TALLINN UNIVERSITY OF TECHNOLOGY
DOCTORAL THESIS
50/2018

Advanced Control of District Heating Processes in Estonia

VITALI VANSOVITŠ

TTÜ PRESS

TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies
Department of Computer Systems

This dissertation was accepted for the defense of the degree of Doctor of Philosophy in Informatics and System Engineering on July 4, 2018.

Supervisors:	Ph.D., Associate Professor Dr. Eduard Petlenkov Department of Computer Systems Tallinn University of Technology Tallinn, Estonia
	Ph.D., Associate Professor Dr. Kristina Vassiljeva Department of Computer Systems Tallinn University of Technology Tallinn, Estonia
Opponents:	Prof. Mikuláš Huba Slovak University of Technology in Bratislava Bratislava, Slovakia
	Associate prof. Celaleddin Yeroğlu Department of Computer Engineering Inonu University Malatya, Turkey
Defence of the thesis:	August 30, 2018

Declaration:
Hereby I declare that this doctoral thesis, my original investigation and achievement, submitted for the doctoral degree at Tallinn University of Technology has not been submitted for doctoral or equivalent academic degree.

Vitali Vansovitš

European Union
European Regional
Development Fund

Investing
in your future

____
signature

Copyright: Vitali Vansovitš, 2018
ISSN 2585-6898 (publication)
ISBN 978-9949-83-310-8 (publication)
ISSN 2585-6901 (PDF)
ISBN 978-9949-83-311-5 (PDF)

TALLINNA TEHNIKAÜLIKOOL
DOKTORITÖÖ
50/2018

Keskkütte soojuse tootmisprotsesside juhtimine

VITALI VANSOVITŠ

TTÜ KIRJASTUS

Contents

List of Publications

P1. V. Vansovits, E. Petlenkov, K. Vassiljeva, and A. Guljajev, “Identification of industrial water boiler for model predictive control of district heat plant,” in Proc. 13th Biennial Baltic Electronics Conf, Oct. 2012, pp. 315–318.

P2. V. Vansovits, E. Petlenkov, K. Vassiljeva, A. Tepljakov, and J. Belikov, “Application of MPC to industrial water boiler control system in district heat plant,” in Proc. 13th Int. Conf. Control Automation Robotics Vision (ICARCV), Dec. 2014, pp. 1609–1614.

P3. V. Vansovits, A. Tepljakov, K. Vassiljeva, and E. Petlenkov, “Towards an intelligent control system for district heating plants: Design and implementation of a fuzzy logic based control loop,” in Proc. IEEE 14th Int. Conf. Industrial Informatics (INDIN), Jul. 2016, pp. 405–410.

P4. V. Vansovits, B. I. Godoy, A. Tepljakov, K. Vassiljeva, and E. Petlenkov, “Model-based control design for a district heating plant,” in Industrial Informatics (INDIN), 2017 IEEE 15th International Conference on. IEEE, 2017, pp. 615–620.

Author’s Contribution to the Publications

All results in [P1]–[P4] were obtained by the author of the thesis under the supervision of Dr. Eduard Petlenkov and Dr. Kristina Vassiljeva.

In [P1] a nonlinear neural network model of the process was obtained and tested for possibility to use it for process simulation and model based control. It was concluded that the model demonstrates good prediction results within a limited operating area of the process it had been trained in, but can produce inaccurate or even infeasible predictions outside it, so it is too risky to use such model for practical model based feedback control implementation. It was decided to proceed with linear models in later publications.

The contribution of the author of the thesis in [P2] lies in heat production model identification, model predictive controller design and evaluation of the control system in a CACSD simulation environment. It was shown that this process can be efficiently controlled by MPC improving control quality and efficiency.

[P3] complemented the main study line with an investigation into the control of a process with unmodeled dynamics. Such process control is hard to build on model based scheme, so Fuzzy Controller was proposed and designed for the real biofuel boiler. Fuzzy Controller efficiency was shown on real process data.

The problem of implementation of MPC to real process control was considered in [P4]. MPC was designed as stand-alone application for this work and tested in simulation environment built in real-life distributed control system software. MPC was tested in conditions that are maximally close to the real-life process control and operation. MPC was compared with PI controller and it was shown that it performs better for considered type of process.

Chapter 1

Introduction

To PID or not to PID?
Vance VanDoren, PhD, PE

Design of efficient and reliable industrial applications is critical for reducing resource consumption and energy waste thereby ensuring ecological sustainability. However, although relevant advanced industrial control techniques are readily available, proportional-integral-derivative (PID) controllers massively enjoy unrivaled popularity in industrial process control applications. One of the reasons for the popularity of the common PID controller is its relative simplicity coupled with its applicability to a wide range of industrial control problems [3]. However, it is also a commonly acknowledged fact that only a fraction of the existing PI/PID controller based loops are tuned to achieve optimal performance [49]. Meanwhile, contemporary industrial applications are growing in complexity and demand a more advanced control scheme to be employed to ensure robustness and efficiency [73].

The work started here is concerned with control of industrial processes. The case considered is a water heating process for a district heat system that is part of a power plant located in Tallinn, Estonia. At the same time proposed solution is not limited for use only in that production unit, but can be applied to other processes of the same kind. After certain revisions and testing it is also applicable to any kind of the process.

During this work the process was studied, modeled and then controlled with a new type of control for existing control system. A model predictive controller (MPC) was developed as stand-alone application and applied to the process control in combination with existing PID controllers. This improved the process control performance significantly, showing at the same time that MPC is a good tool, but it is not able to tackle the complete multitude of challenges in overcoming physical drawbacks of the process control. As it is stated in many texts related to real industrial application, e.g. [15], the first step to improve process control is to check instrumentation and process itself for faults and design mistakes. Advanced control can only be efficient afterwards. We did not have the opportunity to make modifications in the real process, so we studied what results can be achieved with pure mathematical methods applied to existing non-perfect equipment.

As it was cited in the same survey [15]: "An important point to make concerning academic research is that the results are not just algorithms and software (many of which have not directly impacted industrial offerings), but the insight gained for what can and cannot be achieved for a given system and/or controller (Morari, 1997)."

1.1 State of the Art

Modern process control is a huge technological and engineering area with many layers solving various control tasks on different control levels.

The first level that interacts with the process directly is represented by an actuator-sensor layer. These are field devices that measure process physical parameters (e.g., temperature, pressure, flow) and directly affect these parameters with physical actions (such as valves, pumps, mills). To satisfy proper control requirements sensors have to be properly selected, calibrated and installed. Sensor is not able to reflect the real process state, if it is not suitable for the measured environment (e.g., different types of devices are used to measure steam and water flow, these are not replaceable), not calibrated (shows incorrect values) or is installed in the wrong position, where the parameter of the environment is not physically measurable. Also, the actuator should be properly selected and installed. If a valve is always used in the range below, e.g., 20% of its full operating scale, then it is clearly oversized and its sensitivity is low thus providing low quality of control. If a pump frequently works at its peak power, then clearly it is not powerful enough, causing control loop to saturate and degrade in control quality. Nowadays, there are clear rules for equipment selection [38], but according to the 14 year work experience of the author of the thesis as an automation engineer, wrong choices in selecting adequate instrumentation are mainly made due to either engineering mistakes or the motivation to reduce project costs by selecting less expensive equipment of lower quality.

The next level of process control is based on controllers or control applications normally done in PLC (programmable logic controller) or DCS (distributed control system), where sensors and actuators are connected with wires (input-output cards) or information buses or networks. The most widely used control law is still based on the conventional PID (proportional-integral-differential) controller. The reason is obvious — the PID controller is capable of handling the majority of control tasks reasonably well. Almost every DCS contains a PID auto-tuning mechanism, making its implementation easy even for people without control theory knowledge [7]. In the experience of the author, PI controllers are used in almost all possible process control applications, but according to the sources world wide the share of PID controllers is around 90 [2]. The D component is not frequently used, because it is mainly suitable for rapid processes only, so PI controller handles majority of control tasks of slow industrial processes.

The PID algorithm is suitable for controlling a single input single output process loop, so these serve as a departing point to the next level of process control applications where these single loops are connected to each other with higher PID cascades or custom logic applications or some advanced control techniques such as model based control or fuzzy control. This layer calculates optimal set points for the underlying level and is normally made as a part of DCS or a separate software product. Higher level is not regarded as control, but more as planning. It is normally some kind of Enterprise Resource Planning (ERP) System that sets production targets for the process [15, 53].

This thesis is focused on possible benefits of the third level of the hierarchy discussed above and is dedicated to model based control or model predictive control (MPC). It can also play on the level two substituting PID loops, but in industrial processes it is more suitable to optimize lower level loops by computing optimal set point [15, 53] (see Figure 1.1). Level one consists of sensors and actuators for temperature (TC), flow (FC), pressure (PC) nad other process parameters controls.

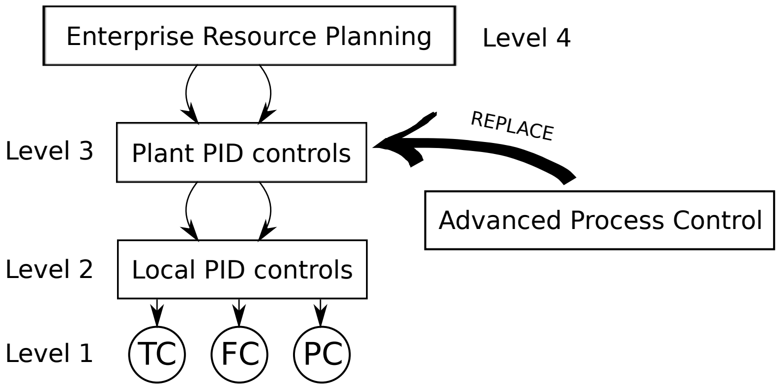


Figure 1.1: Process control hierarchy.

Model based control has a long history going down to 1960's [53], when linear quadratic Gaussian (LQG) controller was presented [4, 30]. It was an unconstrained controller based on state-space model with Kalman filter as observer.

The first real model predictive controller (MPC) systems were first implemented in 70's mainly in petrochemical industry. First publications on this topic appeared in the late 70-s [13,55], those introduced model predictive controllers of the first generation. They showed promising results in real control applications and caused excitement in the process control community. Multiple startup companies developed their own model base control algorithms and implemented them under various trademarks. The major problem of the algorithms of the first decade of MPC was their heuristic nature and lack of a theoretical base [37]. Richalet presented his IDCOM (Identification and Command) controller in 1976. Cutler and Ramaker implemented dynamic matrix control (DMC) without constraints in 1973, but already in 1980 it was endowed with constrained control.

Theoretical research and development of MPC controllers have reached a mature state in the 90's. Model predictive control theory was formulated, terms were unified, controller stability was proven, studies on robustness and nonlinear MPC were made [37]. The same decade was fruitful in terms of development of commercial MPC solutions. A number of related companies were purchased and merged which established the key players present in the market today: Aspen with DMCplus solution, Honeywell with RMPCT and Schneider with Connoisseur [53]. There are also other companies implementing MPC internally or offering it as a commercial product, but there is not much open information about them.

Nowadays MPC can be found in diverse industries such as: petrochemical, chemical [19,46,59], pharmaceutical, energy, food and many others. Vast development of processing power along with the proposal of new and efficient algorithms made it possible to implement model based control in other areas than industry with its slow processes. Nowadays, optimization can be performed within milliseconds which allows to use MPC in fast process control loops. The method finds its application in vehicle engine and suspension control, power electronics and drives control [69], automotive industry [17], power trains, building automation [26,41], power electronics [8,57,69], traffic control [16, 40] and many others, where control cycles go down to milliseconds. Good reviews of MPC development and implementation are given in [21, 25, 37, 53].

Classical approach described by the theory assumes formulation of MPC task as a quadratic programming problem and its solution with suitable optimizer. The general problem is stated mathematically as

$$\min_{\theta} \left(\theta^T \mathbf{\Phi} \theta + \phi^T \theta\right) \tag{1.1}$$

subject to

$$\mathbf{\Omega} \theta \leq \omega, \tag{1.2}$$

where θ is a vector of process inputs that should minimize cost function and $\mathbf{\Phi}$, ϕ, $\mathbf{\Omega}$ and ω are matrices and vectors of known parameters.

There are various ways to solve the problem. Active Set Method and Interior Point are two popular choices that perform well [11, 42]. Interior Point Method was proposed by Rao, Wright, and Rawlings [54]. It is characterized as being fairly easy to implement, yet performing very well. Its computational complexity is linear in the problem size as opposed to exponential complexity of the other methods. That is why interior point method is one of the most widely used to solve the MPC problem.

The following main MPC development directions are highlighted in [15, 18]. Nonlinear models are becoming more popular as majority of processes are nonlinear. Use of respective type of model can produce certain benefits in the form of more accurate control and better performance. Next interesting topic could be study of robust MPC. MPC for uncertain processes (Stochastic MPC) could also see further development, as they are closest to real processes that may have parameters and disturbances causing non-deterministic behavior [44]. Development of adaptive MPC is another concern promising many benefits in real life applications. Identification and modeling for MPC provides space for further development. One of the newest areas is distributed MPC, where optimization task is distributed between many decentralized computation nodes managing optimal control of the whole process at the same time.

Research area of this thesis is industry, where MPC development and implementation continues ever on. Major part of practical research is related to chemical industry with its complicated multi-variable nonlinear processes. Modern research expands classical approach looking for benefits in various non-standard solutions. Many papers consider a popular highly nonlinear continuous stirred tank reactor (CSTR) [6]. There are several research papers related to this process that propose the use of hybrid nonlinear models consisting of several linear models [35], compare use of linear and nonlinear MPC [34], propose robust control with incremental action [5]. Use of nonlinear MPC with neural network model is proposed in [51] for control of boiler unit. There are also multiple research on use of fuzzy logic in combination with MPC [14]. The general trend of today's scientific development is combining MPC algorithm with other control approaches. This is an interesting and perspective way, but it takes many years until novel ideas are tested and reach wide industrial adoption [20].

Energy production that is concerned in this work can also utilize multivariable capability of MPC. There are various research references of using model based control in energy production [32,36,45,76]. Normal way of MPC utilization in energy application is optimal set points calculation for lower level PID based control loops, thus providing optimal control for the whole process. At the same time it is possible to replace all PID loops with MPC, where the advanced controller produces control action directly to actuators.

Typical way of MPC implementation to the real process nowadays includes [15, 38]:

1. Pretest and preliminary MPC design.
2. Plant testing.
3. Model and controller development.
4. Commissioning and training.

In the pretest phase sensors and actuators are checked and low level PID controls are tuned for better performance. During the same step initial MPC design is performed: control targets are defined, controlled and manipulated variables selected. In plant testing phase manipulated variables are excited and model identification data is collected. In the third phase of modeling and control development process data is analyzed, process model is identified and a suitable controller is designed on the base of the acquired model. Initial tuning and testing is performed in a simulation environment. In the commissioning phase the controller is implemented to the real process, its behavior is observed and analyzed, final tuning is performed. Also process operating personnel is trained to use MPC in process control.

1.2 Motivation and Problem Statement

Energy industry is the largest and one of the most important in Estonia. On the background of rising prices for fuels it is extremely important to produce energy in the most efficient way. Proper control is vital for efficiency in any industry including energy production. Modern processes have normally wide range of parameters that have to be supervised and manipulated. It is hard to achieve good control with use of single input single output (SISO) controller like PID. PID is good for simple processes, but there can be multiple inputs multiple outputs (MIMO) loops in industry that require use of advanced PID techniques utilizing combination of many individual PID controllers. Such control loop is hard to tune and its efficiency is often under doubt.

MPC is a strategy to control loops with multiple inputs and outputs. By its nature it can optimize complicated loops and control process with maximal efficiency. First MPC solutions were implemented in 1970s. Nevertheless, still it is rarely used in industry. Variety of well known automation companies such as Aspentech, Honeywell and others [42] offer MPC solutions, but these are expensive and hard to implement, since this requires presence of highly educated experts from other countries.

In this thesis, the development of an MPC application is considered for use in Estonian industry with local implementation and support. The main idea is to increase efficiency of industrial processes in the country. MPC is a general concept, so it can be implemented not only in energy industry, but also in any other process area, such as chemical, water treatment, pulp and other industries [15, 18, 28, 53].

Although the MPC has known advantages, there are also some drawbacks that limit its use in every possible application. One limitation is the necessity to obtain a coherent process model that can be impossible in some cases, e.g., when certain relevant process parameters are not measured. To mitigate this drawback we review another advanced control technique before considering the MPC approach. Specifically, we apply fuzzy control that does not require a model, but only good practical knowledge about the inner functioning of the process. The implementation of fuzzy control is described in this work as an alternative to the MPC solution and a comparison of advantages and drawbacks of these control techniques is provided.

In the present work, a small biofuel boiler house is considered as an example of the process with unmodeled dynamics due to missing fuel quality measurement. At the same time PID control efficiency is low there as control loop has one process input, but two process outputs. Fuzzy control implementation is considered there as efficient method to solve such control tasks.

To support further development of the topic of advanced control, a combined heat plant (CHP) is considered as an example of energy production unit with modeled dynamics. Plant production facilities include a waste to energy (WtE) power unit that produces heat and electricity, and a gas-fueled water boiler that produces only heat. Power plant is controlled by means of DCS that can maintain a high level of automation. WtE unit produces 50 MW of heat power at its maximum capacity. When this heat is below city demand, a water boiler is started. Water boiler produces up to 116 MW of heat. Power unit is normally operated at a constant load and water boiler produces variable heat depending on the district heat (DH) network's demand. The biggest challenge is to control the water boiler so that it could change the power production as fast as DH network load requires. At the moment of starting this work, boiler heat production was controlled with a cascade PI-controller consisting of three cascades: plant outlet water temperature control, water boiler outlet temperature control and boiler gas flow control. This loop had a very slow response and it could not handle process disturbances of the plant, so plant output had significant deviations from specification limits. In addition, the loop was tuned for control of the process in a certain operating point close to the maximum load of the boiler. Most of the time, the boiler is operated with the power production below half of the load, where existing control is not functional at all. Control loop is operated manually in such cases causing out-of-specification production.

The PI includes 3 cascades as shown in Figure 1.2.

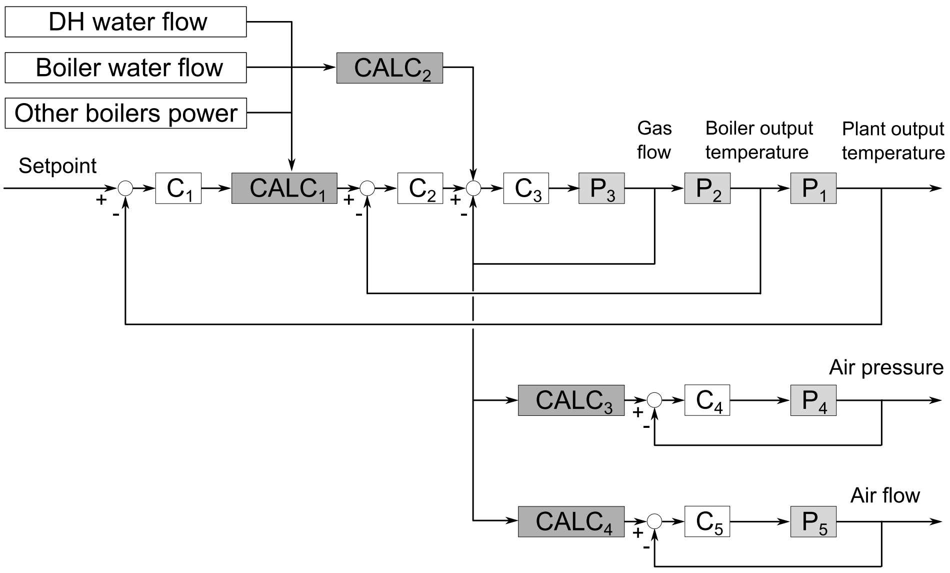


Figure 1.2: Controller layout.

The upper cascade C_1 controls output temperature of the water leaving the plant towards district heat network — process P_1. C_1 output is used in calculation block $CALC_1$ to compute set point for the next cascade C_2. The second cascade controls output temperature of the boiler - process P_2. Its output is used to compute set points for lower cascade. Lower cascade loops C_3, C_4 and C_5 control gas and air flow by manipulating related valves and variable speed fans. Additional calculation block $CALC_2$ is used to take into account other process parameters: DH water flow through the plant, water flow through the boiler and power of other water boilers if any is running. Loops 3-5 are fast, Loops 1-2 are much slower compared to them. See Table 1.1 for all parameters' values of the controllers.

Cascade PI controller is used in the process to build a complex MIMO control. According to [1, 6] it can be successful due to significant difference in time constants between processes P_2 and P_3, where process P_3 is much faster. And yet time constants of the processes P_1 and P_2 are comparable, so these cascades do not produce good performance. The control task is mainly solved by additional calculation blocks than by tuning upper PI cascades. In this configuration controller was tuned during control system commissioning in 2008. Since then it has been working without modifications.

PI control loops with similar characteristics can be found not only in the considered plant, but almost everywhere in the energy industry. Sometimes their unsatisfactory behavior is caused by improper tuning (if appropri-

Table 1.1: Controllers tuning parameters.

Control loop	$\mathbf{K_p}$	$\mathbf{T_i}$
Air pressure 1 (C_4)	0.2	40
Air pressure 2 (C_4)	0.2	40
Air flow 1 (C_5)	0.8	18
Air flow 2 (C_5)	0.3	20
Air flow 3 (C_5)	0.4	29
Gas flow (C_3)	0.25	20
Boiler output (C_2)	0.9	240
Plant output (C_1)	0.5	180

ate tuning is possible at all), sometimes by wrongly selected control devices (valves, actuators etc.), sometimes by technological mistakes. Also PID controller is still the main control tool, but it is not able to solve efficiently some control tasks, e.g. related to the processes with delay, MIMO processes. So, there is an ample space to study this industry for possible benefits of advanced process control and offer solutions to improve the situation overall. Certainly this cannot solve all existing problems widely spread in practical industrial control, but it can mitigate some consequences of technological mistakes. Surely, it can improve performance of the complicated loops, where PID controller does not perform well enough. Model based control was selected for these purposes as one of the most efficient advanced control techniques. It also goes in line with Industry 4.0 modern trends with its big data processing and process optimization. Model based control can utilize huge amount of process data to update models periodically improving control quality and rising production efficiency to unprecedented highs.

According to Statistics Estonia there were 3430 boilers in Estonia by the end of the year 2016 [60]. 25% of these are biofuel boilers and 41% are gas boilers. Biofuel and gas boilers utilized in district heat generation produce 77.6% of all the district heat [62]. Development of advanced process control solution for these processes has huge implementation potential providing sustainability and high efficiency of the heat production.

The main goals of this thesis are as follows:

- Improve significantly performance of the current processes. The precise criteria depends on the particular process and are discussed in the corresponding sections of the thesis.
- Reduce participation of the human factor and increase automation level of the heat production facilities.
- Implement developed advanced proceses control (APC) solutions in the real processes thus proving efficiency rise in practice.

- Analysis of implementation results producing recommendations for APC use in DH generation facilities.

1.3 Author's Contributions

To the best of the author's knowledge, the stated contributions are novel to implementation of model based control for energy production industry in Estonia:

- Analysis of existing heat energy production control loops in Estonian industry for benefits of model based control use. Comparison of processes with modeled and unmodeled dynamics.

- Investigation and analysis of existing control systems' drawbacks influencing the quality and efficiency of energy production.

- Analysis of advanced control strategies and their applicability to different types of District Heating Plants in Estonia. Selection of suitable control technique depending on process characteristics.

- Mathematical modeling—identification of dynamic system models—and CACSD based simulation of the process and closed loop control system.

- Analysis of control strategies based on computer simulations.

- Practical implementation of the designed controller, analysis of the results, and formulation of expert advice towards improving the quality of existing feedback control loops.

1.4 Thesis Outline

The thesis includes seven chapters.

Chapter 2

The control of processes with unmodeled dynamics is considered. The chapter gives example of a fuzzy controller application as alternative to MPC, when modeling of the process is not possible due to unmeasured process variables. A control system based on Takagi-Sugeno approach is designed and implemented on in a live control situation. Results of this integration are provided and analyzed.

Chapter 3

The chapter gives overview of process model identification techniques and includes detailed description of the considered process of CHP together with its mathematical modeling.

Chapter 4

Description of model predictive control theory used in current work is provided. Employed algorithms and methods are described.

Chapter 5

In this chapter, the designed MPC application, design of process simulation environment and preliminary off-line MPC testing in simulation are introduced.

Chapter 6

The chapter provides detailed description of MPC application implementation to the considered process, analysis of the results and list of benefits that can be achieved in such kind of processes.

Chapter 7

In the final chapter, results of the thesis are reviewed and overall conclusions are drawn.

