2014 IEEE Conference on Control Applications (CCA)
Part of 2014 IEEE Multi-conference on Systems and Control
October 8-10, 2014. Antibes, France

An Environment for the Efficient Testing and Implementation of Robust NMPC

Sergio Lucia, Alexandru Tătulea-Codrean, Christian Schoppmeyer and Sebastian Engell

Abstract— **In the last years many research studies have presented simulation or experimental results using Nonlinear Model Predictive Control (NMPC). The computation times needed for the solution of the resulting nonlinear optimization problems are in many cases no longer an obstacle due to the advances in algorithms and computational power. However, NMPC is not yet an industrial reality as its linear counterpart is. Two reasons for this are the lack of good tool support for the development of NMPC solutions and the fact that it is difficult to ensure the robustness of NMPC to plant-model mismatch. In this paper, we address both these issues. The main contribution is the development of an environment for the efficient implementation and testing of NMPC solutions, offering flexibility to test different algorithms and formulations without the need to re-encode the model or the algorithm. In addition, we present and discuss the approach of multi-stage robust NMPC to systematically deal with the robustness issue. The benefits of our approach are illustrated by experimental results on a laboratory plant.**

I. INTRODUCTION

Model Predictive Control (MPC) is a control strategy that has been successfully applied at many industrial plants, especially in its linear variant [1]. The most important reason for this success is its ability to handle coupled multivariable systems with constraints. Its nonlinear variant, Nonlinear Model Predictive Control (NMPC), has been studied intensely by the research community and many simulation results have been published in the last years, including large-scale and highly nonlinear systems based on rigorous models (e.g. [2] or [3]). Although some companies develop industrial NMPC implementations for some processes (see [4], [5], [6]) its practical use is still in its early stages and not a widespread reality.

Some years ago, one of the main reasons for this gap between the academic research and the industrial practice was the computation time that was required to solve the resulting nonlinear programming problems. In the last years the progress on algorithms and computation power has made it possible to dramatically reduce the computation times needed to solve NMPC problems even to the microsecond range [7], if appropriate tools are used.

At the same time, many software tools have recently been developed in the academia such as MUSCOD II [8], ACADO [9], NMPC tools [10], OptCon [11] or the MPT Toolbox [12] among others. The named tools can solve different kinds of problems including Nonlinear Model Predictive Control formulations.

One of the main obstacles for the knowledge transfer from the academia to the industry via these tools is their lack of modularity and the lack of sustainability of the implementations. Most of these tools require an implementation of a model in a particular syntax, and to interface the tools to any other necessary components such as a simulator or an observer. If the model has to be changed, all the codes have to be modified again. The lack of modularity also makes it difficult to compare the computational performance and solution quality of different approaches or tools (or to combine parts of different tools) because they require a completely new implementation, which will result in an obscure and time-consuming comparison.

Another key issue that prevents the widespread use of NMPC is that most of the available formulations do not consider explicitly the presence of uncertainty in the model, although all models are uncertain. Different approaches have been proposed to tackle this problem which are based e.g. on min-max formulations [13], [14] or on tube-based methods [15] [16], [17]. However, these approaches are rarely included in the available NMPC tools, either because they are open-loop approaches and hence give conservative results (cf. [18]), or because they require assumptions that are difficult to fulfill in general, as in the case of tube-based methods.

In this work we propose to integrate multi-stage NMPC [19] in a modular and transparent implementation. Multi-stage NMPC is a robust NMPC approach that has been proven to provide very promising results even for industrial examples [20] with several uncertainties, no matter if they are constant [21] or time-varying [22]. The starting point of the approach is to describe the uncertainty as a scenario tree. In this manner, it can explicitly be considered that at next sampling times, new measurements will be available and the control inputs can be adjusted accordingly to counteract the effects of the uncertainty, acting as recourse variables. This reduces significantly the conservativeness of the approach compared to typical open-loop robust approaches. The main drawback of the approach is the larger size of the resulting optimization problem.

