automatica 38, 3 – 20.

Bonilla, J., Logist, F., Degrve, J., De Moor, B., Van Impe, J., 2012. A reduced order rate based model for distillation in packed columns: Dynamic simulation and the differentiation index problem. Chemical Engineering Science 68 (1), 401 – 412.

Camacho, E. F., Bordons, C., 2003. Model Predictive Control. Springer.

Canale, M., Cerone, V., Razza, V., Regruto, D., 2012. Rapid prototyping of predictive controllers through an industrial platform. In: Proceedings of American Control Conference. pp. 4715 – 4720.

Chambel, A. J., Pinheiro, C. I., Borges, J., ao M. Silva, J., 2011. Methodologies for input-output data exchange between labview and matlab/simulinksoftware for real time control of a pilot scale distillation process. In: 21st European Symposium on Computer Aided Process Engineering. pp. 708 – 712.

Cutler, C., Ramaker., B., 1980. Dynamic matrix control - a computer control algorithm. In: Proceedings of the Joint American Control Conference, San Francisco, 1980. pp. Paper WP5–B.

Darby, M. L., Nikolaou, M., 2012. Mpc: Current practice and challenges. Control Engineering Practice 20 (4), 328 – 342.

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

Golub, G. H., Van Loan, C. F., 1996. Matrix Computations. Johns Hopkins University Press, Baltimore, MD, USA.

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

Hrovat, D., Di Cairano, S., Tseng, H., Kolmanovsky, I., 2012. The development of model predictive control in automotive industry: A survey. In: IEEE International Conference on Control Applications (CCA). pp. 295 – 302.

Huyck, B., De Brabanter, J., De Moor, B., Van Impe, J., Logist, F., 2013. Model predictive control of a pilot-scale destination column using a programmable automation controller. In: Proceedings of the European Control Conference. pp. 1053 – 1058.

Huyck, B., Ferreau, H., Diehl, M., De Brabanter, J., Van Impe, J., De Moor, B., Logist, F., 2012. Towards online model predictive control on a programmable logic controller: practical considerations. Mathematical Problems in Engineering, vol. 2012, Article ID 912603, 20 pages.

Kouro, S., Cortes, P., Vargas, R., Ammann, U., Rodriguez, J., 2009. Model predictive control - a simple and powerful method to control power converters. IEEE Transactions on Industrial Electronics 56, 1826 – 1838.

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

Ljung, L., 2009. System Identification Toolbox User's Guide. The MathWorks, Inc, Natick.

Logist, F., Huyck, B., Fabré, M., Verwerft, M., Pluymers, B., De Brabanter, J., De Moor, B., Van Impe, J., 2009. Identification and control of a pilot scale binary distillation column. In: Proceedings of the European Control Conference. pp. 4659 – 4664.

Luenberger, D. G., 1997. Optimization by Vector Space Methods. John Wiley & Sons.

Maciejowski, J., 2002. Predictive Control with Constraints. Pearson Education Limited.

Mathur, U., Rounding, R. D., Webb, D. R., Conroy, R. J., 2008. Use model-predictive control to improve distillation operations. Chemical Engineering Progress 104, 35 – 41.

MathWorks, T., 2009. Control System Toolbox User's Guide. The MathWorks, Inc, Natick.

Mattingley, J., Boyd, S., 2012. CVXGEN: a code generator for embedded convex optimization. Optimization and Engineering 13, 1 – 27.

National Instruments Corporation, June 2010. Operating instructions and specifications CompactRIO NI cRIO-9024. 375233C-01.

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

Necoara, I., Clipici, D., 2013. Efficient parallel coordinate descent algorithm for convex optimization problems with separable constraints: application to distributed mpc. Journal of Process Control 23 (3), 243 – 253.

Qin, S., Badgwell, T. A., 2003. A survey of industrial model predictive control technology. Control Engineering Practice 11, 733 – 764.

Richalet, J., Rault, A., Testud, J., Papon, J., 1978. Model predictive heuristic control: Applications to industrial processes. Automatica 14, 413 – 428.

Rossiter, J., 2003. Model-based predictive control: a practical approach. CRC.

Skogestad, S., 1997. Dynamics and control of distillation columns: A tutorial introduction. Chemical Engineering Research and Design 75, 539 – 562.

Valencia-Palomo, G., Rossiter, J., 2011a. Efficient suboptimal parametric solutions to predictive control for PLC applications. Control Engineering Practice 19, 732 – 743.

Valencia-Palomo, G., Rossiter, J., 2011b. Programmable logic controller implementation of an auto-tuned predictive control based on minimal plant information. ISA transactions 50 (1), 92 – 100.

Valencia-Palomo, G., Rossiter, J., 2012. Novel programmable logic controller implementation of a predictive controller based on laguerre functions and multiparametric solutions. Control Theory & Applications, IET 6 (8), 1003 – 1014.

VanAntwerp, J., Braatz, R., 2000. Fast model predictive control of sheet and film processes. Control Systems Technology, IEEE Transactions on 8, 408 – 417.

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

Wang, Y., Boyd, S., 2010. Fast model predictive control using online optimization. IEEE Transactions on Control Systems Technology 18, 267 – 278.