Chapter 2

Control of Unmodeled Processes

While considering the processes for model based control we encountered also such processes, where it is not always possible to fully identify the process. If there are unmeasured disturbances that affect process behavior significantly, then acquired model does not produce the same output as the process because of unmodeled dynamics. In this case model based control would produce poor results, so other control methods should be applied. PID itself does not use process model knowledge and is suitable for basic control. As we consider more complicated processes, where PID performance becomes unsatisfactory (certain criteria depends on the process), then we have to search for other techniques. These are numerous like adaptive control, expert systems, artificial neural networks, evolutionary algorithms etc. [18]. Some of them are used in industry widely, others are in development. We selected fuzzy control, as it does not rely on the model, but on knowledge of how process should be operated [38, 50, 61] and it already has tools for industrial implementation.

In Estonian industries, processes with unmodeled dynamics could be encountered in fossil fuel (oil shale) and biofuel boiler houses and power plants. Fossil fuel and biofuel can have various calorific values and humidity that affect heat production process directly. These parameters are not normally measured on-line, so it is not possible to use this information in control. At the same time, the influence of these parameters to the process is so high that model based prediction without taking these into account can be completely erroneous.

This chapter considers the case of using fuzzy control in one of Estonian boiler houses running on biofuel [67].

2.1 Process description

As an example of a process with unmodeled dynamics we consider a process control problem of a boiler house plant located in Rapla town near Tallinn city in Estonia. The boiler house includes three gas boilers and one biofuel (wood chips) boiler. Gas boilers 1 and 2 are currently not in use, so they are not discussed in the present study. Main production facility of the boiler house is biofuel boiler (number 4) as it is more economically efficient due to lower biofuel price compared to natural gas price.

The biofuel boiler is operated permanently during the heating season except for the maintenance periods. It can produce up to 5 MW of heat power. If more heat power is required, then a gas boiler is turned on in parallel with the biofuel facility.

Boiler house layout (excluding boilers 1 and 2) is presented in Figure 2.1.

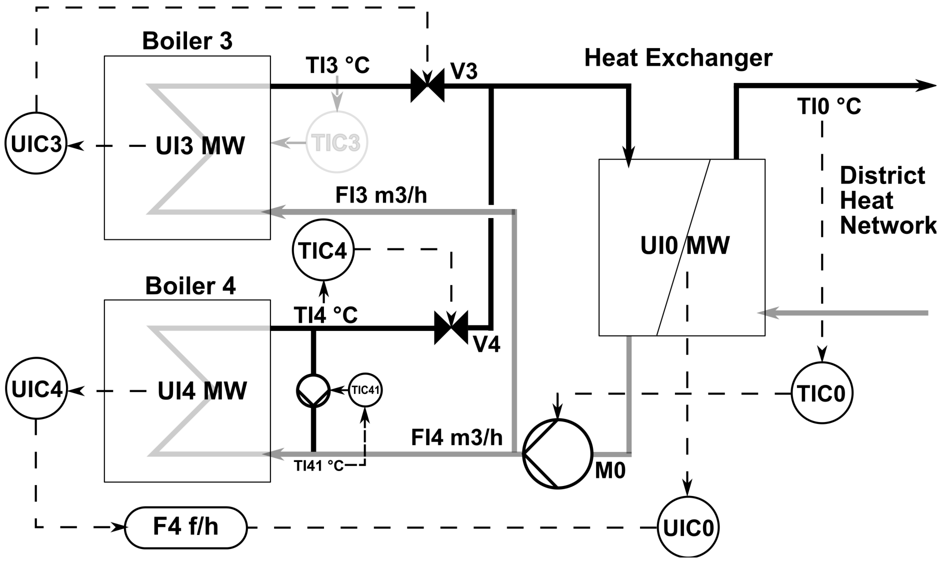


Figure 2.1: Boiler house control layout.

There is an internal water pipeline where water circulates between heat exchanger and boilers. The heat exchanger is a connection point between internal and external pipelines. External pipeline is used for water circulation between the boiler house and the district-heat (DH) network where heat power is delivered to households and industrial consumers.

The biofuel boiler has a grate furnace. Fuel feeding speed can be adjusted by changing the frequency of movement of a hydraulic piston that pushes fixed amount of fuel into the furnace each time.

Boiler house automation is made on the base of a DCS that supports fuzzy control implementation. Information exchange (measurements, control signals) with sensors and actuators is performed through input/output cards, Profibus and Modbus interfaces. All the controllers and logic algorithms are implemented as application programs. The DCS controls boiler 4 and pipeline operations. Each gas boiler has its own simple control based on a PLC. PLC control applications adjust gas flow to keep boiler's outlet water temperature on a certain set point value. There is no data exchange between PLC and DCS, so it is possible to adjust gas boiler's heat production only by manipulating the water flow control valve V3.

There are two scenarios for boiler house operation. The first one is applied when DH network demands less than 5MW of heat power and boiler 4 can be operated alone and produce required amount of heat. Control loop TIC0 keeps boiler house water outlet temperature near the set point. Set point depends on outer air temperature as shown in Table 2.1. A conventional PI controller is used to control boiler house outlet water temperature. Controller signal is supplied to internal pipeline pump frequency converter. Higher frequency leads to the faster pump rotation speed rising water flow through the boiler 4. The higher the flow the more heat is transferred to the heat exchanger.

Table 2.1: Boiler house outlet water temperature dependency on outer air temperature. Intermediate values are linearly interpolated.

Outer air temperature	Boiler house outlet temperature
−22°C	95°C
0°C	69°C
20°C	65°C

Heat power supplied to the DH network is measured and connected to PI controller UIC0 as a controlled variable. This controller keeps boiler 4 heat power production near the set point by adjusting the amount of fuel feeds per hour. Set point is calculated according to the following equation:

$$U_{SP} = c_w f_{w0}(T_{SP} - T_{in}), \tag{2.1}$$

where U_{SP} is DH network required heating power, c_w is water heat capacity, f_{w0} is DH network water flow, T_{in} is boiler house input water temperature and T_{SP} is TIC0 set point.

Boiler 4 has recirculation pipeline that passes part of boiler 4 outlet water to its inlet mixing with water coming from the heat exchanger. This is needed to keep the temperature in all points inside the the boiler above 100°C and prevent moisture from the fuel to condensate on boiler walls on the furnace side causing corrosion. There is a specific limitation related to this. Boiler's outlet water temperature has to be above 110°C. If it is below this value, then recirculation flow is not enough to keep boiler inlet temperature greater than 100°C. At the same time outlet temperature cannot be above 120°C, because materials of the boiler and pipelines are not designed for long term operation in such conditions. This can decrease boiler's lifetime and increase maintenance costs.

As UIC0 PI controller can keep only one measurement near the set point, there is additional logic that prevents the controller to decrease its output when outlet water temperature is below 115°C and goes down from one side and to increase its output when the outlet water temperature is above the same value and rises from another.

This control keeps outlet water temperature fluctuating in the range between 110°C and 120°C. At the same time boiler house outlet water temperature deviates from the set point by 1°C, which is acceptable.

When DH network demand grows above 5MW the gas boiler 3 has to be turned on to provide additional heat.

In this scenario control loop for boiler house outlet water temperature is the same TIC0, but boiler heat power production is now managed by another PI controller that has a static set point set by the operator. Normally its value is the maximum capacity of the boiler 4 — 5MW. Remaining required heat is produced by the boiler 3. There is a power controller UIC3 that controls water flow through the boiler. Boiler 3 has a separate PLC that keeps outlet water temperature near the set point 110°C. Produced heat power can be controlled from DCS by only adjusting valve V3 position.

This kind of control did not work well due to hydraulic dependency of water flows between boilers 3 and 4. When DH network demand is over 5 MW, boiler 4 works on its maximum capacity and boiler 3 produces remaining heat. If DH network demand goes down, UIC3 adjusts V3 position in closing direction. Water flow through the boiler 3 decreases, but due to hydraulic dependency flow rate increases through the boiler 4 and washes out more than 5 MW of heat from it. As a consequence, boiler 4 outlet water temperature falls down. At the same time heat power set point of boiler 3 decreases more, because boiler 4 heat production measurement goes above 5 MW. Valve V3 closes more because of this reason increasing boiler 4 flow more. At the same time UIC4 additional logic forces it to increase feed rate to rise boiler's outlet water temperature causing boiler to function above its maximum capacity.

An additional control loop TIC4 was used for testing purposes to control boiler 4 outlet water temperature by adjusting valve V4, but this caused other problems in process control. In the same situation when DH network demand decreases flow through the boiler 4 rises, temperature falls down and valve V4 starts to close under control of TIC4. This causes increase of the flow through the boiler 3 and valve V3 continues to close. Because of this race condition the whole flow through the boilers decreases, boiler house outlet temperature starts to fall down. TIC0 increases speed until achieves 100%. Saturated control loop cannot keep boiler house outlet water temperature close enough to the set point. In addition, pump motor working with 100% speed consumes maximum electricity causing increase of own costs of the boiler house production process. Race condition between TIC4 and UIC3 ended on approximately equal heat production in boilers 3 and 4. In case of DH network demand of 7MW it meant not full load of boiler 4 and use of extra gas fuel in boiler 3 due to this.

2.2 Fuzzy controller

It would be possible to use additional logic to try to keep both process parameters—boiler 3 heat production and water flow through the boiler 4—on desired level, but at the same time it is a good opportunity to apply fuzzy controller that is natural for multiple input multiple output (MIMO) systems. At the same time it is not reasonable to use model based control here, because this is the process that is difficult to model properly. Biofuel is delivered by different vendors and it has different calorific values and moisture that are not measured online. As these parameters are very important for process modeling, so without these measurements we cannot get model with satisfactory prediction quality. Process behavior would be random comparing to any acquired model prediction. That is why fuzzy control is selected here, because it does not rely on uncertain parameters, but takes into account only measurable values. It is possible to set rules that cover all possible states of the process thus providing desired control in the whole operating range of the boiler.

Our target is to decrease heat production of boiler 3 when the needs of DH network decrease. We are doing it by closing valve V3 and decreasing the flow through the boiler 3. At the same time we would not like to increase flow through boiler 4. So, we have system with two inputs and one output—valve V3.

It was decided to apply a simple fuzzy controller for this case using three triangular membership functions (MSF) for each input and five actions for output. There are nine rules used (see Table 2.2). This combination showed sufficient control efficiency later, so there were no reasons to make controller structure more complicated.

Simple triangular MSF are selected for this application (see Figure 2.2).

The first input is heat power of the boiler 3. It is changing in time, so MSF has to be changeable. To build the MSF we define two parameters—set point (SP) and MSF width. Using these parameters we build three MSFs. Middle one is “OK”, left one for low values of measurement and right one for

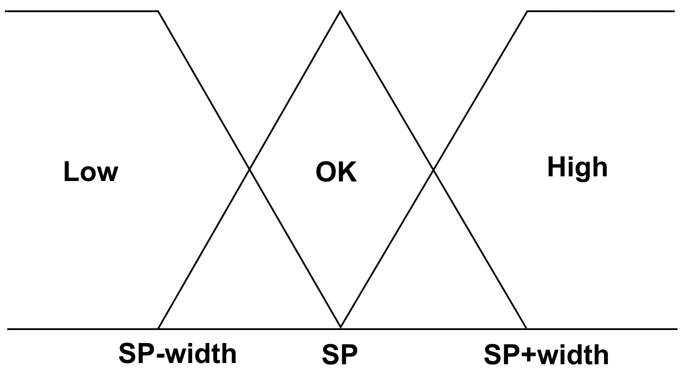


Figure 2.2: Membership function

high values. Left MSF has value 1 from minus infinity to SP-width and then decreases linearly to 0 by achieving SP. "OK" MSF increases linearly from 0 to 1 while moving from SP-width to SP and decreases linearly from 1 to 0 while moving from SP to SP+width. Right MSF increases linearly from 0 to 1 while moving from SP to SP+width and then has value 1 until infinity. Set point value for the first input is calculated on the base of DH network needs:

$$U_3 = U_0 - U_4, \tag{2.2}$$

where U_4 is boiler 4 heat production, U_0 is DH network demand, U_3 is required boiler 3 heat production. MSF width is defined manually as 0.4.

Same mechanism is applied to the second input—water flow through the boiler 4. Required value is calculated on the base of other process parameters:

$$f_{SP} = \frac{U_4}{c_w(T_{out} - T_{in})}, \tag{2.3}$$

where U_4 is boiler 4 heat power, c_w is water heat capacity, T_{in} is boiler 4 input water temperature and T_{out} is boiler 4 outlet water temperature. So, set point value is equal to f_{SP} and MSF width we define manually as 14.

As it was mentioned earlier, closing of the gas boiler outlet flow valve leads to reduction of water flow and heat production, because gas boiler's PLC control keeps outlet temperature constant. Less flow with the same temperature means less heat and vice versa. At the same time decreased flow through the boiler 3 while internal pipeline pump speed is constant causes increase of flow rate through the boiler 4. Taking these facts into account it is possible to describe following reasoning for definition of the fuzzy rules.

There are five possible actions defined: *remove much*, *remove little*, *no action*, *add little*, *add much*. Following nine rules are defined:

Rule 1. When boiler 3 produces less power than needed and flow through the boiler 4 is too low then no actions are needed. Due to low heat produc-

Table 2.2: Fuzzy rules.

Rule	Input 1	Input 2	Action
1	Low	Low	No action
2	Low	OK	Add little
3	Low	High	Add much
4	OK	Low	Remove little
5	OK	OK	No action
6	OK	High	Add little
7	High	Low	Remove much
8	High	OK	Remove little
9	High	High	No action

tion boiler house outlet water temperature will drop and TIC0 will increase internal water flow pump M0 speed. Flow through the boilers 3 and 4 will rise and boiler 3 will increase heat production as well.

Rule 2. If boiler 3 heat production is low and boiler 4 water flow is OK then it is possible to open valve V3 a bit to increase boiler 3 heat production while affecting boiler 4 water flow in a minor way.

Rule 3. If boiler 3 heat production is low and boiler 4 water flow is high then there is obvious need to open valve V3 much to increase boiler 3 heat production and decrease boiler 4 water flow.

Rule 4. If boiler 3 heat production is OK and boiler 4 water flow is low then valve V3 can be closed a bit to increase boiler 4 water flow without affecting boiler 3 heat production too much.

Rule 5. If both inputs are OK then no actions are needed.

Rule 6. If boiler 3 heat production is OK and boiler 4 water flow is high then valve V3 can be opened slightly to decrease boiler 4 water flow without affecting boiler 3 heat production too much.

Rule 7. If boiler 3 heat production is high and boiler 4 water flow is low then valve V3 needs to be closed much to shorten boiler 3 heat production and increase boiler 4 water flow.

Rule 8. If boiler 3 heat production is high, but boiler 4 water flow is OK, valve V3 can be slightly closed to reduce boiler 3 heat production without affecting boiler 4 water flow too much.

Rule 9. When boiler 3 produces more power than needed and flow through the boiler 4 is too fast, no actions are needed. Due to excessive heat generation boiler house outlet water temperature will rise above the set point and

TIC0 will decrease internal water flow pump M0 speed. Flow through the boilers 3 and 4 will be diminished and boiler 3 will decrease heat production as well.

Sugeno type fuzzy inference [61] is selected to produce fuzzy controller output signal. “Product” AND function is used to calculate firing strengths s_i of each rule i (i = *1, ...,* N, where N is number of rules). Rule table defines action a_i for each rule, where every action has its own output level $l(a_i)$. Final control *con* is calculated according to (2.4).

$$con = \frac{\sum\limits_{i=1}^{N} w_i l(a_i)}{\sum\limits_{i=1}^{N} w_i} \tag{2.4}$$

Controller functions in incremental mode, *con* defines change of controller output per second.

Output levels are selected by try & error method. Their values are shown in Table 2.3.

Table 2.3: Output levels.

	Action	Output level
1	Reduce much	-0.12
2	Reduce little	-0.05
3	No action	0
4	Add little	0.05
5	Add much	0.12

2.3 Results

As it was written earlier, before implementation of fuzzy control there was a PI controller for boiler 3 heat power control. It regulated heat production without paying attention to water flow to boiler 4. As boilers 3 and 4 are hydraulically dependent, manipulation of boiler 3 valve caused increase of boiler 4 water flow leading to decrease of boiler 4 outlet temperature. Additional logic of boiler 4 heat power controller prevents it from decreasing fuel feed rate, if outlet temperature is low. Due to this reason boiler 4 was producing 6MW of heat power for a few hours (see Figure 2.3a), which is not allowed from technological point of view. Maximum heat production can be

5.5MW for short period of time. Because of long overload boiler 3 heat power controller was turned into manual mode. Heat production of boiler 3 was controlled by the operator in manual mode for few days until implementation of fuzzy controller.

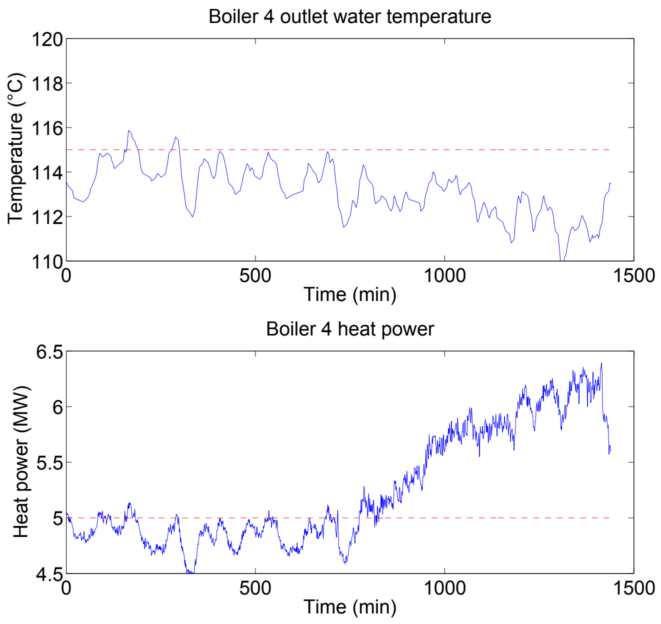
After implementation of fuzzy logic boiler 3 heat power controller started to take into account flow through the boiler 4 keeping heat production as close as possible to the set point and supplying suitable amount of water to boiler 4 for its proper operation on maximum load. Since this moment all the controllers were switched to auto mode keeping all the process parameters close to their set points. Boiler 4 heat production was close to 5MW and outlet temperature close to 115°C (see Figure 2.3b).

In the bottom trend of Figure 2.3b it can be seen that boiler 4 heat production variation frequency reduced since the middle point of the timeline. This was caused by decreasing firing strengths of fuzzy controller rules. Probably it is worth to lower these strengths more to make variations smoother. Unfortunately changed weather conditions did not allow to test modifications immediately. DH network demand is less than 5MW and only boiler 4 is in operation nowadays.

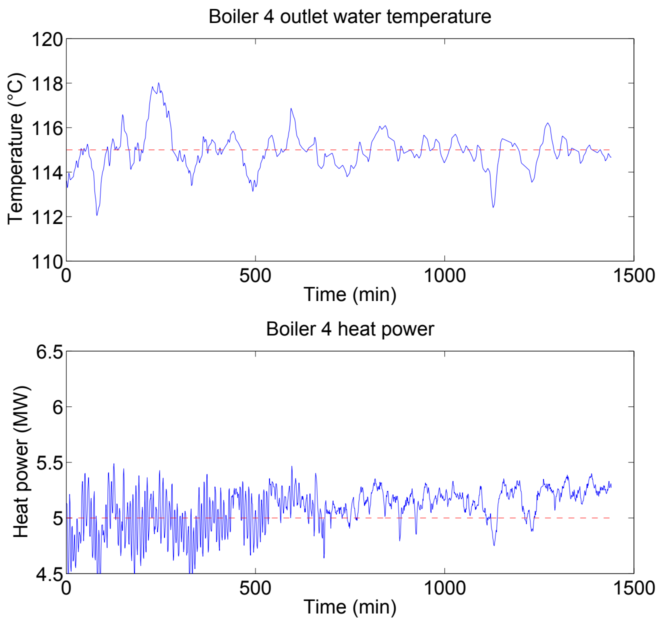
2.4 Conclusions

In this chapter we proposed and implemented control for a district heating plant with unmodeled dynamics. The main contribution of the chapter was the design and practical implementation of an industrial fuzzy controller for a heating plant. The industrial application was tested and was found to improve the performance of the underlying control loop which is supported by experimental evidence. In particular, the new controller allows to properly fulfill the control task, keep the equipment in the correct operating point thus prolonging its lifespan, and prevent the necessity for frequent manual control override. Therefore, the requirements for the controller put forth by a specific heating element and described in this work are satisfied.

The main concern of this thesis is model based control, so we continue by studying the process that can be modeled. Fuzzy control considered in this chapter is efficient tool, but its implementation becomes more complicated when number of process inputs increases. Since the amount of fuzzy controller rules grows according to a power law of membership functions of inputs, manual work on controller design becomes significant. For example, for three inputs with five membership functions each, 125 rules have to be defined. Model predictive control does not suffer from bigger number of inputs. It requires good enough process model and limited amount of suitable tuning parameters.



(a) Heat production before implementation of fuzzy control.



(b) Heat production after implementation of fuzzy control.

Figure 2.3: Comparison of boiler 4 control loop performance.

Chapter 3

Identification of the Process Model

The chapter describes the main process model identification methods used to acquire suitable model for use with MPC in later chapters.

Construction of a good model is a key factor in model based control techniques, while at the same time it is not the main goal, but an auxiliary action. Thus considering modeling for process control we should not try to achieve exact process-like behavior at any price, but build a suitable model that is able to provide high process control performance. It is often possible to achieve good results in real-life applications with use of relatively simple models that reflect basic dynamic characteristics of the process [22]. In case of real process control the feedback is always available. It helps to bring model states in consistency with reality by using a state observer. In these circumstances a key factor is the prediction of correct process dynamical behavior rather than exact prediction of the process output. Model quality is then directly related to process control quality—if the controller manages to keep output value within acceptable range around the set point, then the model can be seen as suitable for control purposes, even if it can not predict exact process output in simulation mode.

Throughout the text of the chapter, the following notation related to submatrix definition is used. Given a matrix $\mathbf{Q} \in \mathbb{R}^{n \times m}$, a submatrix $\mathbf{S} \in \mathbb{R}^{n_1 \times m_1}$, $n_1 \leqslant n$, $m_1 \leqslant m$ having the elements $q_{i,j}$, $i \in [i_l, i_h]$, $j \in [j_l, j_h]$ of the original matrix $\mathbf{Q}$ such that

$$\mathbf{S} = \begin{bmatrix} q_{i_l,j_l} & q_{i_l,j_l+1} & \cdots & q_{i_l,j_h} \\ q_{i_l+1,j_l} & q_{i_l+1,j_l+1} & \cdots & q_{i_l+1,j_h} \\ \vdots & \vdots & \ddots & \vdots \\ q_{i_h,j_l} & q_{i_h,j_l+1} & \cdots & q_{i_h,j_h} \end{bmatrix}$$

shall be denoted as

$$\mathbf{S} = \mathbf{Q}^{[i_l \dots i_h, j_l \dots j_h]}.$$

3.1 Identification methods

In what follows, system identification methods used in this work for modeling the studied system are reviewed. This section is restricted to state-space model identification by subspace and prediction error minimization methods.

The discrete time state-space model considered herein has the following form:

$$\begin{array}{r} x(k+1) = \mathbf{A}x(k) + \mathbf{B}u(k) + w(t) \\ y(k) = \mathbf{C}x(k) + \mathbf{D}u(k) + v(t) \end{array}, \tag{3.1}$$

where $u(k) \in \mathbb{R}^{n_u}$, $x(k) \in \mathbb{R}^{n_x}$, $y(k) \in \mathbb{R}^{n_y}$, $w(k) \in \mathbb{R}^{n_x}$ and $v(k) \in \mathbb{R}^{n_y}$ are the input, state, output, input noise and output noise vectors of the system, respectively.

The purpose of estimation is to find suitable matrices **A**, **B**, **C** and **D** to build a model that is able to predict process behavior. Matrix **D** is assumed to be zero except some special cases, as normally there is no feed-through from the inputs to the outputs.

One of the basic model identification techniques is the subspace method. The method is considered in multiple papers [29,63,64,70]. It is not iterative and it takes short time to estimate model by use of QR-decomposition and singular value decomposition (SVD). The only parameter that is needed to be set prior to estimation by the algorithm N4SID [64] is the order of the model. There are freely defined matrices used in this method that are automatically initialized by the algorithm. Although it is easy to identify a model with this method, it shows generally weaker results compared to prediction error minimization (PEM) method considered further.

In what follows, p is output dimension, m is input dimension, n denotes the number of states.