The main contribution of this paper is the introduction of a new concept for the modularization of a NMPC implementation, dividing it into four main components: model, optimizer, observer and simulator. We also integrate our recent robust NMPC approach to deal with uncertainties in

S. Lucia, A. Tătulea-Codrean, C. Schoppmeyer and S. Engell are with the Process Dynamics and Operations Group, Department of Biochemical and Chemical Engineering, Technische Universität Dortmund, 44227 Dortmund, Germany (e-mail: {sergio.lucia, alexandru.tatulea-codrean, christian.schoppmeyer, sebastian.engell} @bci.tu-dortmund.de)

978-1-4799-7409-2/14/$31.00 ©2014 IEEE

a systematic way. We present the environment DO-MPC as a possibility to achieve these goals. DO-MPC includes a modular implementation of an NMPC approach to which any other available software can be coupled, by means of implementing a standardized interface, in order to represent the needs of the specific problem. We also include automatic plotting and logging of the data as well as the implementation of multi-stage NMPC. This is done in an efficient way by using CasADi [23] for the efficient and automatic calculation of the derivative information.

The remainder of this paper is structured as follows. Section II explains the concepts and main components of a modular implementation of NMPC. The central contribution of this paper is presented in Section III, where the environment DO-MPC is presented. Section IV presents the multi-stage approach for the systematic consideration of uncertainty in NMPC. Illustrative experimental and simulation results are shown in Section V and the paper is concluded in Section VI.

II. A Modular NMPC Development Environment

One of the main outcomes of the European research project EMBOCON (Embedded Optimization for Resource Constrained Platforms) was the development of the open source software platform GEMS (Generic EMBOCON Minimal Supervisor) [24].

The central idea of GEMS is to offer a set of general and standarized interfaces to simplify the process of developing and deploying a model-based control algorithm to a real system, and to offer a so-called supervisor that manages the flow of information between the different parts of the implementation. For this purpose, the implementation of a model-based control approach is divided into four main components: the model, the optimizer, the observer and the simulation or real application (see Fig. 1). The exchange of information between the different modules is managed and logged by a supervisor. Using this conceptual idea, existing or newly developed algorithms for control, for simulation or for state estimation of a system can be implemented based on the GEMS interfaces, which are programmed in plain C-code to facilitate the use of existing tools and the extension of them. The only necessary step is to provide the information required by GEMS in the template form of its interfaces.

The information about the model (ODEs, states, control inputs, etc.) is transferred to the other modules via the interfaces which can then make use of it to solve the corresponding optimization, estimation or simulation problems. When the models are updated or changed, there is no need to update the code in each one of the different implementation parts, because they will automatically get the information about the new model via the interfaces. This enhances the sustainability of an NMPC implementation. Note that if different models need to be used for the different modules, e.g. in the simulator a complex model is used but a simplified one in the controller, this can also be taken into account by defining several model modules for the same problem.

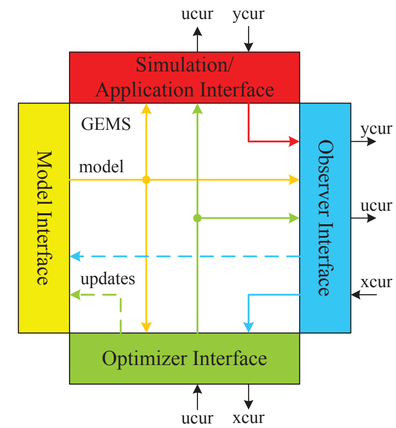

Fig. 1: Modular scheme of an NMPC implementation with four main blocks: Model, Optimizer, Observer and Simulator.

All the different modules are implemented as shared libraries within GEMS. They have to be compiled before hand so that the supervisor can load the corresponding library for each module at run-time. This ensures portability, transparency and configurability of the final NMPC implementation. The implementation is done in the Linux OS which makes the real-time managing of the modules possible. GEMS is freely available and it can be downloaded from [24].

III. DO-MPC: An Environment for an Easy, Modular, Robust and Efficient Development of NMPC

The desired features of an NMPC development platform described before have been implemented in DO-MPC (TU DOrtmund MPC). DO-MPC is a platform that uses the main ideas of GEMS to provide users with an easy, modular, robust and efficient way to realize sustainable implementations of NMPC.