In subspace method [39] first matrix **G** is computed:

$$\mathbf{G} = \frac{1}{N}\mathbf{Y}\Pi^{\perp}_{\mathbf{U}^T}\mathbf{\Phi}^T, \tag{3.2}$$

where $N + r - 1$ is number of data samples, r is maximal prediction horizon, Y is a matrix of output sub-spaces, U is a matrix of input sub-spaces:

$$\mathbf{Y} = \begin{bmatrix} y(1) & y(2) & \cdots & y(N) \\ y(2) & y(3) & \cdots & y(N+1) \\ \vdots & \vdots & \ddots & \vdots \\ y(r-1) & y(r) & \cdots & y(N+r-1) \end{bmatrix}$$

and

$$\mathbf{U} = \begin{bmatrix} u(1) & u(2) & \cdots & u(N) \\ u(2) & u(3) & \cdots & u(N+1) \\ \vdots & \vdots & \ddots & \vdots \\ u(r-1) & u(r) & \cdots & u(N+r-1) \end{bmatrix}.$$

$\Pi^{\perp}_{\mathbf{U}^T}$ is a geometric operator that projects a row space of a matrix onto the orthogonal complement of the row space of the matrix $\mathbf{U}$ [63]:

$$\Pi^{\perp}_{U^T} = \mathbf{I} - \mathbf{U}^T(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{U}.$$

$\mathbf{\Phi}$ is defined as:

$$\mathbf{\Phi} = \begin{bmatrix} \varphi_s(1) & \varphi_s(2) & \cdots & \varphi_s(\mathrm{T}) \end{bmatrix}, \tag{3.3}$$

where $\varphi_s(t)$ is vector of output and input samples and s is a variable parameter normally set equal to r:

$$\varphi_s(t) = \begin{bmatrix} y(t-1) & \cdots & y(t-s) & u(t-1) & \cdots & u(t-s) \end{bmatrix}^T. \tag{3.4}$$

Then weighting matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ are defined

$$\mathbf{W}_1 = \mathbf{I},\ \mathbf{W}_2 = (\frac{1}{N}\mathbf{\Phi}\Pi^{\perp}_{\mathbf{U}^T}\mathbf{\Phi}^T)^{-1}\mathbf{\Phi} \tag{3.5}$$

and SVD performed:

$$\hat{\mathbf{G}} = \mathbf{W}_1\mathbf{G}\mathbf{W}_2 = \mathbf{U}\mathbf{S}\mathbf{V}^T \approx \mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^T, \tag{3.6}$$

where $\mathbf{S}_1$ approximation includes only n the most significant values of singular values while others are set to zero (noise effect elimination).

A full rank matrix $\mathbf{R}$ is selected then and used to define observability matrix:

$$\hat{\mathbf{O}}_r = \mathbf{W}_1^{-1}\mathbf{U}_1\mathbf{R}. \tag{3.7}$$

Matrices $\hat{\mathbf{C}}$ and $\hat{\mathbf{A}}$ are found from:

$$\hat{\mathbf{C}} = \hat{\mathbf{O}}_r^{[1 \ldots p, 1 \ldots n]}, \tag{3.8}$$

$$\hat{\mathbf{O}}_r^{[p+1 \ldots pr, 1 \ldots n]} = \hat{\mathbf{O}}_r^{[1 \ldots p(r-1), 1 \ldots n]}\hat{\mathbf{A}}. \tag{3.9}$$

Matrix $\hat{\mathbf{B}}$ and state vector $\hat{x}_0$ are found from:

$$\underset{\hat{\mathbf{B}},x_0}{\arg\min}\frac{1}{N}\sum_{t=1}^{N}||y(t)-\hat{\mathbf{C}}(q\mathbf{I}-\hat{\mathbf{A}})^{-1}\hat{\mathbf{B}}u(t)-\hat{\mathbf{C}}(q\mathbf{I}-\hat{\mathbf{A}})^{-1}x_0\delta(t)||^2, \quad (3.10)$$

where $\delta(t)$ is the unit pulse at time instance $t = 0$.

Another identification method of the state-space models PEM was presented by Ljung in 1987 [39]. PEM stands for prediction error minimization. Basic idea of the method is to select suitable function that depends on model parameters and is a measure of prediction error. Then, this function value is minimized by optimizing model parameters.

Let model prediction error be

$$e(t,\theta) = y(t) - \hat{y}(t,\theta), \quad (3.11)$$

where $y(t)$ is process measured output for known inputs and $\hat{y}(t)$ is model output for the same inputs.

Output error can be filtered first to remove possible noise from the data:

$$e_F(t,\theta) = Le(t,\theta), \quad (3.12)$$

where L is some stable filter.

The next step would be to use following norm:

$$V(\theta) = \frac{1}{N}\sum_{t=1}^{N} l(e_F(t,\theta)), \quad (3.13)$$

where N is number of samples, $l(\cdot)$ is scalar-valued function. Common choice for $l(\cdot)$ is:

$$l(e) = e^2, \quad (3.14)$$

that is convenient for computation and further analysis.

$V(\theta)$ is a cost function that measures model quality. Model can be optimized by finding better θ parameters.

$$\theta_N = \arg\min V(\theta). \quad (3.15)$$

Various methods exist for minimizing cost function value. These are least mean squares (LMS), Newton's method and its variations, Levenberg-Marquardt method etc. [47].

Estimated parameters must be initialized prior to identification. Fortunately there is a straightforward way to define these parameters by performing initial model parameters estimation via the subspace method. After initial model is acquired, it can be refined by PEM method. Matlab

Model Identification Toolbox is used to identify process model for further use in model predictive control. By default Matlab identification toolbox uses just 20 iterations for refinement, which provides similar result than the one produced by subspace method. When number of iterations is increased to 200–500, estimation becomes much more accurate. Implementation of subspace method and PEM method to the representative data set is able to provide a coherent linear model.

3.2 Process overview

The thesis addresses a problem of control of a water boiler and similar heat energy production processes. Here we consider a water boiler that is a part of bigger combined heat plant (CHP). The main objective of the CHP is to produce heat power for the nearest cities.

The boiler was installed in 1978. This is an old model KVGM-100 unit producing 100 kcal/h (116.3 MW) of heat power [52]. Some major investments were made to renovate the boiler infrastructure several years ago, therefore it is now equipped with modern measurement and control devices as well as DCS. All the control applications are implemented in the DCS on the software level. It is possible to create or modify applications without interrupting the process using set of predefined or programmable function blocks. Still, control methods used in boiler control at present day are quite conservative and are based on PI control algorithm.

Controlled variable in the main control loop of the boiler is the output temperature. The fuel burning process, transfer of heat power from furnace to water and hot water flow to the boiler output is relatively slow, so there is a time delay between gas flow on the furnace input and water temperature measurement on the boiler output. Since PI controller uses output error for control, then it is dependent on the delay. Therefore the integration time is set longer than measurement delay to avoid permanent overshoot. At the moment of process data collection integration time parameter was 240 seconds. As the ideal PI control algorithm is used $c_{out} = K_p(e + \frac{1}{T_i}\int e\,\mathrm{d}t)$ then the real integration time is almost 270 seconds ($K_p = 0.9$). These parameters result in a slow reaction of the controller to set point changes and disturbances.

As the first step of our research we decided to model the process for prediction purposes to reduce the delay effect. Neural network was used to identify the process model [65]. The obtained model provided an acceptable result for 5 minutes incremental prediction with 1 minute step, however it was not suitable for further research in simulation mode. The problem was an accumulated error that after the 10 steps of simulation resulted in oscillating behavior with increasing amplitude.

As the next step we decided to use linear identification of the process. Using the identified model we proceeded with the design of the model predictive control, which is known to be used as efficient solution for many control tasks in industry.

3.3 Model identification

Process layout is shown in Figure 3.1. Only devices related to the considered control loop are shown.

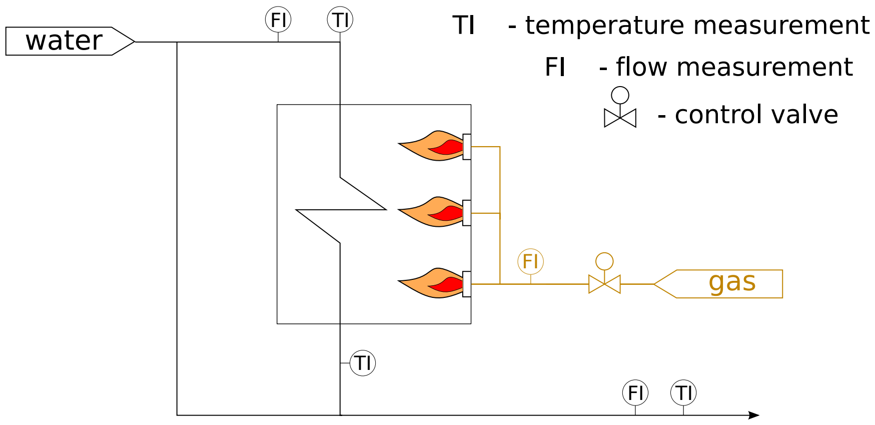


Figure 3.1: Process diagram.

After passing through the power unit (not shown in the layout), part of the water flow with measured temperature is delivered to the water boiler, where it is heated with natural gas. Remaining water flows through a bypass line. After exiting the boiler both flows are mixed and resulting flow and temperature are measured on the plant output. Inputs and outputs listed in Table 3.1 are used for deriving the dynamic model of the process.

Table 3.1: Model inputs and outputs.

Input/output	**Description**
Input 1	Gas flow
Input 2	Boiler water flow
Input 3	Plant water flow
Input 4	Inlet water temperature
Output 1	Boiler outlet temperature
Output 2	Plant outlet temperature

For testing purposes, two different models of the process at different operating points are created. One model is for process simulation and the other one for MPC design. This is with the purpose of having a simulation closer to reality, where the process model normally is not able to reflect exactly the real process behavior. If we succeed in such a challenging simulation, then it is highly probable that MPC will also work for the real process.

We propose to identify a model in state-space form (3.1). Use of this model proved later to be suitable for MPC design for such kind of processes maintaining excellent balance between complexity and control quality. Using the model equations given in Eq. (3.1), we propose to identify two models of the process at different operating points. Upper and lower limits of the process data used for identification are listed in Table 3.2. This table shows some overlapping in the operation points.

Table 3.2: Models operating points comparison.

Input/output	**Process**		**MPC**	
	Min	**Max**	**Min**	**Max**
Gas flow, Nm^3/h	3000	11500	2650	11450
Boiler water flow, m^3/h	1220	1330	1295	1370
Plant water flow, m^3/h	1500	4000	1600	4000
Inlet water temperature, **°C**	58	75	55	76
Boiler outlet temperature, **°C**	80	128	95	135
Plant outlet temperature, **°C**	74	86	76	88

Models were identified with the Matlab System Identification Toolbox [39] using state-space models by means of Predictive Error Minimization method (PEM).

Here, we distinguish two cases in the modeling.

3.3.1 Process model for simulation

Process model was identified using data sets collected from the process in the period from February 27, 2014 to March 6, 2014, 7634 samples. Data sampling interval is 1 minute which is enough to obtain a reliable representation of such a slow process. Resulting model is:

$$A = \begin{bmatrix} 0.3589 & -0.07221 & -0.1482 & 0.05587 \\ 0.2856 & 0.9659 & -0.1358 & -0.2506 \\ -0.7532 & 0.1398 & 0.4404 & -0.08773 \\ -0.03067 & 0.2475 & -0.207 & 0.7836 \end{bmatrix} \tag{3.16}$$

$$B = 10^{-5} \cdot \begin{bmatrix} -1.266 & -0.9273 & 3.994 & -728.6 \\ 2.514 & -2.471 & 1.859 & 676.6 \\ 1.853 & 3.958 & 7.8 & 47.22 \\ 1.797 & 14.56 & 0.7359 & 714.9 \end{bmatrix} \tag{3.17}$$

$$C = \begin{bmatrix} -64.75 & -18.78 & 46.79 & -31.42 \\ -29.62 & -37.28 & 13.03 & -5.59 \end{bmatrix} \tag{3.18}$$

$$D = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.19}$$

3.3.2 Process model for MPC

After analysis of existing data sets it was decided to remove input “Boiler water flow” from the model identification for MPC as its variations in data set are not sufficient to identify its impact on process output. Water flow value does not change most of the time thus it is impossible to identify its dynamics adequately. Figure 3.2 shows typical water flow measurements, where it is seen that data is noisy, but it does not change significantly for many days. Its value is set manually by the operators as the set point for the PI controller that keeps it on the defined level.

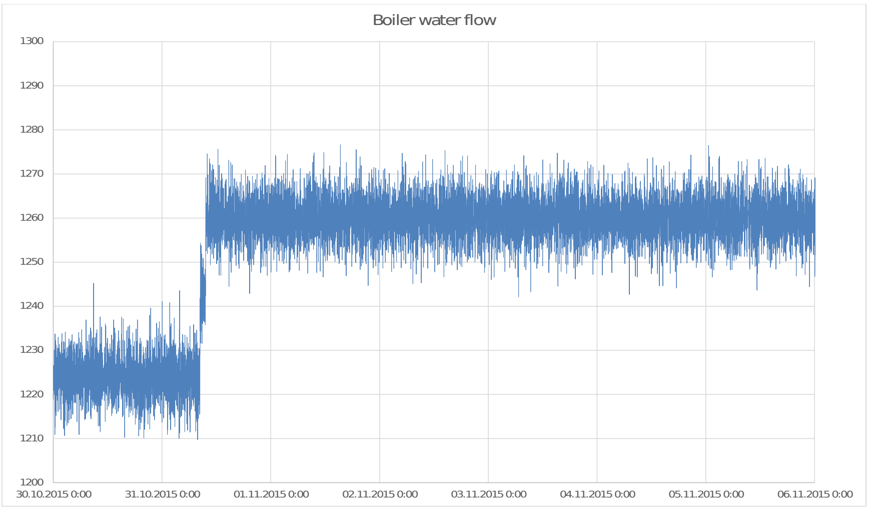


Figure 3.2: Typical example of water flow variations in the boiler.

At the same time output “Boiler outlet temperature” depends on water flow, so it was also removed from the identification. Final decision was to identify multiple input single output (MISO) process model.

Process model was identified using data sets collected from the process in the period from October 10, 2015 to November 6, 2015, 6199 samples. 500 iterations are used for PEM method. Data sampling interval is 1 minute which is enough to obtain a reliable representation of such a slow process. Resulting model is:

$$A = \begin{bmatrix} 0.5496 & -0.1142 & 0.1092 & 0.2782 & 0.3772 \\ -0.3363 & 0.7316 & -0.136 & 0.1998 & 0.2838 \\ 0.263 & 0.3913 & 0.6847 & 0.1086 & -0.395 \\ -0.2257 & -0.4801 & 0.4038 & 0.7623 & 0.09956 \\ -0.2506 & -0.2 & 0.4444 & -0.08627 & 0.4433 \end{bmatrix} \tag{3.20}$$

$$B = 10^{-5} \cdot \begin{bmatrix} -4.466 & 1.966 & -1264 \\ -2.6 & -1.646 & -664.4 \\ 1.104 & -1.927 & -307.5 \\ -7.316 & 1.379 & 917.4 \\ -3.167 & 1.794 & 415.2 \end{bmatrix} \tag{3.21}$$

$$C = \begin{bmatrix} -38.37 & 43.62 & -30.97 & -46.53 & 25.82 \end{bmatrix} \tag{3.22}$$

$$D = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.23}$$

This model will be used for the design of an MPC described in the next section.

3.4 Conclusion

Good process model is essential for model based control design. While making a decision about the type of model to be used for model based control it is very important to find a balance between complexity and efficiency. Use of nonlinear model causes the whole range of complicated problems that are existing in the stage of theoretical investigation at present days. We tried also to make neural network nonlinear model of the process [65] that behaved very similar to the real process in the operating area in which it was identified, but it was not able to provide feasible prediction outside the area, so its use required a separate long-term study. As the purpose of this work is to design a controller suitable for the whole range of real existing processes while simultaneously avoiding the complications arising from nonlinear modeling, it was decided to try linear model first. The resulting model showed good performance in the control of a nonlinear process of heat production. These processes are slightly nonlinear, so their behavior does not change significantly while traversing from one operating area to another. That is why using a linear model with a state observer is enough to predict short term process behavior in the whole operating range.

Chapter 4

Algorithms for Model Predictive Control

4.1 Overview

In MPC we aim to minimize a cost function defined as [42]

$$V(k) = ||Z(k) - T(k)||^2_{\mathbf{Q}} + ||\triangle U(k)||^2_{\mathbf{R}}, \tag{4.1}$$

where $Z(k)$ is the outputs prediction vector within prediction horizon H_p, $T(k)$ is set points trajectory within H_p and $\triangle U(k)$ is vector of process input moves (changes) within control horizon H_c. $Z(k)$ is calculated as:

$$Z(k) = \mathbf{\Phi} X(k), \tag{4.2}$$

where

$$\mathbf{\Phi} = \begin{bmatrix} \mathbf{C} & 0 & \cdots 0 \\ 0 & \mathbf{C} & \cdots 0 \\ \vdots & \vdots & \ddots \vdots \\ 0 & 0 & \cdots \mathbf{C} \end{bmatrix}.$$

The predicted states can be written as

$$X(k) = \hat{\mathbf{\Psi}} x(k) + \hat{\mathbf{\Upsilon}} u(k-1) + \hat{\mathbf{\Theta}} \triangle U(k), \tag{4.3}$$

where

$$X(k) = \begin{bmatrix} x(k+1|k) \\ x(k+2|k) \\ \vdots \\ x(k+H_p|k) \end{bmatrix}, \quad \triangle U(k) = \begin{bmatrix} \triangle u(k|k) \\ \triangle u(k+1|k) \\ \vdots \\ \triangle u(k+H_u-1|k) \end{bmatrix},$$

$$\hat{\mathbf{\Psi}} = \begin{bmatrix} \mathbf{A} \\ \mathbf{A}^2 \\ \vdots \\ \mathbf{A}^{H_p} \end{bmatrix}, \quad \hat{\mathbf{\Upsilon}} = \begin{bmatrix} \mathbf{B} \\ \mathbf{AB} \\ \vdots \\ \sum_{i=0}^{H_p-1} \mathbf{A}^i\mathbf{B} \end{bmatrix},$$

$$\hat{\mathbf{\Theta}} = \begin{bmatrix} \mathbf{B} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{H_u-1} \mathbf{A}^i\mathbf{B} & \cdots & \mathbf{B} \\ \sum_{i=0}^{H_u} \mathbf{A}^i\mathbf{B} & \cdots & \mathbf{AB}+\mathbf{B} \\ \vdots & \vdots & \vdots \\ \sum_{i=0}^{H_p-1} \mathbf{A}^i\mathbf{B} & \cdots & \sum_{i=0}^{H_p-H_u} \mathbf{A}^i\mathbf{B} \end{bmatrix}.$$

Matrix $\hat{\mathbf{\Psi}}$ is used to calculate the effect of current states to the states in the future within the prediction horizon H_p, $\hat{\mathbf{\Upsilon}}$ is the matrix used to calculate the effect of the latest process inputs to the future states and $\hat{\mathbf{\Theta}}$ is the matrix used to calculate the effect of future input changes to future states. H_u is control horizon (number of control actions into the future), $u(k+i|k) = 0$ for each $i > H_u$ with $H_u \leq H_p$. Thus:

$$Z = \mathbf{\Psi}x(k) + \mathbf{\Upsilon}u(k-1) + \mathbf{\Theta}\Delta U(k), \tag{4.4}$$

where $\mathbf{\Psi} = \mathbf{\Phi}\hat{\mathbf{\Psi}}$, $\mathbf{\Upsilon} = \mathbf{\Phi}\hat{\mathbf{\Upsilon}}$ and $\mathbf{\Theta} = \mathbf{\Phi}\hat{\mathbf{\Theta}}$.

Tracking error between free response and tracking trajectory in this case can be written as

$$E(k) = T(k) - \mathbf{\Psi}x(k) - \mathbf{\Upsilon}u(k-1), \tag{4.5}$$

where free response means the prediction of model outputs for the whole prediction horizon, if $\triangle U$ is always zero vector—no input moves assumed.

After this the cost function can be expressed:

$$V(k) = ||\mathbf{\Theta}\Delta U(k) - E(k)||^2_{\mathbf{Q}} + ||\triangle U(k)||^2_{\mathbf{R}}. \tag{4.6}$$

This can be brought to the form (refer to [42] for details):

$$V(k) = \mathit{const} - \triangle U(k)^T G + \triangle U(k)^T \mathbf{H} \triangle U(k), \tag{4.7}$$

where $G = 2\mathbf{\Theta}^T\mathbf{Q}E(k)$ and $\mathbf{H} = \mathbf{\Theta}^T\mathbf{Q}\mathbf{\Theta} + \mathbf{R}$.

To find the optimal $\triangle U(k)$, we set the gradient of $V(k)$ equal to zero, thus obtaining

$$\nabla_{\triangle U(k)} V = -G + 2\mathbf{H}\triangle U(k) = 0. \tag{4.8}$$

Then the optimal future input changes are given by

$$\triangle U(k)_{opt} = \frac{1}{2}\mathbf{H}^{-1}G. \tag{4.9}$$

Once the optimal process inputs are computed, we only take the first element, apply the control action, and send it to the DCS. Once this step is finalized, we apply the same concept in a receding horizon strategy.

Matrix inverse should not be computed directly in (4.9), as $\mathbf{H}$ can be ill-conditioned due to minor difference between process values in consequent prediction steps [42]. Matrix pseudo-inversion could be utilized in this case, but the task can become too computationally demanding because in case of long control horizon and many controlled inputs $\mathbf{H}$ has high dimension. E. g., singular value decomposition is computationally expensive and is not feasible in real-time applications.

At the same time solution of equation $\mathbf{A}x = y$ is very efficient with QR-decomposition of matrix $\mathbf{A}$ [23]. Matrix $\mathbf{A}$ is decomposed onto orthogonal matrix $\mathbf{Q}$ and triangular matrix $\mathbf{R}$. For orthogonal matrix $\mathbf{Q}^{-1} = \mathbf{Q}^T$. So, the equation can be modified to the form:

$$\mathbf{R}x = \mathbf{Q}^T y, \tag{4.10}$$

which is easily solvable triangular system of linear equations.

For QR-decomposition a Housholder algorithm can be used, which solves the problem in a very efficient way [23].

4.2 Augmented model for MPC

As we have different process model for simulation and for the MPC design, it is not possible to guarantee zero tracking error. If we do not measure the real output and do not restore model states respectively, then an offset appears between MPC model output estimation and the real process outputs. When the model output is close to the set point, real process output values can differ significantly. If the model is not accurate, MPC finds a steady-state point where it estimates that no updates to the control law are necessary and the process output should reach the set point without any additional actions. However, this is not the case, and the process does not converge to the set point because it has different behavior than the one predicted by the model of MPC. As a result, a static control error appears.

The following approach can be used to solve this problem. We augment the model with extra states that will compensate the offset in each cycle of process states and outputs estimation [68]. This augmented model has the following form:

$$\hat{\mathbf{A}} = \begin{pmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{C} & \mathbf{0} \end{pmatrix}, \hat{\mathbf{B}} = \begin{pmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0} & -\mathbf{I} \end{pmatrix}, \hat{\mathbf{C}} = \begin{pmatrix} \mathbf{C} & -\mathbf{I} \end{pmatrix}, \tag{4.11}$$

where $\mathbf{I}$ is identity matrix and $\mathbf{0}$ is zero matrix of suitable dimension.

Extra states and extra inputs are added to the model. Extra input vector U_{ext} consists of measured process outputs. Extra states are calculated as follows:

$$X_{ext} = \mathbf{C}X - U_{ext}, \tag{4.12}$$

where X is a vector of original states, X_{ext} is a vector of offsets between model outputs and real measured process outputs. It is calculated every execution cycle and subtracted from the original model output vector using $-\mathbf{I}$ in matrix $\hat{\mathbf{C}}$.

Unfortunately, the same model cannot be used to calculate the optimal process input due to problems with scaling. Coefficients of augmented states (matrix $\mathbf{C}$) and inputs (identity matrix) are significantly higher than coefficients of original states (matrix $\mathbf{A}$) and inputs (matrix $\mathbf{B}$). While making prediction MPC multiplies states with matrix $\hat{\mathbf{A}}$ and inputs with matrices $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ many times. This leads to domination of augmented states in prediction calculation thus masking the effect of original states and inputs. To avoid this, we use another approach. For prediction, we augment the model in a different way:

$$\tilde{\mathbf{A}} = \begin{pmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}, \tilde{\mathbf{B}} = \begin{pmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}, \tilde{\mathbf{C}} = \begin{pmatrix} \mathbf{C} & -\mathbf{I} \end{pmatrix}. \tag{4.13}$$

In this case offset is fixed for all prediction steps and any effect of the original input change will be correctly transferred to the model output. This model is much better for prediction than the original one, because it does not suffer from output offset, maintaining the original model behavior.

4.3 Kalman filter as state observer

As it was tested later, model augmentation works well, when process model is accurate enough and model prediction error is within 5%. When the model dynamics are identified well enough to predict output move direction, but prediction error is around 20%, then static part of augmented model starts to influence the MPC output too much reducing accuracy of control. For such kind of modes, a state observer is needed that restores model states in such way, so that model could produce output prediction with minimum error.

As we deal with the real system, where all measurements are subject to white noise, then it is natural to use Kalman filter as state observer.

Using the model 3.1 to predict process output we would need to correct model states before each prediction is made. The following mechanisms are used for correction and prediction [33, 42, 56]:

$$\hat{x}(k|k) = \hat{x}(k|k-1) + \mathbf{K}(k)\left[y(k) - \hat{y}(k)\right], \tag{4.14}$$

$$\hat{x}(k+1|k) = \mathbf{A}\hat{x}(k|k) + \mathbf{B}u(k), \tag{4.15}$$

where $\mathbf{K}(k)$ is Kalman gain and $\hat{y} = \mathbf{C}\hat{x}(k|k-1)$. Kalman gain is computed according to:

$$\mathbf{K}(k) = \mathbf{P}(k|k-1)\mathbf{C}^T \left[\mathbf{C}\mathbf{P}(k|k-1)\mathbf{C}^T + \mathbf{V}\right]^{-1}, \tag{4.16}$$

where

$$\mathbf{P}(k|k) = \left[\mathbf{I} - \mathbf{K}(k)\mathbf{C}\right]\mathbf{P}(k|k-1), \tag{4.17}$$

$$\mathbf{P}(k+1|k) = \mathbf{A}\mathbf{P}(k|k)\mathbf{A}^T + \mathbf{W}, \tag{4.18}$$

and $\mathbf{W}$ is input noise covariance matrix and $\mathbf{V}$ is output noise covariance matrix.

4.4 Constrained MPC formulation

This section is based on [11, 42, 72].

For example, valves can only operate in the range from 0% (fully closed) to 100% (fully open), pumps are obviously limited by the rated or achievable capacity and RPM specification. If produced value is beyond constraints, then actuator will travel to the limit and stop due to physical reasons, i.e., cause a saturation effect. In some cases this control action is not favorable. In Figure 4.1, an optimal value of unconstrained controller u_u is above maximum possible value, but real optimal value considering constraints u_c could be different than maximum limit of manipulated variable.

In case of MPC it is possible to set constraints on input moves $\triangle u$, input value u and output z:

$$\triangle u_{min} \leq \triangle u(k) \leq \triangle u_{max} \tag{4.19}$$

$$u_{min} \leq u(k) \leq u_{max} \tag{4.20}$$

$$z_{min} \leq z(k) \leq z_{max} \tag{4.21}$$

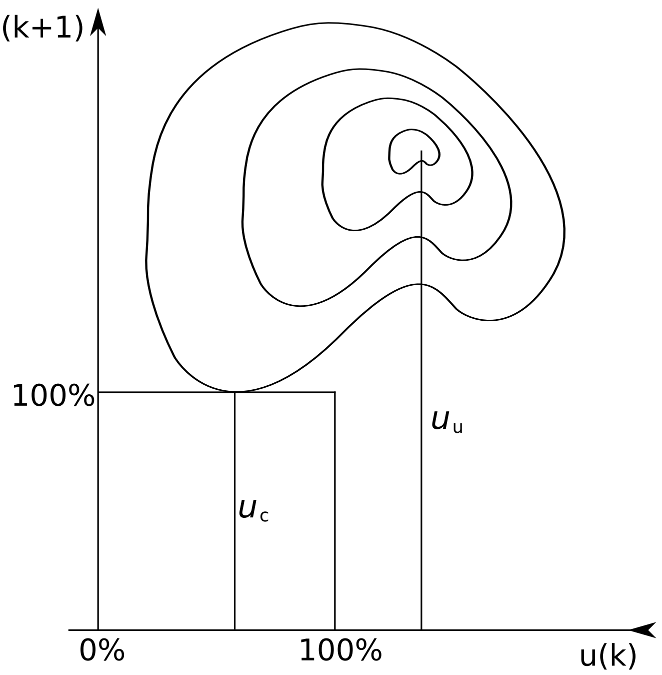


Figure 4.1: Constraints on manipulated variable and optimality

at any time moment, where x_{min} is minimum limit and x_{max} is maximum limit of respective variable.

As controller produces optimal input moves, then we are interested to express all Equations (4.19), (4.20), (4.21) through $\triangle u$. Let's define H_u-dimensional limit vectors $\triangle U_{min}$, $\triangle U_{max}$, U_{min}, U_{max}, Z_{min}, Z_{max}. The whole control horizon constraints can be expressed as:

$$\triangle U_{min} \leq \triangle U(k) \leq \triangle U_{max} \tag{4.22}$$

$$U_{min} \leq U_{(k-1)} + \mathbf{E}_I \triangle U(k) \leq U_{max} \tag{4.23}$$

$$Z_{min} \leq \mathbf{\Theta} \triangle U(k) + \mathbf{\Psi} x(k) + \mathbf{\Upsilon} u(k-1) \leq Z_{max} \tag{4.24}$$

where H_u-dimensional vector $U_{(k-1)}$ is

$$U_{(k-1)} = \begin{bmatrix} u(k-1) \\ u(k-1) \\ u(k-1) \\ \vdots \\ u(k-1) \end{bmatrix},$$

and $H_u \times H_u-$dimensional lower triangular matrix:

$$\mathbf{E}_I = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ 1 & 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Combined inequality has the form:

$$\begin{bmatrix} \mathbf{I} \\ -\mathbf{I} \\ \mathbf{E}_I \\ -\mathbf{E}_I \\ \mathbf{\Theta} \\ -\mathbf{\Theta} \end{bmatrix} \triangle U(k) \leq \begin{bmatrix} \triangle U_{max} \\ -\triangle U_{min} \\ U_{max} - U_{(k-1)} \\ -U_{min} + U_{(k-1)} \\ Z_{max} - \mathbf{\Psi} x(k) - \mathbf{\Upsilon} u(k-1) \\ -Z_{min} + \mathbf{\Psi} x(k) + \mathbf{\Upsilon} u(k-1) \end{bmatrix} \tag{4.25}$$

Finally taking into account Eq. (4.7) we get a problem in the quadratic programming form:

$$\min_{\theta} \tfrac{1}{2}\theta^T \mathbf{H} \theta + h\theta$$

subject to:

$$\mathbf{\Omega}\theta \leq \omega.$$

This type of problems has standard solutions methods. Most popular are active set and interior point. In this work interior point method was used as it was developed later then active set method and is known for the quality of requiring less computational power.

4.5 Interior point method

Interior point or barrier function method is deeply studied in [9,24,42,48,58, 74,75].

Barrier function method brings problem (4.7):

$$V(k) = \triangle U(k)^T \mathbf{H} \triangle U(k) - \triangle U(k)^T G \tag{4.26}$$

subject to constraints (4.25) to the form with no explicit constraints. Modified cost function is:

$$V(k) = \triangle U(k)^T \mathbf{H} \triangle U(k) - \triangle U(k)^T G + \mu B_f(\triangle U(k)), \tag{4.27}$$

where

$$B_f(\triangle U(k)) = -\sum_{i}^{H_u m} \left(\ln(\triangle U_{max_i} - \triangle U_i(k)) + \ln(\triangle U_i(k) - \triangle U_{min_i})\right).$$

B_f is a logarithmic barrier function. i refers to i_{th} element of vectors $\triangle U_{max}$, $\triangle U_{min}$ and $\triangle U(k)$. There are also other types of barrier functions, but the logarithmic function is one of the most convenient to use with standard optimization techniques. Function (4.27) is smooth, so it can be solved with various Newton's methods. As $\mu \to 0$, the solution of (4.27) tends to the solution of (4.26) [74].

With application of Newton's method update of $\triangle U$ results in

$$\triangle U = \triangle U - \alpha \mathcal{H}^{-1} g, \tag{4.28}$$

where g is gradient vector of the cost function, $\mathcal{H}$ is Hessian of the cost function and α is a scalar to ensure reduction of the cost function. Each element of the gradient vector take the form

$$\begin{aligned} g(i) = {} & \mathbf{H}_{\{i,:\}} \triangle U(k) + G(i) \\ & + \frac{\mu}{\triangle U_{max_i} - \triangle U_i(k)} - \frac{\mu}{\triangle U_i(k) - \triangle U_{min_i}}, \end{aligned} \tag{4.29}$$

where i refers to i_{th} element of gradient vector g and vector G, $\mathbf{H}_{\{i,:\}}$refers to i_{th} row of matrix $\mathbf{H}$. Hessian matrix $\mathcal{H}$ is

$$\mathcal{H} = \mathbf{H} + \mu \mathbf{D}, \tag{4.30}$$

where $\mathbf{D}$ is diagonal matrix with the following elements on the diagonal:

$$\mathbf{D}_{\{i,i\}} = \frac{1}{(\triangle U_{max_i} - \triangle U_i(k))^2} - \frac{1}{(\triangle U_i(k) - \triangle U_{min_i})^2}. \tag{4.31}$$

Gradient and Hessian expressions are only true, if $\triangle U(k)$ is feasible. That requires starting point for the optimization to be within limits, which is simple, if for example only input constraints are used:

$$\triangle u(k) = \frac{u_{max} + u_{min}}{2} - u(k-1), \tag{4.32}$$

$$\triangle u(k+i) = 0,\ i = 1, ..., H_u - 1. \tag{4.33}$$

In case of output constraints feasible starting point could be missing, if these constraints contradict each other. In these circumstances output constraints should be softened — moved enough for feasible point to appear. Output constraints are softened due to the reason that input constraints are normally related to physical limitations of the process making their softening impossible [42].

For interior point method implementation we need also scaling factor $\nu \in (0, 1)$ to prevent constraints violations, positive integer k to define number of iteration steps, barrier scaling factor $\mu > 0$ to define initial barrier size and weighting factor ζ to decrease μ in each iteration. The closer ν value is to 1, the close $\triangle u(k)$ goes to the limit. Scaling factor μ is needed prevent jump out of feasible region at the beginning of the algorithm execution. After assignment of initial value (e. g. 10) it decreases in each iteration of algorithm thus approaching problem (4.27) to the problem (4.26). Simple example of barrier function implementation is depicted on the picture 4.2.

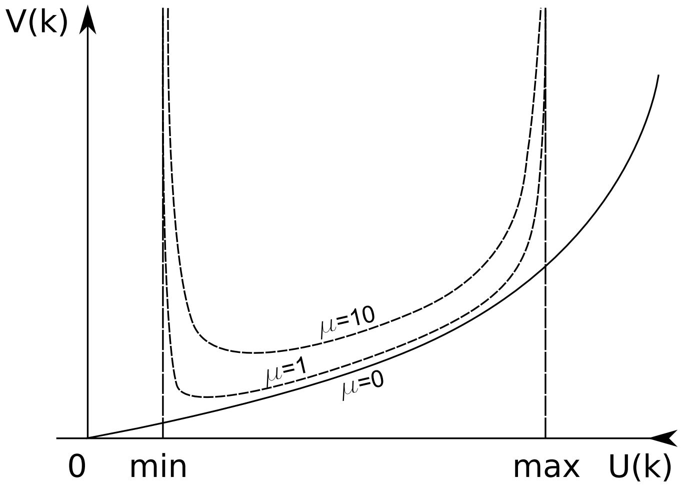


Figure 4.2: Limitation with barrier function.

The procedure of the interior point algorithm is as follows.

```
1. Initialize the control action via (4.32) and (4.33).
2. For $j = 1$ to $k$
3.        Compute gradient vector $g$ via (4.29).
4.        Compute Hessian matrix $\mathcal{H}$ via (4.30).
5.        Compute an estimation of $\triangle U$ update $\rho = \mathcal{H}^{-1}g$.
```

```
6.      For $i = 1$ to $H_u m$
7.          If $\rho(i) < \triangle U_i(k) - \triangle U_{max}$ then
8.              $\rho(i) = \nu\,(\triangle U_i(k) - \triangle U_{max})$.
9.          End If
10.         If $\rho(i) > \triangle U_{min} - \triangle U_i(k)$ then
11.             $\rho(i) = \nu\,(\triangle U_{min} - \triangle U_i(k))$.
12.         End If
13.     End For
14.     Optimize $\alpha$ value to get minimum value of the cost function after
    input move update.
15.     Update input move $\triangle U = \triangle U - \alpha\rho$.
16.     Update the barrier weight via $\mu = \zeta\mu$.
17. End For
```

Inversion of Hessian matrix $\mathcal{H}$ in step 5 is not needed [42] as ρ can be calculated with QR-decomposition with use of Householder algorithm in turn [23]:

$$\mathcal{H}\rho = g, \tag{4.34}$$

$$\mathbf{QR}\rho = g, \tag{4.35}$$

where Q is orthogonal ($Q^T = Q^{-1}$) and R is upper triangular. So, we have:

$$\mathbf{R}\rho = \mathbf{Q}^T g, \tag{4.36}$$

$$\mathbf{R}\rho = \hat{g}. \tag{4.37}$$

After this it is straightforward to compute ρ from:

$$\begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1n} \\ 0 & R_{22} & \cdots & R_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & R_{nn} \end{bmatrix} \times \begin{bmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_n \end{bmatrix} = \begin{bmatrix} \hat{g}_1 \\ \hat{g}_2 \\ \vdots \\ \hat{g}_n \end{bmatrix}. \tag{4.38}$$

After steps 7-12 we have ensured that ρ value does not violate the constraints. It is possible to find the most optimal value of α by taking $\alpha-$gradient from the unconstrained cost function and equate it to zero. From (4.26) we have:

$$V(k) = \triangle\left(U(k) - \alpha\rho\right)^T \mathbf{H}\left(U(k) - \alpha\rho\right) - \left(U(k) - \alpha\rho\right)^T G, \tag{4.39}$$

$$\frac{dV(k)}{d\alpha} = 2\alpha\rho^T \mathbf{H}\rho - 2\rho^T \mathbf{H}\triangle U + \rho^T G = 0, \tag{4.40}$$

$$\alpha = \frac{\rho^T G - 2\rho^T \mathbf{H}\triangle U}{2\rho^T \mathbf{H}\rho}. \tag{4.41}$$

To prevent constraint violation we have to limit α to the range $[0, 1]$.

If unconstrained MPC optimal solution is located beyond constrains, then in each iteration of interior-point method we decrease value of μ and approach value of constraint, but do not cross it.

4.6 Conclusion

In this chapter constrained MPC algorithm was discussed, where Interior-Point method is used to solve the quadratic programming problem. MPC algorithm is fully described for unconstrained and constrained cases. It is shown that it is possible to start search for solution from guaranteed feasible point and remain in feasible region.

The main contribution of this chapter is deep analysis of various sources on MPC theory and practice to develop an algorithm for real implementation suitable for our needs. The algorithm structure was taken from [74], but it was modified using [10, 11, 42, 47, 48] to get a simple, but efficient implementation.

In the following chapter the implementation of this algorithm in Java programming language will be described.

Chapter 5

Development of the MPC Software Application and the Simulation Environment

5.1 MPC application

5.1.1 Overview

MPC history started with a practical implementation as a commercial solution. Today there is a choice of commercially available MPCs. Several of the better known makes were listed in Chapter 1. The problem with the implementation of such a solution to small-scale processes in a small country is high cost. At the same time, benefits of model based control are not obvious for plant owners before implementation. This prevents MPC use even in these cases, where achieved benefits could justify possible costs. At the same time there are also some freeware tools available that provide MPC functionality. GenOpt based MPC tool [12] and GRAMPC tool [31] can be cited as examples of such tools. Related publications describe tools' design principles with test implementation to exemplary processes simulated in MATLAB. An attempt to implement these tools to real-life DCS and process would require significant efforts for building interfaces between MPC and the control system and configuring interconnections. As we are targeting lightweight, simple, but efficient implementation, a decision was made to design own MPC tool suitable for needs of current thesis context [68].

5.1.2 Application structure

Designed MPC is implemented in Java programming language as a command line application. As a first stage, we have chosen to implement the unconstrained case. The constrained case requires further programming in Java, and the implementation of an interior-point method, see e. g. [54], which makes the Java programming more complex. After unconstrained solution was implemented and tested, application was developed further to include constraints.

Java is not the most efficient programming language [43, 71], but it is convenient, reliable and supports cross-platform execution and therefore has become a standard programming language for developing enterprise applications. Cross-platform compatibility can be useful in the future, providing more freedom for a real process control implementation. Java program is compiled to the platform independent byte code that is executed in platform dependent Java Virtual Machine (JVM). This type of execution requires compilation from byte code to machine code in real-time. Each application method is compiled on the fly when it is called by the program. This can cause processing overhead for real-time compilation. Java uses Jast-In-Time (JIT) compilation to avoid it. JIT compiles program methods once during program execution when they are called and stores them in memory for later use. As our program is cyclical, processing overhead for online compilation should be minimal.

We need an MPC application that supports the following functions:

- read configuration from the file,
- communicate with the process via Modbus/TCP,
- calculate optimal inputs for the process,
- interact with the user via command line interface (CLI).

As Java is multi-threading programming language by its nature, it is possible to build these functions as separate threads. Application structure is shown in Figure 5.1. The main thread that starts all other functions is created first. It starts separated threads for the functions listed in Table 5.1.

Table 5.1: MPC java application threads.

Thread	**Function**
1	Command line scanner to read user's commands
2	Modbus/TCP interface to communicate with DCS
3	MPC to calculate optimal process inputs

MPC thread reads process model and MPC parameters from an XML configuration file.

Modbus thread reads its configuration parameters from another XML file. Modbus implementation is based on Jamod library [27].

The CLI shows model, MPC parameters and Modbus parameters loading status after the application has started up. By default it does not provide any other information, but it accepts a variety of commands from the user to force the application to write useful information to CLI, such as calculated optimal process inputs, prediction vector, process input moves, etc.

When the application is running, it reads process values from the DCS via Modbus interface, calculates optimal process input and writes it back to the DCS via Modbus.

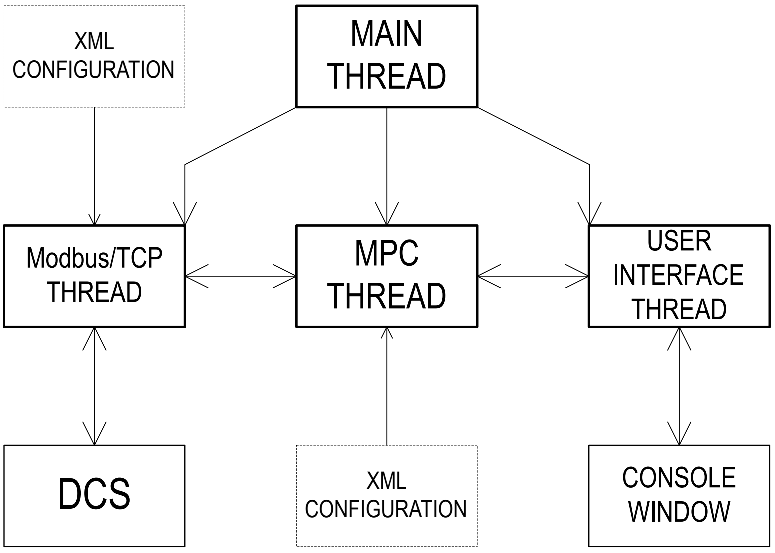


Figure 5.1: MPC application structure

5.1.3 Application highlights

A real-time multi-thread application is created that performs all required functions to manage optimal process control:

1. Communication with DCS;
2. MPC algorithm execution with predefined period;
3. Interaction with the user.

Due to use of popular Modbus/TCP protocol it is possible to apply MPC control to almost any control system, where advanced control techniques are not embedded. As application is written in Java it is executable on the OS platforms commonly used in industry: Windows or Linux. Addition of MPC control to any existing system is simple and straightforward.

5.2 Simulation environment

5.2.1 DCS overview

In this work Valmet DNA DCS was used for simulation environment. This DCS is widely used in industry world wide. It is has been in steady development since 1979 utilizing modern IT-technologies at all times. It has full scope of functionality of modern control systems such as process control on lower level, supervisory control and data acquisition system (SCADA), high level control and many others. It also supports execution of Java functional blocks, but in this case this option was used only for process modeling, but not for MPC application to keep its platform independence.

5.2.2 Simulation environment design

After process model has been obtained, a simulation environment is developed to transfer the results to a real-life application. Simulation is built with Java function block of DCS software. A simple Java code block is created. It reads matrices A, B and C of the state-space model as well as process inputs values and calculates model states and outputs using (3.1).

Without loss of generality, we initialize the model with $x(0) = 0$. Model is run in DCS run-time environment in a 4-second cycle. Simulation is carried out 15 times faster than real process intentionally to make testing faster without need to wait for long process transients.

MPC application is developed in Java using Eclipse development environment in Debian Linux. It uses TCP/IP network to communicate with DCS and can be located in the same computer with the process simulation or in any other accessible through the network. DCS process simulation runs under Windows OS. After a functional version of MPC application was released, it was used on the same computer with DCS process simulation.

To make simulation environment ready for testing we need to only prepare communication with the MPC application using Modbus/TCP interface that is supported by DCS. Related configuration was prepared with DCS role in communication as slave and MPC application as master.

After MPC application has started, it shows to the user that all the parameters (Modbus/TCP settings, model, MPC settings) have been loaded. As the applications acts as Modbus master, it sends requests to the configured slave. In case of no response, the application quits informing the problem to the user. If slave (DCS process simulation) is also configured correctly and is on-line, then communication is established.

It is also important to secure the control system from communication problems or any MPC application functionality issue. For these purposes *keepalives* are used, binary signals that switch their values after a predefined period of time. One signal is generated by the Modbus thread every 2 seconds, another one by the MPC thread in each execution cycle. These signals are also sent through Modbus interface to DCS. The later has a timeout 5 seconds for Modbus thread signal and 10 seconds for the MPC thread signal. Time counter of each signal is reset when related signal changes its value. If timeout period is exceeded in any counter, DCS diagnoses MPC application fault, freezes manipulated input value and gives an alarm to the operator.

MPC application reads all relevant process variables listed in Table 3.1 as well as output set points. All of these variables are used to calculate current states of the model (including offset compensation in current step) and optimal process inputs for the next cycle.

In this case only *Input 1* is controllable. All other process inputs are measurable disturbances. MPC application calculates the optimal value for the gas flow only and sends it to DCS via Modbus interface.

DCS graphical user interface is shown in Figure 5.2. Basic process layout is depicted on the display. All inputs and outputs are placed on the positions where they are measured in the process. Process inputs that are not controlled—water flow through the boiler, water flow through the plant and boiler inlet temperature—can be modified by the user directly simulating process disturbance (in reality these are modified by other parts of the process). Process outputs—boiler outlet temperature and plant outlet temperature—are calculated by the simulation model in DCS. These are shown at the bottom of the corresponding value boxes. Topmost parts are relative set points that can be modified by the user.

Trends for the outputs and for the controllable input are added to facilitate the understanding of the results. These are updated in real time.

Process simulation can be operated in three modes. *Mode 1* is manual. Operator can modify process inputs (including gas flow) and observe resulting outputs values. For *mode 2,* PI controller is used in DCS. Related data box is right under the gas pipe on the display. PI controller is driven by the plant output temperature error signal and controls gas flow to keep temperature close to set point. All other inputs can be modified by the user. The controller is made only for simulation purposes. This loop does not exist in the real process control system. PI controller was tuned using direct synthesis method [6]: $k_p = 1.24$, $t_i = 10$. In *mode 3*, gas flow value is calculated by the MPC application.