The main added value of DO-MPC with respect to other available MPC tools is that instead of offering a black-box solution to the MPC problem, it provides a platform to develop own NMPC realizations with a very low effort. We use CasADi [23] as a building block to develop the necessary modules and interfaces. CasADi is a tool for automatic differentiation and dynamic optimization which makes it possible to manage the model and derivative information in an easy and efficient way. Furthermore, we provide templates using the scripting language Python for each one of the modules including an implementation of the multi-stage NMPC approach, which is presented in the next section. These templates provide a user-friendly environment for the formulation of a new NMPC problem that only requires the definition of the model equations, the control problem and the estimation problem. However, it is not necessary to use the provided templates and the user can couple any existing software just by writing an interface to DO-MPC.

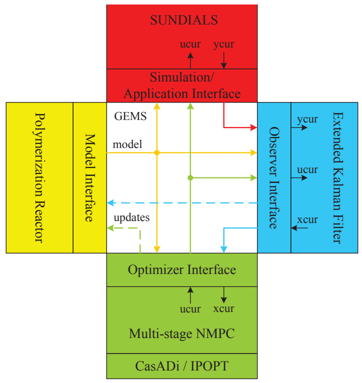

Fig. 2: Example of a DO-MPC configuration.

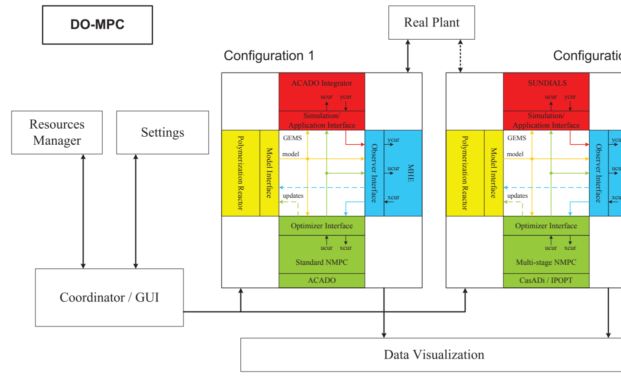

Fig. 3: Example of an NMPC implementation using DO-MPC with two configurations in parallel.

Once the information for the different modules is provided using the templates, CasADi automatically generates C-code. The autogenerated C-code is compiled to shared libraries, which are loaded at run-time by DO-MPC. This ensures that the exchange of the different modules can be done by just loading a different shared library at run-time.

An example of the configuration of DO-MPC with the four modules using a full state measurement and the SUNDIALS [25] integrators as simulator can be seen in Fig. 2.

Along with the simplicity of the implementation, one of the main features of DO-MPC is its modular implementation that makes it possible to use different configurations in parallel. Different optimizers, cost functions, or observers can be tested online, while maintaining the rest of the modules. This can be very useful for monitoring or comparison purposes. Additionally, a degree of redundancy can be introduced to cope with possible failures of some of the algorithms. DO-MPC also features an automatic data visualization tool which shows any variable previously defined, along with the predictions that the NMPC controller is computing to enhances the understanding of the controller performance. An example of the structure of DO-MPC can be seen in Fig. 3. The monitoring of the performance of an NMPC controller is an important subject of research and the DO-MPC environment facilitates its use, while already including techniques based on multivariate statistics [26]. Other techniques presented recently in the literature will also be implemented in the future [27],[28]. The structure of the tool and a minimal graphical user interface has been developed using the Qt5.1 framework for C++ [29]. In order to ensure the real-time capabilities of the tool, the implementation is done in a multithreaded way. This ensures that the different computationally demanding tasks (optimization, plotting, monitoring) are deployed on concurrent threads so that their execution is performed without interruptions. Together with the use of the auto-generated efficient C-code in the form of shared libraries for each one of the modules, this leads to a very efficient implementation which at the same time is transparent, modular and sustainable. The tool will be freely available by the end of 2014.