In the current work, only the plant output temperature is of interest. There is no need to control boiler output because we assume that this output is maintained within the normal range of operation (between 80 to 140°C). In the real world, the boiler is operated in conditions where its output is in the range of 85 to 125°C. Because we control a slow process, it is not possible that the output goes outside of the normal operating condition without the operator seeing it. If this happens, then the operator can take corrective

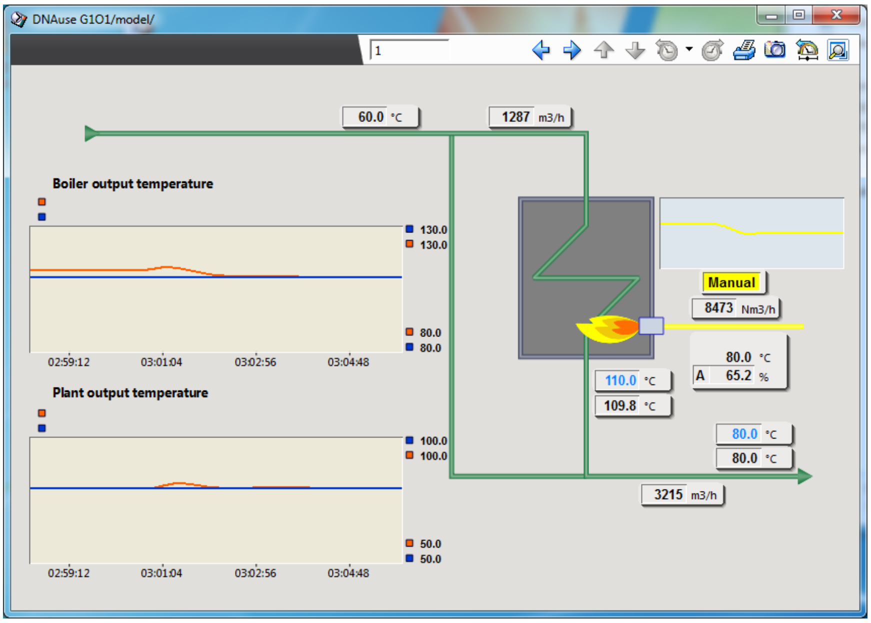


Figure 5.2: DCS user interface.

actions. Of course, MPC including constraints would be ideal in order to take these limitations into account. Nonetheless, unconstrained MPC can also operate in these circumstances. We keep boiler output temperature in the model for later use in the future continuation of this work.

To tune the MPC strategy, we consider that the plant temperature output is the main point of interest. Weights for *Output 1* (boiler output temperature) and for *Output 2* (plant output temperature) are set to 0.1 and 350, respectively. These values are found by trial and error.

We also need to penalize input changes to prevent overreaction in the gas flow that makes the whole system unstable. We find that a suitable weight for this case is 0.1.

Testing of MPC in the simulation framework will be performed by giving an extreme disturbance that cannot happen in the real world. By doing this, we want to assess whether the MPC application is able to stabilize the process in such extreme conditions. In case of success, there is a high chance of having a stable control in the real process. The disturbance is chosen as a instant decrease in the boiler inlet temperature by $2°C$. In real operations, this value only changes around $\pm 0.2°C/min$.

Simulation results are shown in Figure 5.3. Also DCS PI-controller result is added to the trend. We can see that PI-controller behavior is similar to MPC result, but MPC shows a quicker reaction.

The main purpose of this work is to assess whether MPC is capable of

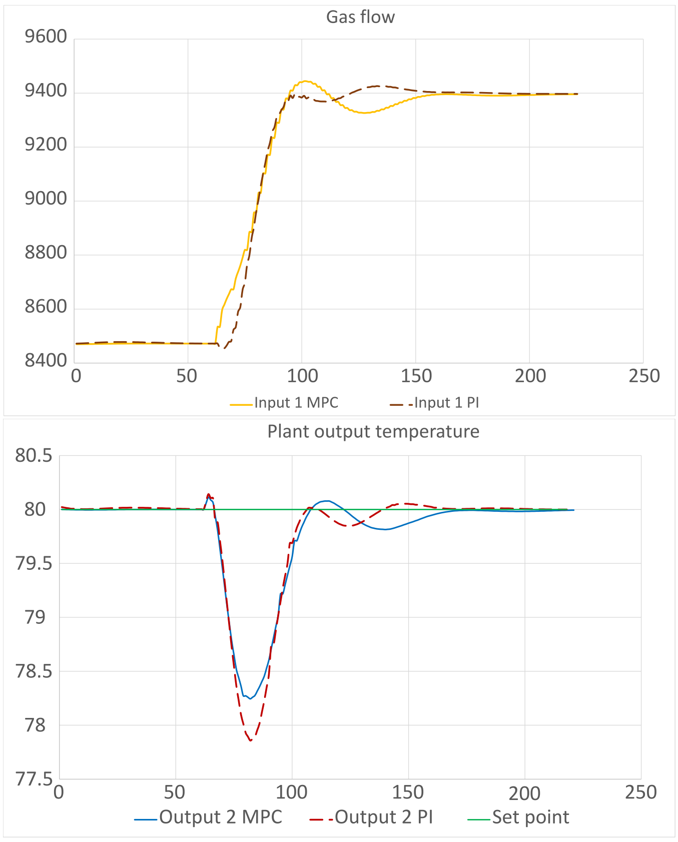


Figure 5.3: Simulation results

making the closed loop stable, and to overcome the possible difficulties in the implementation of the MPC strategy in the real world. We also performed other tests with other large-scale disturbances. In all those cases, we obtained a stable closed loop.

5.2.3 Simulation environment highlights

Simulation environment is made on the base of real DCS, that provides control system functionality, Modbus/TCP interconnection and also emulates process dynamics on the base of state-space model. Communication between MPC application and DCS was established using Modbus/TCP protocol via common Ethernet network. In real industrial process DCS is connected to the process through sensors and actuators, but communication between MPC application and DCS remains the same. So, in this simulation environment it is possible to test full functionality of MPC application before implementation to the real process. In the next chapter real process control implementation will be described.

5.3 Conclusion

Matlab MPC toolbox is a commonly available application to study MPC in practice. It provides good opportunities to test MPC algorithm functionality with all kind process simulations and also real processes (through Data Acquisition toolbox) in laboratory conditions. Licensing issues and powerful hardware demand prevent its easy implementation to real industrial process. This was the reason to apply significant efforts to develop light weight MPC application free of licenses that can be applied to almost any existing control system based on DCS of PLC as almost every industrial controller supports Modbus/TCP protocol nowadays. Testing in simulation environment based on real DCS software showed that resulting application is applicable for real process control. Results of real implementation will be described in the next chapter.

Still, the current application has many drawbacks that can prevent it from easy use in the future as it is not yet user friendly and requires a lot of manual work to prepare it for implementation. So, there are few obvious directions for further development:

- Develop web based user interface that provides initial MPC configuration and further maintenance in a user friendly manner locally or remotely over network connection;
- Implement model identification functionality inside application. At the moment model is identified externally in Matlab.
- Previous item will require connection of database to collect process data for identification.
- At the moment only one MPC thread is utilized in the application. In the future application could support many threads to implement many MPC controllers to one control system.
- Fuzzy controller module could be added to the application in the future to provide control for processes with unmodeled dynamics from the same software.
- Modbus/TCP is well supported interface, but almost obsolete nowadays. Very soon implementation of more modern interfaces will be required. OPC UA interface support could be added to the software in the future.

Nevertheless all these desirable extensions do not prevent one from implementing existing application's MPC functionality to the real process already now. The next chapter will give detailed description of the MPC real life implementation.

Chapter 6

Application of MPC to the Industrial Process

6.1 MPC implementation phases

As was written earlier, typical way of MPC implementation to the real process nowadays includes [15]:

1. Pretest and preliminary MPC design.

2. Plant testing.

3. Model and controller development.

4. Commissioning and training.

In the pretest phase this process and its existing PI control were investigated. View Chapter 1 for details of existing PI controller.

Lower control loops of the process were tuned sufficiently well solving control tasks in a proper manner. Retuning of upper control loops of the cascaded PI controller to achieve better performance would require too many efforts and separate study. As it was decided to go on with MPC retuning of existing upper loops was skipped.

The second phase of MPC implementation requires plant testing to collect process data for model identification. As plant DCS includes information system with a database for process data collection, then all necessary data was acquired with these tools. Plant process was running with different loads for many weeks. Collected data with sample time of 1 minute was used for offline model identification.

After process model was identified and MPC designed and tested first in Matlab/Simulink software [66] and then in simulation environment as described in the previous chapter [68], it was commissioned to the process as the final phase of the implementation. Further, the application of both unconstrained and constrained variants of MPC control is verified with the real process in the loop. Finally, the obtained results are discussed.

6.2 Application of unconstrained MPC to the real process

The power plant process control system has a process network, where controllers, operator stations, engineering and information servers are combined into one communication segment. As engineering server is used only in control modifications, it does not have permanent computation load, so it is natural to use it as the running environment for the MPC application. The server meets the hardware requirements to run the developed MPC Java application.

One of DCS controllers was configured to be a Modbus slave (server), while MPC application was a Modbus master. After proper configuration of both ends, communication between MPC and controller was established. The same communication applications with minor modifications were loaded to the controller as were used in simulation environment.

Existing gas flow controller of the boiler was reconfigured to use one more source of set point — computed by MPC. Selection button was added to operator displays, so it became possible to select between MPC and old PI controller.

The first version of MPC application didn't include handling of constrains. It was first implemented to the process with use of augmented model (See Section 4.2) in the spring of 2017. As it was the very end of heating season there was no time to test application thoroughly. It worked in process control only a few hours, but during this time it showed satisfactory control performance (Figure 6.1).

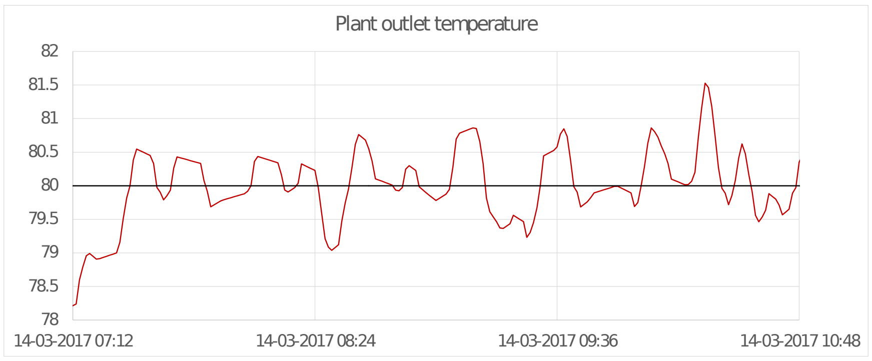


Figure 6.1: Process control with augmented model.

The model described in Subsection 3.3.1 having four inputs and two outputs was used to design MPC that time. Basic model prediction error was in the range $2 - 4°C$, so augmented states could fully compensate it making augmented model suitable for MPC design without use of state observer.

Before the start of the next heating season MPC application was extended with constrains handling functionality. Also, the model was replaced with three inputs one output structure. Use of augmentation was tested with this model as well, but the model error peaked at about $20°C$. In these circumstances MPC performance was not satisfactory providing control error up to $2°C$. Model augmentation was replaced with Kalman observer for the future test.

Model augmentation is easier to use than an observer, so it could provide certain benefits in MPC implementation, but it requires a little bit more study and testing to confirm its usability.

6.3 Application of constrained MPC to the real process

In January 2018, a new version of MPC was started with the parameters set in simulation environment. It took two hours to fine tune these for better performance in real conditions. Suitable tuning parameters were decided as follows:

$$H_p = 12, \tag{6.1}$$

$$H_c = 3, \tag{6.2}$$

$$Q = \mathrm{diag}(400, 400, 400, 400, 400, 500, 500, 500, 500, 600, 600, 600), \tag{6.3}$$

$$R = \mathrm{diag}(0.01, 0.01, 0.01), \tag{6.4}$$

where $\mathrm{diag}(x)$ is diagonal matrix with parameters x on the main diagonal.

There was only one input constraint that could be useful for process operators—gas minimum flow should not be below 3700 m^3/h. If gas flow goes lower, then boiler automation will turn off one of the burners due to low load. Later, a large number of preparations by the operator is required to turn the burner on again, so operators prefer to limit minimum load, even if the plant output will be higher than set point for a short period of time.

Operators were instructed to follow controller performance and switch it off, if its behavior is deemed inappropriate, i.e., leading to poor performance (deviations from the set point more than $2°C$ and inability of the controller to stabilize the output within short period of time, e.g., one hour). The controller was left to run in automatic mode for some days in different process loads.

After evaluation of newly collected data, it became clear that the linear model does not work satisfactory in a long prediction horizon, as prediction mistakes are too significant and result in the degradation of control performance. Also, tuning parameters of matrix Q are too rapid making process output fluctuate on lower loads.

New parameters were set as follows:

$$H_p = 8, \tag{6.5}$$

$$H_c = 2, \tag{6.6}$$

$$Q = \text{diag}(200, 200, 300, 300, 400, 500, 800, 800), \tag{6.7}$$

$$R = \text{diag}(0.01, 0.01). \tag{6.8}$$

With new parameters MPC performance was satisfactory. Deviations from the set point were in the range $1°C$. Only significant disturbances (such as plant flow increase for $400\,m^3/h$ within less than an hour time, see Figure 6.2 for details) caused controlled variable deviate more than $1°C$.

6.4 Implementation results

It is possible to highlight following results of MPC implementation to control of the district heat plant:

- New controller provides satisfactory control on the whole range of boiler operation starting from lowest loads 15–20 MW and to the full load of 116 MW. This is a significant improvement compared to existing control that is usable only starting from 40 MW load. On lower loads manual operation was used.
- On comparable loads new controller has twice better performance than existing controller. For 7000 data samples sum of square errors is almost twice higher for PI controller: 1534.63 against 3624.35.
- MPC handles disturbances better as it can predict disturbance effect on output and starts to act earlier than PI controller to mitigate consequences (as illustrated in Figure 6.3).

- Overall MPC controller is suitable for such kind of power production processes where multivariable model can be identified with all measurable process values.

Figure 6.3 shows comparison of MPC (green line) and PI (red line) controlled process on comparable power production. It is seen that in approximately the same operating conditions MPC controlled process output has smaller output deviation from the set point. Also disturbance (plant flow significant change) is handled better in MPC case.

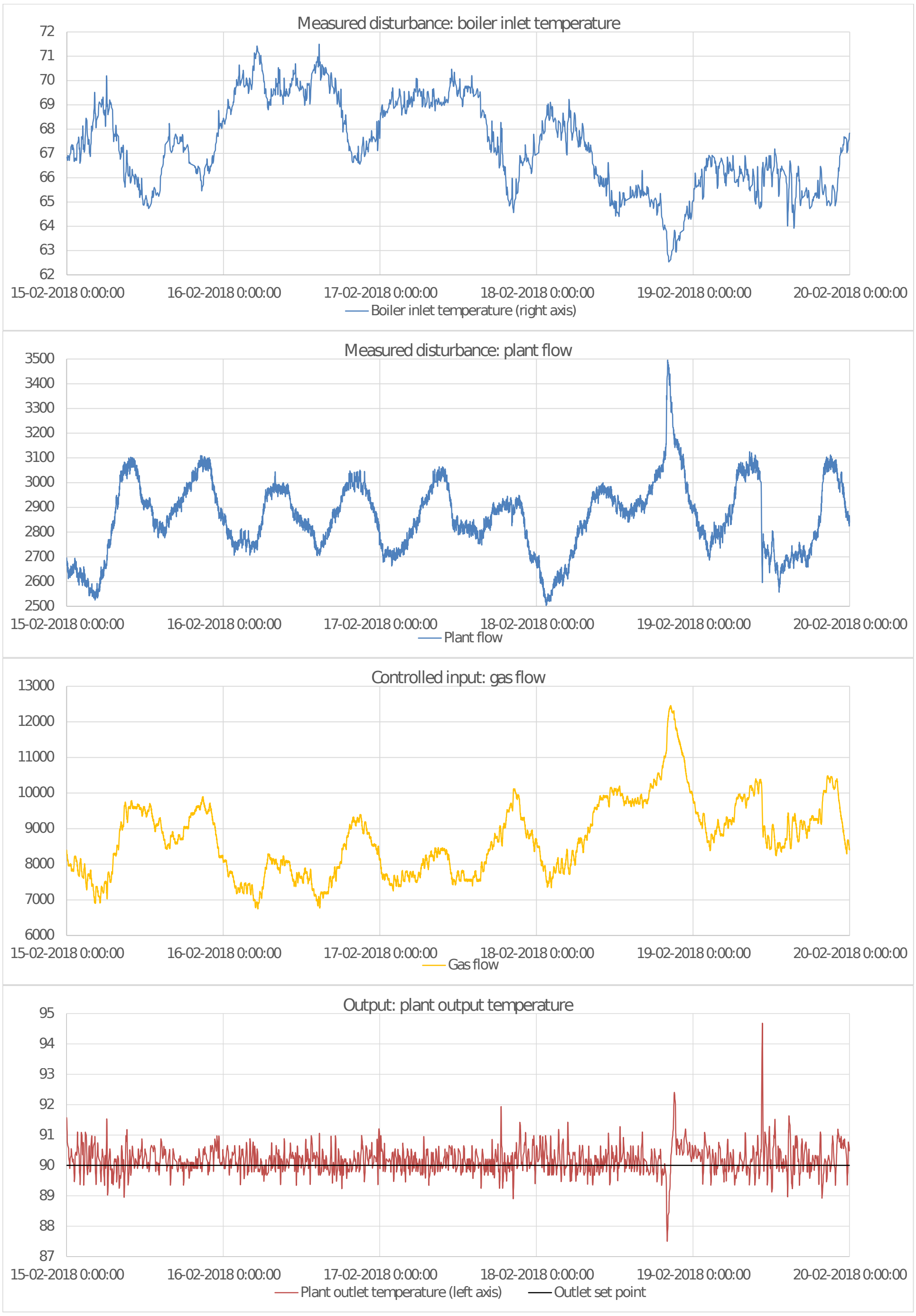


Figure 6.2: MPC performance trends.

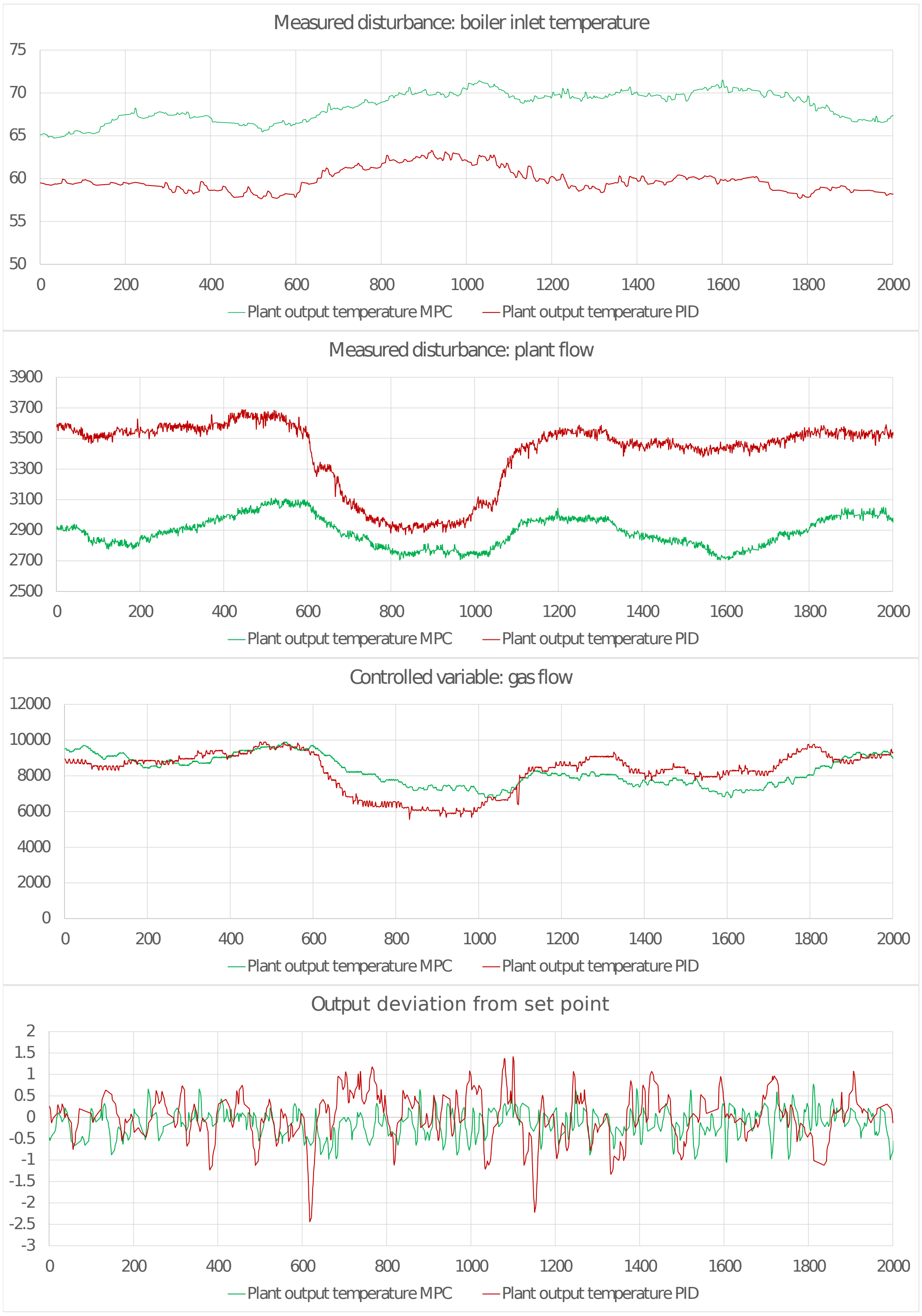


Figure 6.3: MPC and PID behavior comparison.

Conclusions

This work was inspired by observation of current state of industrial automation with dominating PID control law even in such loops, where its functionality is insufficient for solving control tasks. Nevertheless, PID controller can handle majority of control tasks in industry, there are loops with output delays and multiple inputs, where PID cannot provide suitable solution. Normal way to proceed in these cases is to build a more complicated cascade PID controller with additional logic and calculations. This construction is hard to tune. At the same time control performance of such controller is not sufficient. This kind of process and cascade controller were selected in real-life industry of Estonia to study possibility for use of more efficient control techniques.

Model based control is very suitable for such cases, as it is able to handle multiple input multiple output process control loops by its nature thus overcoming delay challenge by prediction of process output beyond delay horizon. There are also certain limitations for use of model based control in situation, when process cannot be accurately modeled. That is why two types of processes were considered in this study.

Initially, a process type with unmodeled dynamics was considered, where accurate modeling is not possible due to missing measurements or other process uncertainties prevent us to build a model capable of prediction of the process behavior with sufficient accuracy. Another advanced process control technique was implemented for this case—fuzzy control is an acceptable substitution for model based control in such cases. A real-life process—small biofuel boiler house—was considered as an example of the process with unmodeled dynamics, where fuzzy control was successfully implemented. Unfortunately, simple enough fuzzy control for smaller loops changes to complicated one as the number of controller inputs increases, because number of applicable rules grows in geometric progression. Therefore, it was decided to proceed with processes with modeled dynamics in this study.

Processes with modeled dynamics fit well into model based control. A real-life industrial process was selected for model based control application—a combined heat power plant in Tallinn, Estonia. It was proposed to control plant heat output with MPC instead of existing cascade PI controller that was not able to handle process disturbances due to slow control dynamics. There is a stationary part of the process—waste to energy unit that produces normally constant amount of electricity and heat. Plant output is adjusted to district heat network demand by invoking an additional water boiler working on natural gas fuel. Gas flow was chosen to be the controlled variable for MPC while other process inputs (plant water flow, boiler water flow and boiler inlet temperature) are seen as measured disturbances. Boiler outlet temperature and plant outlet temperature were selected to be controlled process outputs. Process was thoroughly studied before modeling. During the study, several attempts were made to identify and test a MIMO model. Unfortunately, boiler water flow was not manipulated before (it was constant most of the time), so collected data did not include suitable data sets to identify effect of this variable on process outputs. It is clear from process physics that the boiler water flow has only effect on boiler output temperature, so it was decided to exclude these variables from the identification of the model. Final model had 3 inputs and 1 output. Prediction error minimization method was used to identify industrial process model. Model predictive controller was built on the base of the model. It was first tested in simulation environment specifically created in real control system software to make tests as close to reality as possible.

After successful tests in simulation, MPC was applied to real process control and it showed much better results than existing PI controller provided. Squared deviation from the set point was decreased more than twice with use of MPC. There were chosen few data sets of 7000 samples from the process data before implementation and after implementation. In all comparisons there was approximately the same result—squared error of MPC on the level 1500 versus squared error of PI controller on the level 3500. Another benefit of MPC implementation was observed in the fact that controller's reaction to disturbances was faster due to prediction function of the new solution. Since the plant's heat output is hot water with certain temperature, its set point is kept in the middle of operation specification to guarantee this temperature to be in specification even under disturbance effect. With more precise and reliable control it is possible to move set point down thus reducing use of natural gas simultaneously with producing output of the same quality, which is essential in modern world of sustainable and environment friendly technologies.

Use of standard interface for the communication allows to use this MPC not only with certain control system, but with any existing PLC controlling any heat production process. This study showed in practice benefits of used model based techniques for process control that can be extended to any other similar process. Required efforts to do this include process model identification, configuration of the communication interface between MPC application and process control system and MPC parameters tuning.

Two typical cases of industrial heat production were considered in this thesis, where implementation of advanced process control brought certain benefits for production quality. There are hundreds of other processes of similar type, where the same of even bigger improvements could be achieved. At the same time process owners often have no idea about possible benefits. Typical practice in this field is to offer APC solution to process owners for free, while charging customer after performance improvement has been implemented, documented and started to bring the profit to the process owner. Development of optimization solutions has a bright future, because it decreases production costs, increases quality and competitiveness, and generally goes in line with trends of the new industrial revolution, defined in the Industry 4.0 specification.

In what follows, the contributions of this thesis are reviewed.

Contribution

- Analysis of existing heat energy production control loops in Estonian industry for benefits of model based control use. Comparison of processes with modeled and unmodeled dynamics was done. Two different types of processes were compared — biofuel boiler and gas boiler. These are the main types of district heat production facilities in Estonia generating up to 80% of the whole DH heat energy [62].

- Investigation and analysis of existing control systems' drawbacks influencing the quality and efficiency of energy production. PI controllers were replaced by advanced controllers to mitigate limitations of PI algorithm, such as inability to control MIMO loops and poor performance controlling loops with significant time delay. Advanced controllers showed better performance in both cases.

- Analysis of advanced control strategies and their applicability to different types of District Heating Plants in Estonia. Selection of suitable control technique depending on process characteristics. This work showed that control quality and thus production quality can be significantly improved with advanced process control techniques. Squared deviation from the set point was decreased more than twice with use of APC. Better control quality reduced operator involvement into process control thus increasing level of automation of the production facility.

- Mathematical modeling—identification of dynamic system models—and CACSD based simulation of the process and closed loop control system. This is a mandatory part of the model based control implementation.

- Analysis of advanced control strategies based on computer simulations. Successful simulation tests are required prerequisites for advanced control implementation to the real process.

- Practical implementation of designed controller, analysis of the results, and formulation of expert advice towards improving the quality of existing feedback control loops. In case of the processes with unmodeled dynamics, a fuzzy controller was successfully implemented to the real DH biofuel boiler to keep it in correct operating area all the time and keep process output close to the set point. For the process with modeled dynamics model was identified and MPC was designed and implemented to the real DH gas boiler to reduce deviations from the set point significantly — more than twice.

Future Research

Current research included identification and model based control of the process with just a few inputs and one output. It would be reasonable to continue study and extend model and control to more inputs and outputs. The process includes more loops that could be combined together in one multiple input multiple output controller that calculates optimized control actions for the whole plant.

Combination of fuzzy controller and MPC for the same process control could also be a useful topic for further study. This could produce a robust MPC, that is in modern trends of MPC research.

It would be also interesting to develop nonlinear MPC in the future and compare it with linear one to understand, if it provides better performance. If yes, then does it happen in all processes? Is implementation complexity compensated with improved result?

There are plenty of questions that arise when we talk about new technologies in new implementations. New questions will arise after previous ones will have been answered. This is an infinite way of development.

References

[1] K. J. Åström and T. Hägglund, "PID controllers: Theory," *Instrument Society of America, Research Triangle Park, NC*, pp. 284–287, 1995.

[2] K. J. Åström and T. Hägglund, "The future of PID control," *Control Engineering Practice*, vol. 9, no. 11, pp. 1163–1175, nov 2001.

[3] K. J. Åström and T. Hägglund, "Advanced PID control," in *The Instrumentation, Systems, and Automation Society*, 2006.

[4] K. J. Åström and P. Kumar, "Control: A perspective," *Automatica*, vol. 50, no. 1, pp. 3–43, jan 2014.

[5] M. Bakošová, J. Oravec, and K. Matejičková, "Model predictive control-based robust stabilization of a chemical reactor," *Chemical Papers*, vol. 67, no. 9, pp. 1146–1156, 2013.

[6] B. W. Bequette, *Process control: modeling, design, and simulation.* Prentice Hall Professional, 2003.

[7] T. L. Blevins, "PID advances in industrial control," *IFAC Proceedings Volumes*, vol. 45, no. 3, pp. 23–28, 2012.

[8] S. Bolognani, S. Bolognani, L. Peretti, and M. Zigliotto, "Design and implementation of model predictive control for electrical motor drives," *IEEE Transactions on industrial electronics*, vol. 56, no. 6, pp. 1925–1936, 2009.

[9] T. T. C. Roos and J.-P. Vial, *Theory and Algorithms for Linear Optimization.* Wiley, 1997.

[10] E. F. Camacho and C. Bordons, *Model predictive control in the process industry.* Springer Science & Business Media, 2012.

[11] E. Camacho and C. Bordons, *Model Predictive Control.* London: Springer-Verlag, 2004.

[12] B. Coffey, F. Haghighat, E. Morofsky, and E. Kutrowski, "A software framework for model predictive control with GenOpt," *Energy and Buildings*, vol. 42, no. 7, pp. 1084–1092, 2010.

[13] C. R. Cutler and B. L. Ramaker, "Dynamic matrix control - a computer control algorithm," in *Joint automatic control conference*, no. 17, 1980, p. 72.

[14] L. Dalhoumi, M. Chtourou, and M. Djemel, "On the fuzzy model predictive control of interconnected nonlinear systems," *Arabian Journal for Science and Engineering*, pp. 1–18, 2017.

[15] M. L. Darby and M. Nikolaou, "MPC: Current practice and challenges," *Control Engineering Practice*, vol. 20, no. 4, pp. 328–342, 2012.

[16] B. De Schutter, "Model predictive traffic control for green mobility," in *Control Conference (ECC), 2014 European*. IEEE, 2014, pp. 2260–2263.

[17] L. del Re, F. Allgöwer, L. Glielmo, C. Guardiola, and I. Kolmanovsky, Eds., *Automotive Model Predictive Control*. London: Springer-Verlag, 2010.

[18] M. Dotoli, A. Fay, M. Miśkowicz, and C. Seatzu, "Advanced control in factory automation: a survey," *International Journal of Production Research*, vol. 55, no. 5, pp. 1243–1259, 2017.

[19] J. Eaton and J. Rawlings, "Model predictive control of chemical processes," *Chemical Engineering Science*, vol. 47, pp. 705–720, 1992.

[20] M. G. Forbes, R. S. Patwardhan, H. Hamadah, and R. B. Gopaluni, "Model predictive control in industry: Challenges and opportunities," *IFAC-PapersOnLine*, vol. 48, no. 8, pp. 531–538, 2015.

[21] C. Garcia, D. Prett, and M. Morari, "Model predictive control: Theory and practice–a survey," *Automatica*, vol. 25, pp. 335–348, 1989.

[22] M. Gevers, "Identification for control: From the early achievements to the revival of experiment design," *European journal of control*, vol. 11, no. 4-5, pp. 335–352, 2005.

[23] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU Press, 2012, vol. 3.

[24] E. M. Hendrix, G. Boglárka *et al.*, *Introduction to nonlinear and global optimization*, 2010.

[25] D. Hrovat, S. Di Cairano, H. E. Tseng, and I. V. Kolmanovsky, “The development of model predictive control in automotive industry: A survey,” in *Control Applications (CCA), 2012 IEEE International Conference on.* IEEE, 2012, pp. 295–302.

[26] H. Huang, L. Chen, and E. Hu, “Model predictive control for energy-efficient buildings: An airport terminal building study,” in *Control & Automation (ICCA), 11th IEEE International Conference on.* IEEE, 2014, pp. 1025–1030.

[27] Jamod, “Java modbus library,” [Accessed: February, 2017]. [Online]. Available: http://jamod.sourceforge.net/. [Online]. Available: http://jamod.sourceforge.net/

[28] S.-L. Jämsä-Jounela, “Future trends in process automation,” *Annual Reviews in Control*, vol. 31, no. 2, pp. 211–220, 2007.

[29] M. Jansson, “Subspace identification and ARX modeling,” *IFAC Proceedings Volumes*, vol. 36, no. 16, pp. 1585–1590, 2003.

[30] R. E. Kalman *et al.*, “Contributions to the theory of optimal control,” *Bol. Soc. Mat. Mexicana*, vol. 5, no. 2, pp. 102–119, 1960.

[31] B. Kapernick and K. Graichen, “The gradient based nonlinear model predictive control software GRAMPC,” in *Control Conference (ECC), 2014 European.* IEEE, 2014, pp. 1170–1175.

[32] M. Klaučo and M. Kvasnica, “Control of a boiler-turbine unit using MPC-based reference governors,” *Applied Thermal Engineering*, vol. 110, pp. 1437–1447, 2017.

[33] V. Kordic, *Kalman Filter*, V. Kordic, Ed. Intech, 2010.

[34] A. Krishnan, B. V. Patil, P. Nataraj, J. Maciejowski, and K. Ling, “Model predictive control of a CSTR: A comparative study among linear and nonlinear model approaches,” in *Control Conference (ICC), 2017 Indian.* IEEE, 2017, pp. 182–187.

[35] M. Kvasnica, M. Herceg, L. Čirka, and M. Fikar, “Model predictive control of a CSTR: A hybrid modeling approach,” *Chemical papers*, vol. 64, no. 3, pp. 301–309, 2010.

[36] P.-O. Larsson, F. Casella, F. Magnusson, J. Andersson, M. Diehl, and J. Akesson, “A framework for nonlinear model-predictive control using object-oriented modeling with a case study in power plant start-up,” in *Computer Aided Control System Design (CACSD), 2013 IEEE Conference on.* IEEE, 2013, pp. 346–351.

[37] J. H. Lee, "Model predictive control: Review of the three decades of development," *International Journal of Control, Automation and Systems*, vol. 9, no. 3, pp. 415–424, 2011.

[38] B. G. Liptak, *Instrument Engineers' Handbook, Vol. 2: Process Control and Optimization, 4th Edition.* CRC Press, 2006.

[39] L. Ljung, *System Identification: Theory for the User*, 2nd ed. Prentice-Hall, 1999.

[40] L.-h. Luo, H. Liu, P. Li, and H. Wang, "Model predictive control for adaptive cruise control with multi-objectives: comfort, fuel-economy, safety and car-following," *Journal of Zhejiang University-Science A*, vol. 11, no. 3, pp. 191–201, 2010.

[41] J. Ma, S. J. Qin, and T. Salsbury, "Experimental study of economic model predictive control in building energy systems," in *American Control Conference (ACC), 2013.* IEEE, 2013, pp. 3753–3758.

[42] J. Maciejowski, *Predictive control: with constraints.* Prentice Hall, 2002.

[43] G. G. Magalhaes, A. L. Sartor, A. F. Lorenzon, P. O. A. Navaux, and A. C. S. Beck, "How programming languages and paradigms affect performance and energy in multithreaded applications," in *Proc. VI Brazilian Symp. Computing Systems Engineering (SBESC)*, Nov. 2016, pp. 71–78.

[44] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Systems*, vol. 36, no. 6, pp. 30–44, dec 2016.

[45] U.-C. Moon, Y. Lee, and K. Y. Lee, "Practical dynamic matrix control for thermal power plant coordinated control," *Control Engineering Practice*, vol. 71, pp. 154–163, 2018.

[46] K. Muske and J. Rawlings, "Model predictive control of with linear models," *AIChE Journal*, vol. 39, pp. 262–287, 1993.

[47] O. Nelles, *Nonlinear system identification: from classical approaches to neural networks and fuzzy models.* Springer Science & Business Media, 2013.

[48] Y. Nesterov and A. Nemirovskii, "Interior-point polynomial algorithms in convex programming," 1994.

[49] A. O'Dwyer, *Handbook of PI and PID controller tuning rules.* Imperial college press, 2009.

[50] K. M. Passino and S. Yurkovich, *Fuzzy Control.* Addison-Wesley, 1998, vol. 20.

[51] K. Patan and J. Korbicz, “Nonlinear model predictive control of a boiler unit: A fault tolerant control study,” *International Journal of Applied Mathematics and Computer Science*, vol. 22, no. 1, pp. 225–237, 2012.

[52] Production Association “SAEM”, “Boiler KVGM-100 data sheet,” [Accessed: May, 2018]. [Online]. Available: https://saem.su/kotel-kv-gm-100-150-kv-gm-116-3-150.

[53] S. J. Qin and T. A. Badgwell, “A survey of industrial model predictive control technology,” *Control engineering practice*, vol. 11, no. 7, pp. 733–764, 2003.

[54] C. V. Rao, S. J. Wright, and J. B. Rawlings, “Application of interior-point methods to model predictive control,” *Journal of optimization theory and applications*, vol. 99, no. 3, pp. 723–757, 1998.

[55] J. Richalet, A. Rault, J. Testud, and J. Papon, “Model predictive heuristic control: Applications to industrial processes,” *Automatica*, vol. 14, no. 5, pp. 413–428, 1978.

[56] P. Y. C. H. Robert Grover Brown, *Introduction to Random Signals and Applied Kalman Filtering with Matlab Exercises*, 4th ed. Wiley, 2012.

[57] J. Rodriguez, M. P. Kazmierkowski, J. R. Espinoza, P. Zanchetta, H. Abu-Rub, H. A. Young, and C. A. Rojas, “State of the art of finite control set model predictive control in power electronics,” *IEEE Transactions on Industrial Informatics*, vol. 9, no. 2, pp. 1003–1016, 2013.

[58] L. E. Scales, “Introduction to non-linear optimization,” 1985.

[59] P. Sistu and B. Bequette, “Nonlinear predictive control of uncertain processes: Application to a CSTR,” *AIChE Journal*, vol. 37, pp. 1711–1723, 1991.

[60] Statistics Estonia, “Boilers and generated heat by type of boiler,” 2017, [Accessed: May, 2018]. [Online]. Available: http://andmebaas.stat.ee/.

[61] M. Sugeno, *Industrial Applications of Fuzzy Control.* New York, NY, USA: Elsevier Science Inc., 1985.

[62] R. Vaks, “Current and future developments of district heating in Estonia,” Energy Department Ministry of Economic Affais and

Communications of Estonia, techreport, Mar. 2018. [Online]. Available: http://epha.ee/images/2018/Kaugkütte%20seminar%20Taani%20saatkonnas%2002.03.2018/Rein%20Vaks,%20DH%20Estonia_DK_02.03.2018.pdf

[63] P. Van Overschee and B. De Moor, *Subspace identification for linear systems: Theory–Implementation–Applications.* Kluwer Academic Publishers, 1996.

[64] P. Van Overschee and B. De Moor, “N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems,” *Automatica*, vol. 30, no. 1, pp. 75–93, 1994.

[65] V. Vansovits, E. Petlenkov, K. Vassiljeva, and A. Guljajev, “Identification of industrial water boiler for model predictive control of district heat plant,” in *Proc. 13th Biennial Baltic Electronics Conf*, 2012, pp. 315–318.

[66] V. Vansovits, E. Petlenkov, K. Vassiljeva, A. Tepljakov, and J. Belikov, “Application of MPC to industrial water boiler control system in district heat plant,” in *Proc. 13th Int. Conf. Control Automation Robotics Vision (ICARCV)*, 2014, pp. 1609–1614.

[67] V. Vansovits, A. Tepljakov, K. Vassiljeva, and E. Petlenkov, “Towards an intelligent control system for district heating plants: Design and implementation of a fuzzy logic based control loop,” in *Proc. IEEE 14th Int. Conf. Industrial Informatics (INDIN)*, 2016, pp. 405–410.

[68] V. Vansovits, B. I. Godoy, A. Tepljakov, K. Vassiljeva, and E. Petlenkov, “Model-based control design for a district heating plant,” in *Industrial Informatics (INDIN), 2017 IEEE 15th International Conference on.* IEEE, 2017, pp. 615–620.

[69] S. Vazquez, J. Leon, and L. Franquello, “Model predictive control: A review of its applications in power electronics,” *IEEE Industrial Electronics Magazine*, vol. 8, pp. 16–31, 2014.

[70] M. Viberg, “Subspace-based methods for the identification of linear time-invariant systems,” *Automatica*, vol. 31, no. 12, pp. 1835–1851, 1995.

[71] R. Vivanco and N. Pizzi, “Computational performance of Java and C++ in processing large biomedical datasets,” in *Proc. (Cat. No.02CH37373) IEEE CCECE2002 Canadian Conf. Electrical and Computer Engineering*, vol. 2, 2002, pp. 691–696 vol.2.

[72] L. Wang, *Model predictive control system design and implementation using MATLAB®.* Springer Science & Business Media, 2009.

[73] B. J. West, *Fractional calculus view of complexity: Tomorrow's science.* CRC Press, 2016.

[74] A. Wills, A. Mills, and B. Ninness, "FPGA implementation of an interior-point solution for linear model predictive control," vol. 44, pp. 14 527–14 532, 2011.

[75] S. J. Wright, "Primal-dual interior-point methods," 1997.

[76] H. Xiaoying, W. Jingcheng, Z. Langwen, and W. Bohui, "Data-driven modelling and fuzzy multiple-model predictive control of oxygen content in coal-fired power plant," *Transactions of the Institute of Measurement and Control*, vol. 39, no. 11, pp. 1631–1642, 2017.

Acknowledgments

The author wishes to express his deep gratitude to his supervisors Dr. Eduard Petlenkov and Dr. Kristina Vassiljeva. And special thanks to Dr. Aleksei Tepljakov for valuable comments during thesis preparation and for the professional editing of the thesis text.

Abstract

Advanced Control of District Heating Processes in Estonia

The current thesis considers the problem of implementation of model based control to real industrial processes. It covers theoretical preliminaries for building model predictive controller, its development as a stand-alone software and successful implementation to heating process in the combined heat power plant located in Tallinn, Estonia. Before considering model based control an unmodeled advanced process control was considered to show that MPC is not able to control processes with unmodeled dynamics, but there are also other techniques that can handle such cases. Fuzzy controller implementation to the boiler house located in Rapla town, Estonia was considered as an example of such solution.

Overall the thesis covers all stages of MPC implementation starting with process study, MPC design, testing in simulation environment (specially designed for our case in real DCS software), implementation to the real process and further analysis of results. Implementation was successful showing significant improvement in process control comparing to previously used solution based on cascade PI controller. MPC controller was used and tested in the power plant during heating season of years 2017/2018. Process owner is satisfied with achieved results and is going to use the controller in the future as well.

Kokkuvõte

Keskkütte Soojuse Tootmisprotsesside Juhtimine

Väitekirjas uuritakse matemaatilisel mudelil põhineva regulaatori (MPC) kasutamist soojusenergia tootmise protsesside juhtimisel Eestis. Töö käsitleb MPC teooriat, regulaatori realiseerimist eraldi tarkvarapaketina ja edukat rakendamist Tallinnas töötavas soojuselektrijaamas. Töös on uuritud ka mittemodelleeritava protsessi juhtimist aruka regulaatoriga. Seda tüüpi lahenduste näidisena onhägusregulaator rakendatud katla koormuse juhtimiseks Rapla katlamajas.

Doktoritöö demonstreerib kõiki MPC rakendamise samme alates protsessi uurimisest, läbi MPC arendamise ja testimise simulatsioonikeskkonnas (spetsiaalselt selleks loodud reaalse juhtimissüsteemi tarkvara alusel), kuni regulaatori rakendamiseni tegelikus tootmisprotsessis ning lõpetades tulemuste põhjaliku analüüsiga. MPC asendas edukalt reguleerimisahelas seni teatud puudustega töötanud PI regulaatorite kaskaadi. Autori poolt väljatöötatud lahendus on kasutuses elektrijaamas alates 2017/2018 hooajast ningjaama meeskond on tulemustega väga rahul. Regulaatori kasutamine, arendamine ja sellega seotud uuringud jätkuvad ka tulevikus.

Appendix 1

Publication 1

Reference

V. Vansovits, E. Petlenkov, K. Vassiljeva, and A. Guljajev, “Identification of industrial water boiler for model predictive control of district heat plant,” in Proc. 13th Biennial Baltic Electronics Conf, Oct. 2012, pp. 315–318.

Abstract

Widely used PID controller has number of limitations that do not allow using it effectively to solve complicated control issues. The framework of the solution is presented in the paper. A nonlinear model of a district heat plant boiler is identified by training an artificial neural network. The model is used to predict the behavior of real plant.

Identification of Industrial Water Boiler for Model Predictive Control of District Heat Plant

V. Vansovits[1], E. Petlenkov[2], K. Vassiljeva[2], A. Guljajev[3]

[1]Metso Automation Inc., Pikk St. 9, 10123 Tallinn, Estonia, E-mail: vitali.vansovits@metso.com
[2]Department of Computer Control, TUT, Ehitajate tee 5, 19086 Tallinn, Estonia, E-mail: {eduard.petlenkov;kristina.vassiljeva}@dcc.ttu.ee
[3]Eesti Energia AS, Laki St. 24, 12915 Tallinn, Estonia, E-mail: aleksandr.guljajev@energia.ee

ABSTRACT: Widely used PID controller has number of limitations that do not allow using it effectively to solve complicated control issues. The framework of the solution is presented in the paper. A nonlinear model of a district heat plant boiler is identified by training an artificial neural network. The model is used to predict the behavior of real plant.

1 Introduction

Almost every automation supplier is ready to apply advanced control technique nowadays. However, PID is the most popular and widely spread control method [1]. There are several reasons for that. First of all PID is well known and relatively easy, it has just few parameters to tune and every control engineer knows its principle [2]. On the other hand, advanced control techniques require special knowledge. Moreover, advanced knowledge is needed for its maintenance. This makes advanced control implementation extremely expensive comparing to traditional PID.

PID is suitable for controlling simple linear Single Input Single Output (SISO) loops, but it is quite weak for more complicated processes with several inputs and several outputs. Control quality of PID significantly depends on the chosen P, I and D parameters and moreover can vary on different loads. There are also advanced control tasks where it is not applicable at all. Such kind of task could be selection of the most efficient way to solve control issue.

In this paper a district heat plant controlled by a Distributed Control System (DCS) is concerned. Only PI-controllers are used to overcome all control challenges. Perspective goal for the control system is to be able to select optimal combination of heat producing units and control it with the least possible fuel consumption and best efficiency. There is no possibility to achieve this target with existing control methods.

Current goal is to control district heat water temperature on the plant output. The general structure of the plant includes three similar water boilers. Boilers are connected to pipeline in parallel. There is also a bypass line. Boilers use natural gas as fuel. Each boiler has three burners. It is possible to control gas flow on the input of the whole boiler and air flow on the input of each burner (see figure 1).

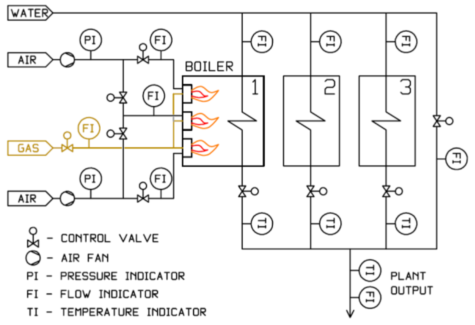


Figure 1. Boiler scheme

Boiler can work in base mode (constant load) and control mode (variable load). In control mode a cascade controller [3], [5] with three PI (Proportional Integral) cascades is used (see figure 2).

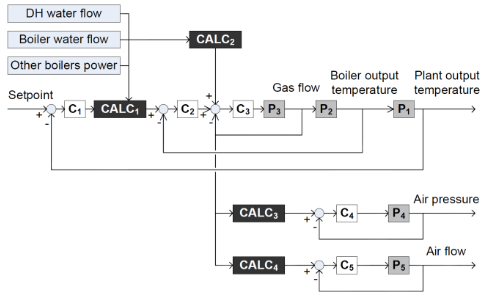


Figure2. Control structure

Master controller consists of feedforward coefficient calculation (CALC$_2$) and PI controller (C$_1$) in series with setpoint calculation (CALC$_1$). Master controller controls plant output temperature by giving setpoint to the second cascade. Second cascade controls temperature on the boiler output with PI-controller (C_2). It uses setpoint and feedforward coefficient acquired from the master and calculates setpoint for the third cascade. This in turn controls gas flow to the boiler with another PI controller (C_3) by giving commands directly to the gas flow valve actuator. From gas flow measurement air pressure and air flow setpoints are calculated and transferred to the related PI controllers (C_4 and C_5).

Tuning of cascade controller requires some extra efforts comparing to tuning of a simple PID-controller [5]. Integration times of the controller cascades are selected to be quite big (up to 240 seconds on the 2nd cascade) to provide stability. At the same time it causes control action to be too slow. As a result control of boiler output temperature (C_2) is not fast enough – temperature setpoint changes faster than system achieves equilibrium (see figure 3).

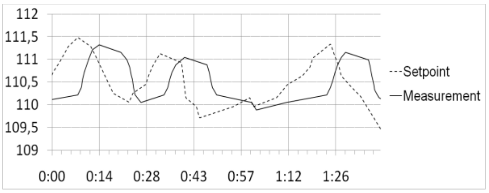


Figure 3. Boiler output setpoint and measurement

As the boilers are controlled with PI-controllers there is no any knowledge about the process in concern. There is no any algorithm applied to the system that could select the most efficient equipment to fulfill district heat pipeline requirements. Only human can perform this task at the moment.