IV. Multi-stage Nonlinear Model Predictive Control

As the optimizer module in DO-MPC the multi-stage robust NMPC approach [19] can be used to handle the uncertainty that is inherent to any real model. Multi-stage NMPC is based on the representation of the uncertainty as a scenario tree (see Fig. 4), as in the field of stochastic programming [30]. The key idea of this representation is that the fact that new measurements will be available at the next sampling times is taken into account explicitly in the prediction done by the NMPC controller. In this manner, the future control inputs depend on the realization of the uncertainty, acting as recourse variables, which decreases the conservativeness of the approach compared to open-loop min-max approaches. We consider the following discrete-time nonlinear system

$$x_{k+1}^j = f(x_k^{p(j)}, u_k^j, d_k^{r(j)}),$$

where $x_{k+1}^j \in \mathbb{R}^{n_x}$ is the j-th node of the scenario tree at stage $k+1$, and $x_k^{p(j)}$ is the parent node from which if the control input $u_k^j \in \mathbb{R}^{n_u}$ is applied and the realization $d_k^{r(j)} \in \mathbb{R}^{n_d}$ occurs, the child node x_{k+1}^j is obtained (for example in Fig. 4, $x_2^6 = f(x_1^2, u_1^6, d_1^3)$). In order to make the notation clear, the index set of all occurring indices (j,k) is denoted by I. The control inputs cannot anticipate the realization of the uncertainty and therefore the control inputs branching from the same node have to be equal. This is enforced by the so-called non-anticipativity constraints. To avoid the exponential growth of the tree with the prediction horizon, the uncertainty is assumed to be constant after a certain point (robust horizon) until the final prediction horizon (cf. [19]). The optimization problem that is solved at each sampling time can be formulated in a general framework as:

$$\min_{x_k^j, u_k^j \forall (j,k) \in I} \left(\sum_{i=1}^{N} (\omega_i J_i(x_{k+1}^j, u_k^j))^{\alpha} \right)^{1/\alpha} \quad \forall\, x_{k+1}^j, u_k^j \in S_i, \tag{1a}$$

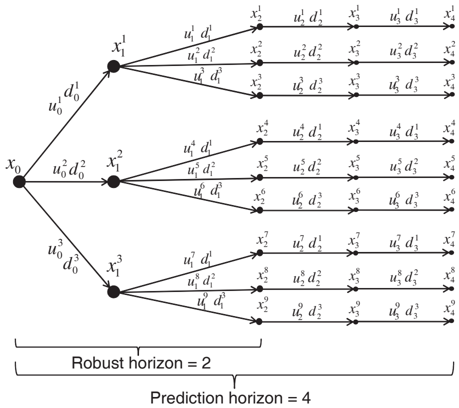

Fig. 4: Scenario tree representation of the uncertainty evolution for multi-stage NMPC.

subject to:

$$
\begin{aligned}
& x_{k+1}^j = f(x_k^{p(j)}, u_k^j, d_k^{r(j)}), && \forall\, (j, k+1) \in I, && \text{(1b)}\\
& g(x_k^j, u_k^j) \leq 0\ , && \forall\, (j,k) \in I, && \text{(1c)}\\
& u_k^j = u_k^l \text{ if } x_k^{p(j)} = x_k^{p(l)} && \forall\, (j,k), (l,k) \in I, && \text{(1d)}
\end{aligned}
$$

where $g(x_k^j, u_k^j)$ represents general and possibly nonlinear constraints on the states and the inputs at each node of the tree. The cost of each scenario S_i with probability ω_i is denoted by $J_i(x_{k+1}^j, u_k^j)$ defined as:

$$
J_i(x_{k+1}^j, u_k^j) = \sum_{k=0}^{N_p-1} L(x_{k+1}^j, u_k^j),\ \ \forall\, x_{k+1}^j, u_k^j \in S_i, \quad (2)
$$