These are the reasons that lead us to the necessity of having some other control strategy that allows accumulating knowledge about controlled object and creating its model. The model can be further utilized for control purposes.

2 Proposed concept

It is obvious that good knowledge of a controlled process is needed to be able to evaluate its efficiency and predict its behavior for control purposes.

There are three similar boilers, but each of them is unique because of different age, different time in operations and some minor mechanical differences. That is the reason why boilers are the most efficient on different loads and that is why it is important to select proper boiler for certain district heat system requirements.

In this case object identification and modeling is vital, because it gives a method to compare boilers and select the most suitable of them for certain load.

At the same time knowing the model it is possible to predict object behavior over a specified time horizon [4], [6] and perform more efficient control actions without delays which are inevitable in existing PI-controller application.

Online identification can be used to identify the model of the considered process. An intelligent controller that is able to follow object parameters, identify it and create model. After that it is possible to start model based control continuing to follow object parameters and correcting model when it is needed.

Controller needs to have variable number of inputs (in certain limits) to make its maintenance easier. In this case it is not mandatory for control engineer to know controller from inside. Number of inputs can be just increased or reduced to follow changes in real process under control. New model of the process will be identified automatically after some time.

At the same time it should be ensured that controller does not go out of equilibrium and interrupt the controlled process. This could happen if object's properties change due to some reasons. A special comparator is needed that compares setpoints and measurements and switches to conventional control strategy, if intelligent controller is not able to provide correct control actions. Conventional control based on PI-algorithm is often not the most efficient, but reliable enough.

Intelligent controller, switched out of the control, continues to monitor the process and correct its model. When it is able to predict object behavior again with suitable precision it can return to control. For maintenance purposes controller can also be manually switched out of the control.

This kind of intelligent controller is not restricted by a fixed model and can be reused without consumption of additional labor. It can be applied over existing conventional PI-controller to increase quality of control.

As the key point of intelligent controller is correct plant model we concentrate in this paper on plant identification.

3 Modeling for Predictive Control

One of the possibilities to overcome the above mentioned problems is to apply a nonlinear Model Predictive Controller (MPC) [3], [6], [7]. This controller contains a model of the controlled plant which is used to predict future behavior of the plant in order to take proactive control actions. It allows us to overcome some delays in control of a complex plant consisting of several subsystems connected between each other. The quality of the prediction directly depends on the accuracy of the identified model. Our goal is to obtain the highest possible accuracy of the plant's model and consequently the accuracy of the prediction made by this model. The behavior of each node of the complex system will be modeled separately.

In this paper we present identification of one of the nodes: one of three boilers used to heat the water flow. The process takes three measurements as its inputs: flow of air, gas and water (measured in m^3/s). Temperature of the output water flow can be considered as the output of the heating process. As all measurements are related only to the considered boiler, we don't need to take into account other boilers. They can be in or out of operation, but they do not affect any of current boiler parameters. Thus we have a Multiply Input – Single Output (MISO) dynamic system with 3 inputs and one output.

The identification was made on real plant data. We measured input and output values during 48 hours. We took one measurement per minute. Thus we got a data set consisting of 2880 measurements of each parameter. 2000 of them were selected for model identification and 880 were left for validation of the obtained model.

The process can be identified by both linear and nonlinear model. PID controller used at the moment is based on linear description of the process. *Mean squared error* (*MSE*) of the identified linear model on the validation set was $79 \cdot 10^{-3}$. On the same validation set MSE of the obtained nonlinear model was $61 \cdot 10^{-3}$ that clearly shows the effectiveness of implementing a nonlinear model and demonstrates of nonlinearities in the process. This comparatively small difference (about 23%) may lead to a significant loss in accuracy of recursive prediction where model error will accumulate on each iteration.

We identified a nonlinear model of the dynamic process by training a fully connected feedforward artificial neural network (two-layer perceptron) with external feedback. Neural network implements the following difference equation

$$\hat{y}(t) = f(z(t-1), z(t-2), z(t-3), z(t-4))$$

representing 4th order nonlinear model of the plant. Here $z(t) = [u_1(t)u_2(t)u_3(t)\ y(t)]$ is a vector of inputs and outputs at the corresponding time instance. The structure of the NN-based model is depicted in figure 4, where $u(t)$ is a three element vector of inputs (air, gas and water).

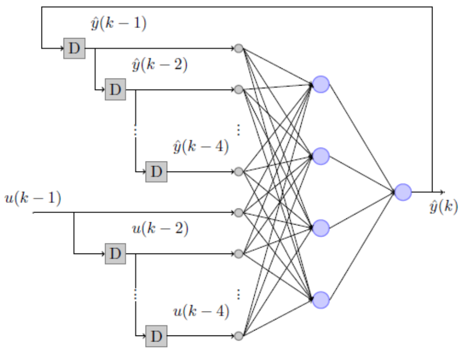


Figure 4. Neural Network based model

This model was successfully applied to recursive prediction of changes in temperature of the water flow. Mean error of 5 minutes ahead predictions 0,1 °C. Absolute values of 5 minutes ahead prediction error are presented in figure 5.

Levenberg-Marquardt (LM) algorithm was chosen to perform the training, since it is much more efficient compared to other techniques when the network contains no more than a few hundred weights. Also the training speed of LM algorithm is much higher and the feed-forward neural network trained with it can better model nonlinearity.

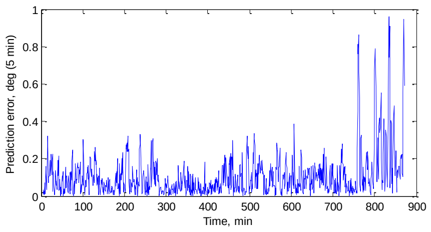


Figure 5. Five minutes prediction error

It can also be seen from the figure that maximal prediction error is less than 1 °C. Only 4 neurons with nonlinear hyperbolic tangent activation function were used to identify this model. It means that the obtained NN-based nonlinear model is comparatively simple, it can easily be trained and adapted, easily implemented in software and will work very fast (prediction can be calculated in some milliseconds).

4 Conclusions

The main problem in distributed control system of the district heat plant is stated in the paper. At the moment, each node of the plant is controlled by a separate PI controller. After being interconnected into a large distributed system, delays between different nodes occur and lead to inaccuracy and late control actions. Control of each node depends on the result of the control action taken on the previous node earlier. Experimental results have shown that average delay is about 5 minutes.

In this paper we consider identification of one part of a complex system presented in figure 6. Identification and MPC control of all interconnected sub-systems will make the subject of our further research leading to the solution for distributed control applied to system of systems.

To overcome the problem stated in the paper, a MPC based approach is proposed. It requires accurate models of each controlled node and used actuators. A neural network based nonlinear model of one of the controlled parts (water boiler heated by using gas and air) is presented in the paper. Our experiments have shown that using this model it is possible to predict the behavior of the plant for 5 minutes with high accuracy which will be just enough to overcome the delay problems.

The proposed technique can be implemented with Java programming language. Metso DNA control system is used for control of the considered district heat plant. It utilizes Function Block Language [8], [9] that provides possibility to create function blocks with Java code inside.

Acknowledgements

The work was partially supported by the European Union through the European Regional Development Fund and the target project

SF0140113As08, the Estonian Science Foundation Grant no. 8738 and the Estonian Doctoral School in Information and Communication Technology.

References

[1] K.J. Åström, T. Hägglund, “The future of PID control”, IFAC J. Control Engineering Practice, Vol. 9, pp. 1163-1175, 2001.

[2] D.P. Atherton, S. Majhi, „Limitations of PID controllers“, Proceedings of the American Control Conference, pp. 3843-3847, 1999.

[3] C. Brosilow and B. Joseph, "Techniques of Model-Based Control", Prentice Hall Professional, 2002, 680 p.

[4] N. Paraschiv, A. Baiesu, G. Stamatescu, “Using an Advanced Control Technique for Controlling a Distillation Column”, IEEE International Conference on Control and Automation, pp. 581-584, 2009.

[5] M. Zhuang, D. P. Atherton, “Optimum Cascade PID Controller Design for SISO Systems”, CONTROL’94 Conference Publications, No. 389, pp. 606-611, 1994.

[6] B. W. Bequette, “Process Control: Modeling, Design, and Simulation”, Prentice Hall PTR, Upper Saddle River, N.J., 769 p, 2003.

[7] B. Roffel and B. Betlem, “Advanced Practical Process Control”, Springer, Berlin, Heidelberg, 309 pp, 2004.

[8] M. Karaila, T. Systä, “Applying Template Meta-Programming Techniques for a Domain-Specific Visual Language – An Industrial Experience Report”, Proc. of the 29th International Conference on Software Engineering (ICSE 2007), pp.571-580, May 2007.

[9] M. Karaila, “Domain Specific Template-based Visual Language and Tools for Automatic Industries“, Tampere University of Technology, PhD thesis, 2008.

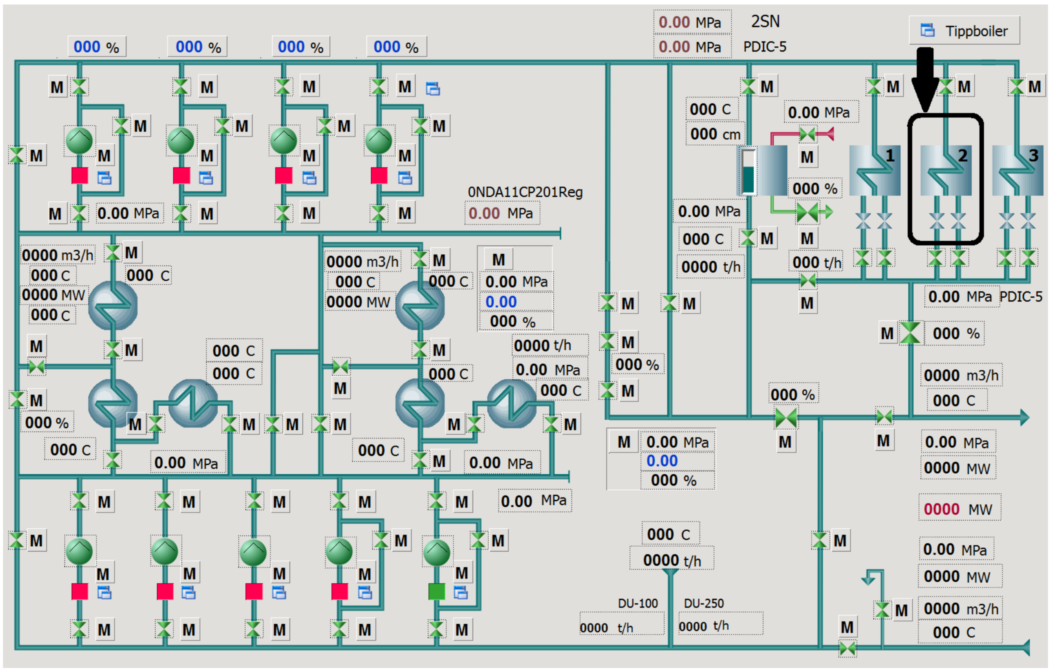


Figure 6. Plant layout in Metso DNA control system HMI

Appendix 2

Publication 2

Reference

V. Vansovits, E. Petlenkov, K. Vassiljeva, A. Tepljakov, and J. Belikov, “Application of MPC to industrial water boiler control system in district heat plant,” in Proc. 13th Int. Conf. Control Automation Robotics Vision (ICARCV), Dec. 2014, pp. 1609–1614.

Abstract

The main goal of this paper is to identify an industrial water boiler model and design a model predictive controller (MPC). The boiler model was identified from the real process data collected during a heating season. Controller was designed and tested in virtual environment and its performance was compared with performance of classical PI control algorithm that is currently used to control a boiler system. Use of the designed controller leads to significant improvement of accumulated output error.

Application of MPC to Industrial Water Boiler Control System in District Heat Plant

Vitali Vansovits, Eduard Petlenkov,
Kristina Vassiljeva and Aleksei Tepljakov
Department of Computer Control
Tallinn University of Technology
Ehitajate tee 5 A-II-319, 19086 Tallinn, Estonia
Email: vitali.vansovits at metso.com,
{eduard.petlenkov, kristina.vassiljeva,
aleksei.tepljakov} at ttu.ee

Juri Belikov
Institute of Cybernetics
Tallinn University of Technology
Akadeemia tee 21, 12618 Tallinn, Estonia
E-mail: jbelikov at cc.ioc.ee

***Abstract*—The main goal of this paper is to identify an industrial water boiler model and design a model predictive controller (MPC). The boiler model was identified from the real process data collected during a heating season. Controller was designed and tested in virtual environment and its performance was compared with performance of classical PI control algorithm that is currently used to control a boiler system. Use of the designed controller leads to significant improvement of accumulated output error.**

I. INTRODUCTION

The paper addresses a problem of control of a water boiler which is part of bigger combined heat power plant (CHP). The main objective of the CHP is to produce heat power for the nearest cities. CHP production capacity is over 200 MW of electrical power and over 800 MW of heat power (458 MW in combined production).

The boiler was installed in year 1978. This is an old model KVGM-100 producing 100 Gcal/h (116.3 MW) of heat power. Some major investments were made to renovate the boiler infrastructure several years ago, therefore it is now equipped with modern measurement and control devices as well as distributed control system (DCS). All the control applications are implemented in the DCS on the software level. It is possible to create or modify applications without interrupting the process using set of predefined or programmable (on Java language) function blocks. Nevertheless control methods used in boiler control at present day are quite conservative based on PI control algorithm.

Controlled variable in the main control loop of the boiler is the output temperature. The fuel burning process and transfer of heat power from furnace to water takes some time, so there is a time delay between gas flow on the furnace input and water temperature measurement on the boiler output. Since PI controller uses output error for control, then it is dependent on the delay. Therefore the integration time is set longer than measurement delay to avoid permanent overshoot. At the moment of process data collection integration time parameter was 240 seconds. As the ideal PI control algorithm is used $[c_{out} = K_p(e + \frac{1}{T_i}\int e\ \mathrm{d}t)]$ then the real integration time is almost 270 seconds ($K_p = 0.9$). Given parameters provide slow reaction of the controller to setpoint changes and disturbances.

As the first step of our research we decided to model the process for prediction purposes to reduce the delay effect. Neural network was used to identify the process model [9]. The obtained model provided an acceptable result for 5 minutes incremental prediction with 1 minute step, however it was not suitable for further research in simulation mode. The problem was an accumulated error that after the 10 steps of simulation resulted in oscillating behaviour with increasing amplitude.

As the next step we decided to use linear identification of the process. Using the identified model we proceeded with the design of the model predictive control, which is known to be widely used and efficient solution for many control tasks in industry. MPC theory has well established foundation [2] and a lot of various applications, see e.g. [3], [4], [7].

The rest of the paper is organised as follows. Sections II describes the process under control. Section III considers existing control algorithm and its problems. In Section IV model predictive controller general description is given. In Section V process model identification and MPC design procedures described and simulation results are presented. The paper is finalised with Conclusion and Discussion Sections.

II. COMBINED HEAT PLANT AND WATER BOILER

The boiler is a part of bigger CHP that includes also another two water boilers and three power units. General power plant layout can be seen in Fig. 1.

Abbreviations on the picture are as follows

WB	Water Boiler
PU	Power Unit
HP	Heat Power
EP	Electric Power

Water coming from district heat network is supplied to the first level of heat producing facility – running power units or bypass. After passing the first level it is supplied to the second

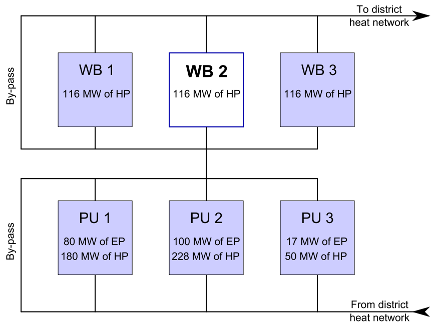


Fig. 1. Power plant layout

one – running water boilers or by-pass. Then heated water is supplied back to the district heat network.

Water boilers are fueled with natural gas. Originally all the boilers can also run on heavy oil. In year 1999 it was decided to decrease emissions of the power plant. Since heavy oil burning produces more emissions, all boilers were switched to use of natural gas. Heavy oil is still kept as a reserve fuel, but it has not been used since then.

CHP can function in different modes. In heat producing mode only water boiler(s) run(s). In combined production mode power unit(s) run(s) also. Simplified heat production process can be described from the physical laws as follows. Produced heat per time unit is

$$\Delta Q_{prod} = r \cdot \Delta m_f. \tag{1}$$

Consumed heat per time unit is

$$\Delta Q_{cons} = c \cdot \Delta m_w \cdot \Delta T_{water}. \tag{2}$$

These values relate to each other as follows

$$\Delta Q_{prod} = \Delta Q_{cons} + \Delta Q_{loss}, \tag{3}$$

where ΔQ_{loss} is heat lost with flue gases and other ways. The physical meaning of parameters is the following

r	gas specific heat of combustion
Δm_f	gas flow
c	water heat capacity
Δm_w	water flow
ΔT_w	water temperature increase

We are interested in water temperature increase on the boiler output

$$\Delta T_{water} = \frac{r \cdot \Delta m_f}{c \cdot \Delta m_w} - \frac{\Delta Q_{loss}}{c \cdot \Delta m_w}. \tag{4}$$

In principle, Q_{loss} is unknown, but it should correlate with gas and air flow – the more gas and air are supplied to the furnace the more flue gases produced the more heat lost with it. Rough estimation yields

$$\Delta T_{water} = (r - l) \cdot \frac{\Delta m_f}{c \cdot \Delta m_w}, \tag{5}$$

where l is unknown constant. Parameters c and r are also constants. Equation (5) is linear, if water flow (Δm_w) is constant and nonlinear otherwise. As will be seen later this can cause certain difficulties while designing linear MPC for the considered nonlinear process.

III. Existing control strategy and problem statement

Water boilers have two modes of operation--stationary and control. Only one boiler can run in control mode simultaneously. If more than one boiler run at the same time, then others are operated in stationary mode. Control mode necessity is caused by variable district heat network needs. Temperature setpoint and water flow change several times per day, depending on heat power requirements of the city caused by weather dependent heating of the houses and different use of hot water in the morning, afternoon and evening. Setpoint comes from the district heat system.

It was mentioned that PI controllers are used for the plant control. PI is single input single output (SISO) system, but plant has more inputs and outputs. Different combinations of control loops are used to utilize all signals. Plant controller layout is shown in Fig. 2.

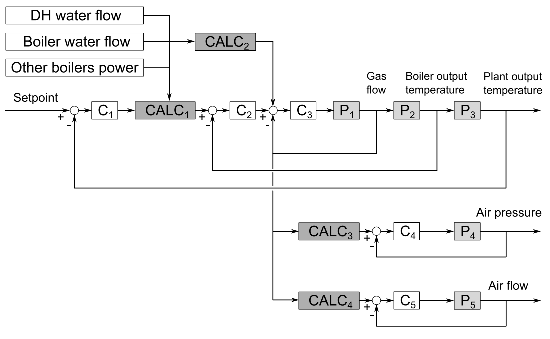


Fig. 2. Controller layout

This cascade PI controller regulates output of the plant loop P1/C1. Inner loop P2/C2 controls output of the water boiler. Loops P3/C3, P4/C4 and P5/C5 control certain gas and air actuators. Loops 3-5 are fast, Loops 1-2 are much slower compared to them. See Table I for all parameters' values of the controllers.

TABLE I
Controllers tuning parameters

Control loop	Kp	Ti
Air pressure 1	0.2	40
Air pressure 2	0.2	40
Air flow 1	0.8	18
Air flow 2	0.3	20
Air flow 3	0.4	29
Gas flow (C3)	0.25	20
Boiler output (C2)	0.9	240
Plant output (C1)	0.5	180

We do not consider the whole controller structure in this paper, but only boiler controller C2.

Boiler layout with sensors and actuators is presented in Fig. 3.

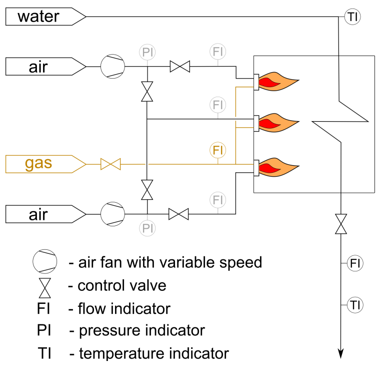


Fig. 3. Boiler layout

There is a bottom layer of controllers containing two air pressure controllers, three air flow controllers and one gas flow controller.

Since process is relatively slow in our case, controllers integration time has to be long enough to avoid permanent output overshoot. Due to this fact controller has very slow reaction to setpoint change. This is critical for the control loop C2, because its setpoint depends on master controller C1 and changes all the time.

IV. Model Predictive Controller

MPC uses model to predict process behavior and generate manipulated process input(s) so that it would be possible to minimize process output(s) deviation from the setpoint. Controller uses receding horizon H_p to predict process behavior p steps ahead and control horizon H_c to generate manipulated variables c steps ahead assuming that after cth step process input remains constant. The first value of control horizon is applied at present moment. Existing estimations are discarded in the next time instance and the same output prediction and input generation actions are repeated. Cost function is used to penalize output deviation from the setpoint. Also changes of the input vector can be penalized to control process dynamics.

$$V(k) = \sum_{i=1}^{H_p} \|z(k-i) - y(k-i)\|^2_{Q(i)} + \sum_{i=1}^{H_c} \|\Delta u(k-i)\|^2_{R(i)}, \quad (6)$$

where Q_i and R_i are weighting matrices and $\|x\|^2_A = x^T A x$.

If we cannot measure the full state vector or if the measured outputs consist of some linear combinations of the states, so that the states cannot be measured directly, then an observer can be used to estimate the state vector [2].

A very useful MPC property is its ability to obey constraints. It is possible to limit output, input and input changes. This property helps to take into account such physical limitation as tank levels (which cannot be below 0) or valve closing/opening limit (which cannot open less than 0% and more than 100%) etc. Constrains are defined in the form of linear inequalities

$$A \begin{pmatrix} \Delta U(k) \\ 1 \end{pmatrix} \leq 0, \quad (7)$$

$$B \begin{pmatrix} U(k) \\ 1 \end{pmatrix} \leq 0, \quad (8)$$

$$C \begin{pmatrix} Y(k) \\ 1 \end{pmatrix} \leq 0, \quad (9)$$

where A, B and C are matrices of constraints parameters. $\Delta U(k)$ and $U(k)$ are vectors of estimated changes of inputs and inputs values within control horizon, $Y(k)$ is a vector of estimated outputs within prediction horizon. By solving this system we can come to inequality, which limits inputs' moves only. So, we can keep all limits by constraining vector ΔU, as follows

$$\Omega \Delta U(k) \leq \omega(x(k), u(k-1)). \quad (10)$$

Here we come to the core of constrained MPC, an optimizer, which solves standard optimization problem known as quadratic programming

$$\min_{\theta} \frac{1}{2} \theta^T \Phi \theta + \phi^T \theta \quad (11)$$

subject to

$$\Omega \theta \leq \omega. \quad (12)$$

This is an optimization problem. A number of methods exists to solve it, e.g. *Active Set method* or *Interior Point method* [2].

V. MPC for water boiler

According to [4] MPC design procedure includes

1) definition of used inputs outputs
2) process data collection
3) process model design
4) configure MPC with initial parameters
5) testing in simulation
6) testing in real process in open loop
7) testing in close loop and final fine tuning

Current work covers five first steps of the list. Required inputs and outputs were selected after process analysis. Relevant process data was collected from several running periods.

A. Model design for process simulation and MPC

The process cannot be modeled from the first principles, because there is not enough information for this. Therefore we have to use identification methods based on real process data. MATLAB Identification Toolbox was used to create process models for simulation and for MPC. Different data sets were used for modeling to avoid ideal control with identical models. Also different data sets were used for modeling and validation in each case.

Process data of water boiler 2 (WB2) is available for the period of over 500 hours with 1 minute resolution. There were few running periods during this time. Only running period data was used for the identification. Start-up and shut-down period data was not used to avoid major nonlinearities and reduction of linear model quality.

Model for process simulation was identified from a data set where process variables' values fit into the following ranges

- water flow: 340–370 m^3/h;
- gas flow: 6400–10700 Nm^3/h;
- output temperature: 42–63 $^\circ\mathrm{C}$.