where $L(x_{k+1}^j, u_k^j)$ is the stage cost, which represents a general - possibly economic - cost function. The non-anticipativity constraints in (1d) enforce that the decisions u_k^j with the same parent node $x_k^{p(j)}$ must be the same. Using the formulation parametrized on α, it is possible to represent the multi-stage NMPC approach if $\alpha = 1$ is chosen, whereas if $\alpha = \infty$ is chosen, a closed-loop min-max approach is obtained, in which feedback is taken explicitly into account. A comparison of both approaches was presented in [19]. If the number of scenarios is $N = 1$, the problem is reduced to standard NMPC. If the non-anticipativity constraints in (1d) are modified such that a single sequence of control inputs has to satisfy the constraints for all cases of the uncertainty then an open-loop approach is obtained. As it can be seen the formulation of the robust NMPC problem in this framework makes it possible to have a direct comparison with several other robust approaches. In addition, the uncertainty is treated in a systematic way, which makes easy to apply the approach to different problems. The only additional effort compared to a standard NMPC implementation is to define the uncertain parameters and a range of uncertainty to build a suitable scenario tree. Although this cannot be guaranteed for the general nonlinear case, very often the worst case lies at some of the vertices of the uncertainty set and therefore a reasonable scenario tree can be generated by considering the combinations of the different possible values of the uncertainty. The implementation of multi-stage NMPC is done using CasADi's interface to IPOPT [31] and compiling the resulting code as a shared library that is used by DO-MPC. As reported in previous work [20], [32], the implementation with CasADi is highly efficient making it possible to solve challenging problems with several uncertainties in real time.

V. Results for a Two-tank System

In this section we use DO-MPC for the real-time control of a laboratory two-tank system at the Process Dynamics and Operations Group at TU Dortmund. The plant consists of two tanks that are linked by a connecting pipe (see Fig. 5). In our setting the available control inputs are the valves V3 and V7 that control the flow from tank B1 to tank B2 and the outflow from tank B2 respectively. A constant inflow through V10 of 150 l/h is assumed to for tank B1 and there is no additional inflow into tank B2.

The two-tank system can be represented by a hybrid model that includes 2 differential equations for each mode of operation. Note that the dynamics of the system change if the level in tank B2 h_2 is above or below the connection point which connects the bottom of tank B1 with tank B2 at height H. The equations can be written as follows:

$$
\begin{aligned}
&\text{If } h_2 \leq H:\\
&\quad \frac{dh_1}{dt} = \frac{1}{A_1}\left(Q - \frac{u_1\sqrt{h_1}}{\sqrt{c_{13}u_1^2 + c_{23}}}\right),\\
&\quad \frac{dh_2}{dt} = \frac{1}{A_2}\left(\frac{u_1\sqrt{h_1}}{\sqrt{c_{13}u_1^2 + c_{23}}} - c_7 u_2\sqrt{h_2}\right),\\
&\text{If } h_2 > H:\\
&\quad \frac{dh_1}{dt} = \frac{1}{A_1}\left(Q - \frac{u_1\sqrt{h_1 - h_2 + H}}{\sqrt{c_{13}u_1^2 + c_{23}}}\right),\\
&\quad \frac{dh_2}{dt} = \frac{1}{A_2}\left(\frac{u_1\sqrt{h_1 - h_2 + H}}{\sqrt{c_{13}u_1^2 + c_{23}}} - c_7 u_2\sqrt{h_2}\right),
\end{aligned}
\quad (3)
$$

where h_1 and h_2 represent the water levels in the tanks and A_1 and A_2 are the cross-sectional areas, being. u_1 and u_2 are the control commands that act on the valves V3 and V7. The constants that define the behavior of the valves are c_{13}, c_{23} and c_7. All the parameters of the model are summarized in Table I, where d_1 and d_2 are the diameters of the tanks.

TABLE I: Parameter values of the two tank model

Parameter	Value	Units
d_1	0.13	m
d_2	0.06	m
H	0.405	m
Q	150	l/h
c_{13}	3.4275×10^7	s^2/m^5
c_{23}	0.9128×10^7	s^2/m^5
c_7	2.7154×10^{-4}	$m^{2.5}/s$

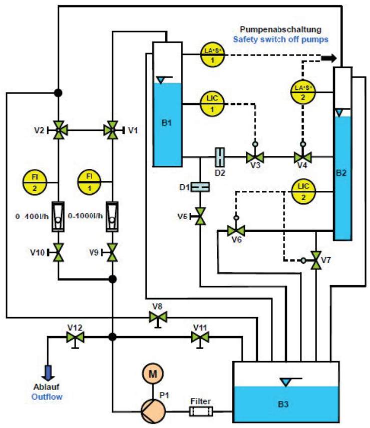

Fig. 5: Schematic representation of the two-tank system under consideration.

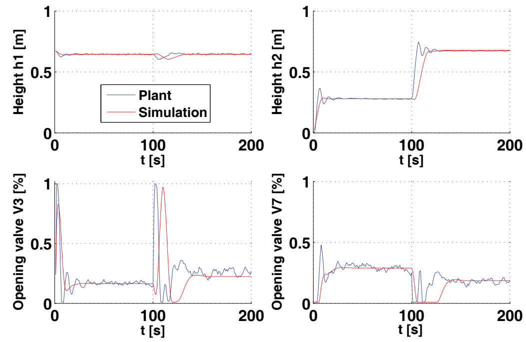

Fig. 6: Simulation and experimental results for the NMPC of the two-tank system using DO-MPC.

The optimal control problem that has to be solved at each sampling time can be written as:

$$\min_{u_1,u_2} \quad \sum_{k=0}^{N_p} \pi_1 \left(h_1 - h_1^{\text{ref}}\right)^2 + \left(h_2 - h_2^{\text{ref}}\right)^2 \qquad \text{(4a)}$$

subject to:

$$\begin{aligned} &\text{nonlinear model defined in (3)} && \text{(4b)}\\ &h_1^{\min} \le h_1 \le h_1^{\max} && \text{(4c)}\\ &h_2^{\min} \le h_2 \le h_2^{\max} && \text{(4d)}\\ &u_1^{\min} \le u_1 \le u_1^{\max} && \text{(4e)}\\ &u_2^{\min} \le u_2 \le u_2^{\max} && \text{(4f)} \end{aligned}$$

That is, we use as a cost function the tracking of a predefined set point ($h_1^{\text{ref}}, h_2^{\text{ref}}$) subject to constraints on the inputs and on the states. π_1 weights the importance of tracking h_1 over h_2. Unless explicitly mentioned, we use $h_1^{\min} = h_2^{\min} = 0$ m, $h_1^{\max} = h_2^{\max} = 0.95$ m, $u_1^{\min} = u_2^{\min} = 0.01$ and $u_2^{\max} = u_2^{\max} = 0.95$. The valves are not allowed to open or close fully to avoid unmodeled behaviors. We choose a prediction horizon $N_p = 10$ steps with a sampling time $t_s = 1$ s and a weight $\pi_1 = 1.8$. This set of tuning parameters is chosen because it provides a satisfactory performance but their choice does not influence qualitatively the results (unless they are chosen very wrongly). We first test standard NMPC on the real plant using DO-MPC via the Data Acquisition (DAQ) system Labjack [33]. For this, we developed a simple interface to the application module in DO-MPC. We compare the experimental results with the simulations obtained with DO-MPC and the integrator CVODES instead of the real plant. Note that as one of the main features of DO-MPC, once the different interfaces are available, the only modification necessary to run NMPC on the real plant or with an integrator is to exchange the simulation/application module with the desired one. The results of this comparison are presented in Fig. 6. Here we consider a step change at time $t = 100$ s of h_2^{ref} from 0.28 m until 0.678 m while the setpoint $h_1^{\text{ref}} = 0.65$ is maintained constant throughout the experiment. It can be seen in Fig. 6 that the NMPC controller achieves the control task. Due to the measurement noise and to the imperfect behavior of the valves, the performance is slightly different in the real experiment and in the simulation. Note that once the desired performance has been achieved in simulations, the only necessary step to achieve the experimental results is to exchange the simulation module for the module that contains the interface with the real plant. In the following, we consider the case where a fault enters the system at a certain point. We assume that at time $t = 100$ s the real control input acting on the valve V7 system is disturbed so that $u_2 = \beta \cdot u_2$, where β is an uncertain parameter that can take values between {0.5, 1, 1.5}. This models the case where the valve jams or opens more than expected. At the same time, we set a constraint on the level in tank B2 ($h_2 \le h_2^{\max} = 0.7$ m) near to the setpoint to illustrate the advantages of multi-stage NMPC. As can be seen in Fig. 7, the use of standard NMPC results in violations of the constraints (marked with a dashed line) when the uncertain parameter β changes from 1.0 to 0.5 at time $t = 100$ s. In contrast, multi-stage NMPC is able to maintain the height of the tank within the bounds, because the different possible values of the uncertainty are taken into account in the scenario tree. For the multi-stage NMPC approach we consider a scenario tree that contains all the possible values of the uncertainty with the same probability ω_i and we branch the tree only in the first stage (robust horizon = 1). The controller is implemented on a netbook equipped with an Intel Atom processor at 1.7 GHz with 2 GB of RAM running Linux. The worst-case computation times per NMPC iteration are 0.2 s for the multi-stage NMPC case and 0.05 for the standard NMPC case.

VI. Conclusion

In this contribution, we presented a generic development environment for nonlinear model predictive control (called DO-MPC) that supports the development and testing of NMPC solutions. The elements of the controller can be specified independently and different numerical tools can

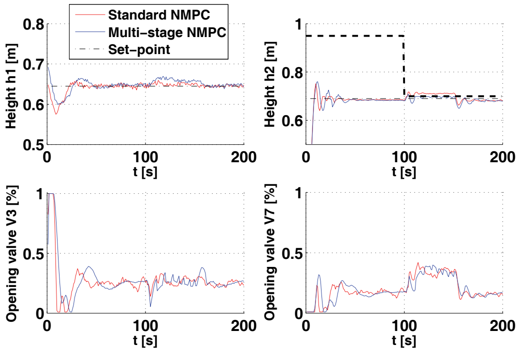

Fig. 7: Experimental results for the standard and multi-stage NMPC of the two-tank system when a step in the uncertainty parameter occurs at time $t = 100$ s. The constraint is indicated by a dashed line.

be used without re-coding of the model or the control formulation. Other algorithms can be easily connected to the DO-MPC environment. We use CasADi to calculate automatically and exactly the derivative information and to create shared libraries that contain this information and are used at run time for enhanced performance. It was demonstrated that using the environment, a robust NMPC solution based upon a stochastic multistage optimization formulation can be successfully implemented with efficient numerical algorithms that lead to computation times that are shorter than needed for a real-time implementation.

VII. Acknowledgements

We thank Joel Andersson for useful suggestions regarding the implementation with CasADi. The research leading to these results has received funding from the European Commission under grant agreement number 291458 (MOBOCON) and from the Deutsche Forschungsgemeinschaft under grant agreement number EN 152/39-1.

References

[1] S. J. Qin and T.A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11:733–764, 2003.

[2] R. Huang, V. Zavala, and L.T. Biegler. Advanced step nonlinear model predictive control for air separation units. *Journal of Process Control*, 19:678–685, 2009.

[3] E. Idris and S. Engell. Economics-based NMPC strategies for the operation and control of a continuous catalytic distillation process. *Journal of Process Control*, 22:1832–1843, 2012.

[4] IPCOS. IPCOS NV. http://www.ipcos.com, 2014.

[5] B. Pluymers, J. Ludlage, L. Ariaans, and W. Van Brempt. An industrial implementation of a generic NMPC controller with application to a batch process. In *Proc. of the 17th IFAC World Cogress*, 2008.

[6] Cybernetica. Cybernetica AS. http://www.cybernetica.no, 2014.

[7] B. Houska, H. Ferreau, and M. Diehl. An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range. *Automatica*, 47:2279–2285, 2011.

[8] M. Diehl, D. Leineweber, and A. Schäfer. *MUSCOD-II Users Manual*. IWR Preprint 2001-25, 2001.

[9] B. Houska, H.J. Ferreau, and M. Diehl. ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization. *Optimal Control Applications and Methods*, 32:298–312, 2011.

[10] R. Amrit and J. Rawlings. Nonlinear model predictive control tools (NMPC tools). Online report, 2008.

[11] Z.K. Nagy. OptCon - an efficient tool for rapid prototyping of nonlinear model predictive control applications. In *Proc. of the AIChE Annual Meeting*, pages 16–21, 2008.

[12] M Herceg, M. Kvasnica, C.N. Jones, and M. Morari. Multi-parametric toolbox 3.0. In *Proc. of the European Control Conference*, pages 502–510, 2013.

[13] P.J. Campo and M. Morari. Robust model predictive control. In *Proc. of the American Control Conference*, pages 1021–1026, 1987.

[14] P.O.M. Scokaert and D.Q. Mayne. Min-max feedback model predictive control for constrained linear systems. *IEEE Transactions on Automatic Control*, 43(8):1136–1142, 1998.

[15] D.Q. Mayne, M.M. Seron, and S.V. Rakovic. Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, 41:219 – 224, 2005.

[16] D. Q. Mayne, E. C. Kerrigan, and P. Falugi. Robust model predictive control: advantages and disadvantages of tube-based methods. In *Proc. of the 18th IFAC World Congress Milano*, pages 191–196, 2011.

[17] S. Yu, H. Chen, and F. Allgöwer. Tube MPC scheme based on robust control invariant sets with application to Lipschitz nonlinear systems. In *Proc. of the 50th IEEE Conference on Decision and Control*, pages 2650–2655, 2011.

[18] J. H. Lee and Z. H. Yu. Worst-case formulations of model predictive control for systems with bounded parameters. *Automatica*, 33(5):763–781, 1997.

[19] S. Lucia, T. Finkler, and S. Engell. Multi-stage nonlinear model predictive control applied to a semi-batch polymerization reactor under uncertainty. *Journal of Process Control*, 23:1306–1319, 2013.

[20] S. Lucia, J. Andersson, H. Brandt, A. Bouaswaig, M. Diehl, and S. Engell. Efficient robust economic nonlinear model predictive control of an industrial batch reactor. In *Prof. of the 19th IFAC World Congress*, pages 11093–11098, 2014.

[21] S. Lucia, T. Finkler, D. Basak, and S. Engell. A new robust NMPC scheme and its application to a semi-batch reactor example. In *Proc. of the International Symposium on Advanced Control of Chemical Processes*, pages 69–74, 2012.

[22] S. Lucia and S. Engell. Robust nonlinear model predictive control of a batch bioreactor using multi-stage stochastic programming. In *Proc. of the 12th European Control Conference, Zurich*, pages 4124–4129, 2013.

[23] J. Andersson, J. Åkesson, and M. Diehl. CasADi – A symbolic package for automatic differentiation and optimal control. In S. Forth, P. Hovland, E. Phipps, J. Utke, and A. Walther, editors, *Recent Advances in Algorithmic Differentiation*, Lecture Notes in Computational Science and Engineering, pages 297–307, Berlin, 2012. Springer.

[24] C. Schoppmeyer. GEMS - Generic EMBOCON Minimal Supervisor. https://github.com/EMBOCONcs/GEMS, 2013.

[25] A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward. SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software,*, 31:363–396, 2005.

[26] A. AlGhazzawi and B. Lennox. Model predictive control monitoring using multivariate statistics. *Journal of Process Control*, 19:314–327, 2009.

[27] R. S. Patwardhan, S. L. Shah, and K. Z. Qi. Assessing the performance of model predictive controllers. *Canadian Journal of Chemical Engineering*, 80:954–966, 2002.

[28] N. Zagrobelny, L. Ji, and J.B. Rawlings. Quis custodiet ipsos custodes? *Anual Reviews in Control*, 37:260–270, 2013.

[29] Qt5.1. Qt5.1 framework. http://www.qt-project.org, 2014.

[30] A. Shapiro. *Lectures on Stochastic Programming: Modeling and Theory*. SIAM, 2009.

[31] A. Wächter and L. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106:25–57, 2006.

[32] S. Lucia, J. Andersson, H. Brandt, A. Boauswaig, M. Diehl, and S. Engell. Handling uncertainty in economic nonlinear model predictive control: a comparative case-study. *Journal of Process Control*, 24:1247–1259, 2014.

[33] LabJack. LabJack U12: low Cost Multifunction DAQ with USB. http://www.labjack.com/u12, 2014.