First, autoregressive with exogenous input (ARX) structure for the model was used. It showed fit to validation data $\sim 50\%$). With output error (OE) models fit to validation data $> 90\%$ was achieved, which is good enough. OE is one of the most popular model types, because it is often closer to reality than others [3]. OE model has the form

$$y(k) = \frac{B(q)}{F(q)}u(k) + v(k). \tag{13}$$

Its estimation is a little bit more complicate than ARX estimation, because it does not use output measurements (y), but output simulations ($\tilde{y}$) [3]

$$y(k) = \sum_{i=1}^{m} b_i u(k-i) - \sum_{i=1}^{m} f_i \tilde{y}(k-i). \tag{14}$$

One of the OE modelling methods is to start with ARX estimation (linear least square optimization) for $F(q)y = B(q)u + v(k)$, then filtering inputs and outputs through estimated filter $F(q)$

$$u^F(k) = \frac{1}{F(q)}u(k), \tag{15}$$

$$y^F(k) = \frac{1}{F(q)}y(k) \tag{16}$$

and finally making ARX estimation for filtered inputs $u^F(k)$ and outputs $y^F(k)$. MATLAB Identification Toolbox performs OE model estimation automatically.

It is not possible to start simulation from predefined state using OE, thus polynomial was converted to state-space model for convenience of use.

$$A = \begin{pmatrix} 1.80 & -0.80 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2.61 & -1.14 & 0.34 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} 0.25 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0.06 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$$C = \begin{pmatrix} -3.8 & -2.8 & 6.5 & 1.9 & -2.3 & 2.1 & -0.7 \end{pmatrix} \cdot 10^{-2}$$

Model output fit to validation data was 94.76%. Fig. 4 shows real process output and model output for the same real input values.

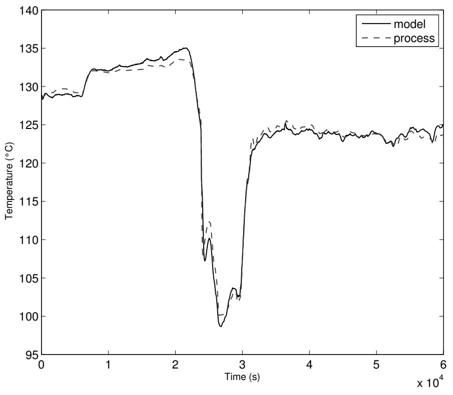


Fig. 4. Model and process outputs

Similar procedure was repeated to create linear model for MPC. Data set from another running period was used for this purpose

- water flow: 320–390 m^3/h;
- gas flow: 6000–12700 Nm^3/h;
- output temperature: 40–75 $^\circ\mathrm{C}$.

Linear state-space representation of identified model is

$$A = \begin{pmatrix} 1.86 & -0.95 & 0.35 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.25 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2.11 & -0.69 & 0.27 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} 0.25 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0.03 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$$C = \begin{pmatrix} -6.27 & -2.58 & 35.37 & 0.94 & 0.45 & -1.81 \end{pmatrix} \cdot 10^{-2}$$

Model output fit to validation data was 82.68%. MPC linear model was identified from a wider ranges of process data, so its quality is lower due to process nonlinearities. This is also a reason for the difference between process and MPC models.

B. MPC design

MATLAB MPC Toolbox was used to design model predictive controller based on identified model. The following inputs and outputs were selected for MPC design

- Gas flow (manipulated variable / process input);
- Water flow (measured disturbance / process input);
- Water temperature (measured output).

The parameters to be tuned are

- Prediction horizon H_p;
- Control horizon H_u;
- Overall coefficient W;
- Input move weight R_i;
- Output value weight Q_i.

The controller is going to manipulate gas flow by affecting underlying loop with actuator with certain physical limitations. It would be good to take these limitations into account to avoid saturations. From process data it is possible to obtain gas flow maximum (approximately 16000 m^3/h) and gas flow variance dynamics (± 2000 m^3/h per step). These values can be used to define MPC constraints.

Many simulations were made with different controller settings iterating to the most suitable parameters. The best performance of MPC in the selected operating range was achieved with the following settings

$$H_p = 6, H_u = 1, W = 1, R_i = 0.1, Q_i = 50.$$

Prediction and control horizons are short, because we need fast enough behavior of MPC. Keeping output temperature deviation close to setpoint is much more important than input move speed, that is why Q_i is much bigger than R_i. Due to the same reason overall coefficient is chosen as $W = 1$. It is used to balance output and input changes penalties. If W is close to 0, then MPC is slow, but more robust. If W is close to 1 then MPC is fast.

Designed MPC was used in Simulink environment to compare it with PI controller. Two loops were created in Simulink with the same simulation model – one for MPC test, another one for PI test. Classical serial PI layout was used for testing (as the same layout is used in the real control system). Original PI controller settings $K_p = 0.9$ and $T_i = 240$ were used in the beginning, but later Zeigler-Nichols tuning method [1] was applied to obtain more suitable controller parameters. Many simulations were made with these settings and also with some deviations from these settings looking for better performance. The best result with this model and PI controller was achieved with parameters $K_p = 1.35$ and $T_i = 405$. These were used for comparison of PI and MPC, see Fig. 5.

MPC showed approximately 10% lower accumulated error, see Fig. 6.

Checking MPC with Matlab tool $review(mpcobject)$ showed that controller is stable internally and in closed-loop assuming that process model is perfect. Simulations with imperfect model showed also its stability in selected operating range of process variables.

To see how model accuracy affects MPC performance we consider also the ideal case, when simulation model and MPC model are identical. Another MPC with the same parameters was designed on the base of the process simulation model. Its prediction is accurate, therefore overshoot is small and

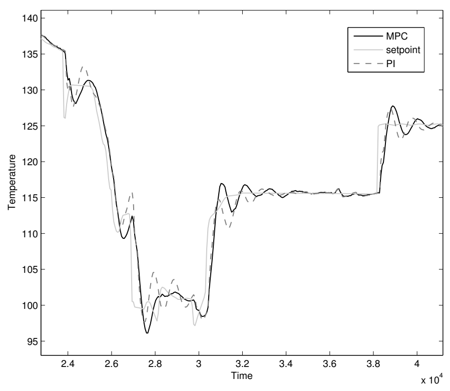


Fig. 5. MPC and PI comparison

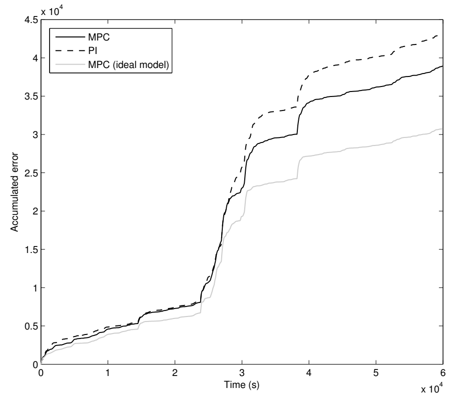


Fig. 6. Sum of absolute error

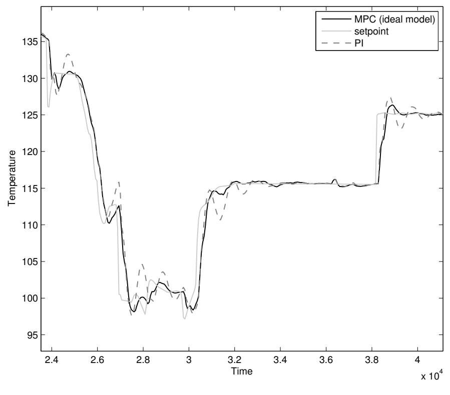


Fig. 7. MPC with ideal model and PI comparison

performance is better compared with MPC with different model, see Fig. 7.

This result shows importance of model accuracy and highlights the fact that linear MPC will act most efficiently around the linearization point and less efficiently in other operating range [5].

VI. Coclusion

Process model was identified based on real data. Obtained model was used to design model predictive controller for water boiler control. The controller was tested in virtual environment using real data. Even in case of a model identified around operating point different from that one process is running around, MPC showed approximately 10% better performance then PI controller. This result makes it reasonable to apply the MPC to real process control, especially in situation when suitable programming environment exists in the form of Java programmable function blocks that can be downloaded to the DCS and used instead of existing PI control loops.

VII. Discussion

This work resulted in an MPC design for water boiler control. The controller was compared to PI controller in simulation and showed better results. This was the main goal of the work to continue development in the replacement of PI with model based control. PI can be left in hot reserve following the controlled variable, but not affecting the process. It can be activated in case MPC gets to an infeasible region.

At first sight it could seem that replacement of a simple PI with a more complex MPC is not reasonable yielding only a relatively small performance improvement. We should not forget here that only part of existing control loop was replaced this time. We could improve performance of lower level control loop, but there is higher level controller that is still based on PI algorithm. It is obvious that better results can be obtained by replacing the whole plant controller with one MPC. A matter of further research is the design of multiple input multiple output MPC instead of cascade PI controller.

This work also showed that direct use of linear control methods for nonlinear plant can cause certain difficulties (e.g. overshoot). In the future development this should be taken into account and some nonlinear methods should be used, e.g. multiple linear models [3] or nonlinear model [3], [6], [8].

The general idea of our work is to create adaptive controller that can identify process model automatically and perform control actions accordingly. Similar topic was presented in e.g. [5] and [11], where it was also admitted that adaptive MPC can decrease implementation and maintenance costs. Since costs reduction and nonlinearity handling are in the list of the main challenges in MPC researches at present day [10] our current work provides suitable field for further development.

Acknowledgment

This research was supported by the Estonian Doctoral School in Information and Communication Technology.

References

[1] B.W. Bequette. *Process control: modeling, design, and simulation*. Prentice Hall Professional, 2003.

[2] J. Maciejowski. *Predictive control: with constraints*. Pearson education, 2002.

[3] O. Nelles. *Nonlinear system identification: from classical approaches to neural networks and fuzzy models*. Springer, 2001.

[4] S.J. Qin and T.A. Badgwell. A survey of industrial model predictive control technology. *Control engineering practice*, 11(7):733–764, 2003.

[5] V.R. Ravi, T. Thyagarajan, and M.M. Darshini. A multiple model adaptive control strategy for model predictive controller for interacting non linear systems. In *Process Automation, Control and Computing (PACC), 2011 International Conference on*, pages 1–8. IEEE, 2011.

[6] J.B. Rawlings. Tutorial: model predictive control technology. In *American Control Conference, 1999. Proceedings of the 1999*, volume 1, pages 662–676, 1999.

[7] J. Richalet. Industrial applications of model based predictive control. *Automatica*, 29(5):1251–1274, 1993.

[8] A. Tiagounov and S. Weiland. Model predictive control algorithm for nonlinear chemical processes. In *Physics and Control, 2003. Proceedings. 2003 International Conference*, volume 1, pages 334–339. IEEE, 2003.

[9] V. Vansovits, E. Petlenkov, K. Vassiljeva, and A. Guljajev. Identification of industrial water boiler for model predictive control of district heat plant. In *Electronics Conference (BEC), 2012 13th Biennial Baltic*, pages 315–318, 2012.

[10] Yu-Geng Xi, De-Wei Li, and Shu Li. Model predictive control - status and challenges. *Acta Automatica Sinica*, 39(3):222–236, 2013.

[11] Y. Zhu. A new adaptive mpc system. In *Advanced Control of Industrial Processes (ADCONIP), 2011 International Symposium on*, pages 447–449, 2011.

Appendix 3

Publication 3

Reference

V. Vansovits, A. Tepljakov, K. Vassiljeva, and E. Petlenkov, “Towards an intelligent control system for district heating plants: Design and implementation of a fuzzy logic based control loop,” in Proc. IEEE 14th Int. Conf. Industrial Informatics (INDIN), Jul. 2016, pp. 405–410.

Abstract

In this paper, we lay the foundation for a specialized Intelligent Control System (ICS) for district heating plants and investigate the problem of integrating particular knowledge into the developed system. In particular, we consider the fuzzy logic based control module. Based on control system knowledge, a particular control loop is developed and successfully implemented on a live industrial heating plant. The resulting automatic control system improves the performance of the existing control loop, allows to prolong the lifespan of industrial equipment by allowing to maintain it in a nominal operating condition, and alleviates the necessity for frequent manual control override by heating plant operators. The obtained knowledge forms a valuable asset to the ICS as a part of the corresponding module.

Towards an Intelligent Control System for District Heating Plants: Design and Implementation of a Fuzzy Logic based Control Loop

Vitali Vansovits, Aleksei Tepljakov, Kristina Vassiljeva, and Eduard Petlenkov
Department of Computer Control
Tallinn University of Technology
Ehitajate tee 5, 19086 Tallinn, Estonia
E-mail: `vitali.vansovits at a-lab.ee` and
`{aleksei.tepljakov, kristina.vassiljeva, eduard.petlenkov} at ttu.ee`

***Abstract*—In this paper, we lay the foundation for a specialized Intelligent Control System (ICS) for district heating plants and investigate the problem of integrating particular knowledge into the developed system. In particular, we consider the fuzzy logic based control module. Based on control system knowledge, a particular control loop is developed and successfully implemented on a live industrial heating plant. The resulting automatic control system improves the performance of the existing control loop, allows to prolong the lifespan of industrial equipment by allowing to maintain it in a nominal operating condition, and alleviates the necessity for frequent manual control override by heating plant operators. The obtained knowledge forms a valuable asset to the ICS as a part of the corresponding module.**

***Index Terms*—Intelligent control, Fuzzy logic, Industrial control, Heating plant**

I. Introduction

Control theory has formed a basis for implementation of intelligent control methods in the industry. One of the most important open issues lies in the combination of classical industrial controllers with knowledge based reasoning within a single programming framework [1]. This forms the motivation for our present work.

We restrict our attention to district heating plants. The motivation behind this choice is dictated by the importance of the role these play in Northern Europe [2], [3]. Practical implementation and verification of the ICS is also facilitated due to the rise of interest of local and international industrial entities towards these new developments.

The idea behind the design of an intelligent control system based on an underlying set of Computer-Aided Control System Design (CACSD) tools is not new [4], [5], [6] and some of the results have already been successfully applied in the industry [7] and, in particular, in critical heating applications [8]. The approach proposed in this paper aims at establishing a suitable software framework that leverages particular fields of expertise of the involved researchers, and then using it for creating a specialized ICS for efficient control of district heating plants. Towards that end, several studies have already been conducted. In [9] we have investigated the issue of neural network based industrial water boiler identification for Model Predictive Control (MPC). In [10], a suitable MPC was designed such that improved the performance of the heating plant.

In this paper, a fuzzy logic based approach is considered [11]. Fuzzy logic provides the necessary universal tools for the developed framework [12]. There are numerous industrial applications of fuzzy control. In [13] a fuzzy controller is designed as part of a wastewater treatment process. Of special interest in this work are applications related to heating processes, where fuzzy control of heat exchangers are investigated [14] and [15]. Study of literature also reveals that practically applicable fuzzy rule sets are usually not very complicated [16].

The main contribution of this paper lies in the design and experimental verification of a fuzzy logic based control loop in the context of a live heating plant. Our aim is to integrate the obtained knowledge into the fuzzy modeling and control module of the developed ICS.

The structure of the paper is as follows. In Section II the initial design of the ICS is presented and its relation to the industrial fuzzy control application is outlined. In Section III the specific industrial problem is detailed, and a corresponding fuzzy controller is designed, implemented, and verified on a live heating plant. The results are presented and analyzed. Finally, conclusions are drawn in Section IV.

II. Design of the Intelligent Control System

A. Description of the Prospective Software Platform

The prospective ICS shall have a modular structure depicted in Fig. 1. The modules are designed independently, but form a whole by means of interaction through the ICS core. The complete system is envisioned as a software package for providing control solutions for various loops in district heating plants. The modules are based around specific knowledge of the team of Control System Research Laboratory [17]. In particular, the following competences are considered:

- Nonlinear control design, dynamic linearization, and artificial neural networks [18], [19].

- Fractional-order modeling and control, including retuning existing PI/PID control loops with FOPI/FOPID controllers to improve performance [20]. This particular competence revolves around developing a CACSD system—FOMCON toolbox for MATLAB [21], [22]. Research shows that fractional-order calculus is very useful in heating control applications [23].
- Fuzzy logic based modeling and control—a part of this contribution.
- Optimal controller structure determination using genetic algorithms [24], for which a software solution is in development.
- Model predictive control [10].

The ICS will therefore be formed by a combination of classical control techniques with intelligent control methods by means of the interacting modules. This expert system will be capable of determining the most suitable control strategy for a particular part of a heating plant with respect to a predefined set of control quality criteria through collection and analysis of experimental data, analytical and nonanalytical modeling, and computer based simulation and verification.

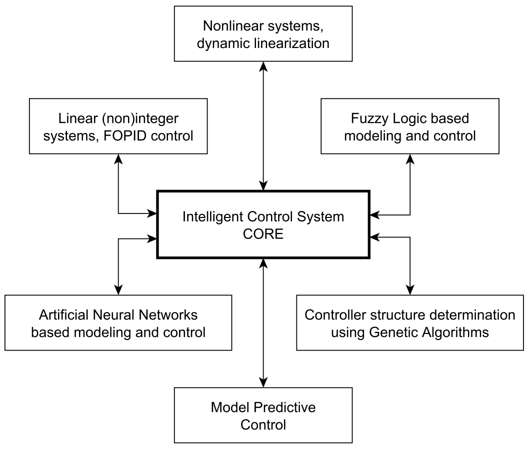

Nonlinear systems, dynamic linearization
Linear (non)integer systems, FOPID control
Fuzzy Logic based modeling and control
Intelligent Control System CORE
Artificial Neural Networks based modeling and control
Controller structure determination using Genetic Algorithms
Model Predictive Control

Fig. 1. Structure of the Intelligent Control System core

The initial implementation of the ICS will have the form of a CACSD software system, the novelty of which lies in the interconnected applications of various modules described above, and in the application of knowledge and data collected from different district heating plants in the specific region.

If implemented as an autonomous distributed control system (DCS), the modular structure of the proposed ICS fits well with the prospective design of Industry 4.0 applications [25]. Furthermore, in real industrial applications, a modular approach is desired because the resulting structure is easier to analyze. The interconnection of modules in the proposed ICS allows to tackle several problems at once with the modules being "aware" of the tools applied from other modules. In this work, we focus on the fuzzy logic based modeling and control module.

B. *Specifics of Fuzzy Logic based Control Design*

In this contribution, we are interested in a controller that is capable of accomplishing the following tasks put forth by a typical control problem in heating plants:

- It must ensure stable and correct operation of the overall control loop;
- It has to maintain nominal operating conditions for various particular pieces of equipment (e.g., boilers) involved in the process;
- It should make possible running the system in automatic mode.

Control loops may already be present to solve these problems, but may do so inefficiently and require frequent manual intervention from the plant operators. Retuning of the involved controllers may improve the performance of the loop [20]. However an additional controller can be used such that generates correcting signals based on the patterns of manual control override.

To design such a controller, the engineer must take into consideration the concrete specifications of the equipment used in process control, acceptable control action value ranges, as well as the resulting system dynamics. This specific knowledge must be transfered to the ICS. In what follows, we assume that the system then makes a decision based on the available data to use a fuzzy controller to solve the control problem. This scenario is described next in the context of a particular industrial application.

III. Industrial Application

A. *Process description*

In this article we consider a process control problem of a boiler-house plant located in Rapla town near Tallinn city in Estonia. The boiler-house includes three gas boilers and one biofuel (wood chips) boiler. Gas boilers 1 and 2 are currently not in use, so they are not discussed in present investigation. Main production facility of the boiler-house is biofuel boiler (number 4) as it is more economically efficient due to lower biofuel price compared to natural gas price.

The biofuel boiler is operated permanently during the heating season except the maintenance periods. It can produce up to 5 MW of heat power. If more heat power is required, than a gas boiler is turned on in parallel with the biofuel facility.

Boiler-house layout (excluding boilers 1 and 2) is presented in Figure 2.

There is an internal water pipeline where water circulates between heat exchanger and boilers. The heat exchanger is a connection point between internal and external pipelines. External pipeline is used for water circulation between the boiler-house and the district-heat (DH) network where heat power is delivered to households and industrial consumers.

The biofuel boiler has a grate furnace. Fuel feeding speed can be adjusted by changing the frequency of movement of a hydraulic piston that pushes fixed amount of fuel into the furnace each time.

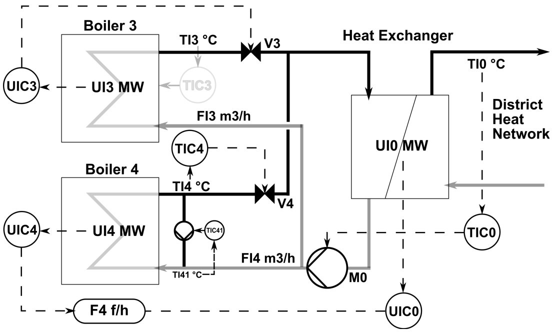


Fig. 2. Boiler-house control layout.

Boiler-house automation is made on the base of a distributed control system (DCS). Information exchange (measurements, control signals) with sensors and actuators is performed through input/output cards, Profibus and Modbus interfaces. All the controllers and logic algorithms are implemented as application programs. The DCS controls boiler 4 and pipeline operations. Each gas boiler has its own simple control based on a programmable logic controller (PLC). PLC control applications adjusts gas flow to keep boiler's outlet water temperature on a certain setpoint value. There is no data exchange between PLC and DCS, so it is possible to adjust gas boiler's heat production only by manipulating the water flow control valve V3.

There are two scenarios for boiler-house operation. The first one is applied when DH network demands less than 5MW of heat power and boiler 4 can be operated alone and produce enough heat. Control loop TIC0 keeps boiler-house water outlet temperature near the setpoint. Setpoint depends on outer air temperature as shown in Table I. A conventional PI controller is used to control boiler-house outlet water temperature. Controller signal is supplied to internal pipeline pump frequency converter. Higher frequency leads to the faster pump rotation speed rising water flow through the boiler 4. The higher the flow the more heat is transferred to the heat exchanger.

Table I. Boiler-house outlet water temperature dependency on outer air temperature. Intermediate values are linearly interpolated.

Outer air temperature	Boiler-house outlet temperature
-22°C	95°C
0°C	69°C
20°C	65°C

Heat power supplied to the DH network is measured and connected to PI controller UIC0 as a controlled variable. This controller keeps boiler 4 heat power production near the setpoint by adjusting the amount of fuel feeds per hour. Setpoint is calculated according to the following equation:

$$U_{SP} = c_w f_{w0}(T_{SP} - T_{in}), \tag{1}$$

where U_{SP} is DH network needs, c_w is water heat capacity, f_{w0} is DH network water flow,T_{in} is boiler-house input water tepemperature and T_{SP} is TIC0 setpoint.

Boiler 4 has recirculation pipeline that passes part of boiler 4 outlet water to its inlet mixing with water coming from the heat exchanger. This is needed to keep the temperature in all points inside the the boiler above 100°C and prevent moisture from the fuel to condensate on boiler walls on the furnace side causing corrosion. There is a specific limitation related to this. Boiler's outlet water temperature has to be above 110°C. If it is below this value, then recirculation flow is not enough to keep boiler inlet temperature greater than 100°C. At the same time outlet temperature cannot be above 120°C, because materials of the boiler and pipelines are not designed for long term operation in such conditions. This can decrease boiler's lifetime and rise maintenance costs.

As UIC0 PI controller can keep only one measurement near the setpoint, there is additional logic that prevents controller to decrease its output when outlet water temperature is below 115°C and goes down from one side and to increase its output when the outlet water temperature is above the same value and rises from another.

This control keeps outlet water temperature fluctuating in the range between 110°C and 120°C. At the same time boiler-house outlet water temperature deviates from the setpoint by 1°C, which is acceptable.

When DH network demand grows above 5MW the gas boiler 3 has to be turned on to provide additional heat.

In this scenario control loop for boiler-house outlet water temperature is the same TIC0, but boiler heat power production is now managed by another PI controller that has a static setpoint set by the operator. Normally its value is the maximum capacity of the boiler 4 5MW. Remaining required heat is produced by the boiler 3. There is a power controller UIC3 that controls water flow through the boiler. Boiler 3 has a separate PLC that keeps outlet water temperature near the setpoint 110°C. Produced heat power can be controlled from DCS by only adjusting valve V3 position.

This kind of control did not work due to hydraulic dependency of water flows between boilers 3 and 4. When DH network demand is over 5 MW, boiler 4 works on its maximum capacity and boiler 3 produces remaining heat. If DH network demand goes down, UIC3 adjusts V3 position in closing direction. Water flow through the boiler 3 decreases, but due to hydraulic dependency flow rate increases through the boiler 4 and washes out more than 5 MW of heat from it. As a consequence boiler 4 outlet water temperature falls down. At the same time heat power setpoint of boiler 3 decreases more, because boiler 4 heat production measurement goes above 5 MW. Valve V3 closes more because of this reason increasing boiler 4 flow more. At the same time UIC4 additional logic forces it to increase feed rate to rise boiler's outlet water temperature causing boiler to function above its maximum capacity.

An additional control loop TIC4 was used with testing purposes to control boiler 4 outlet water temperature by adjusting valve V4, but this caused other problems in process control. In the same situation when DH network demand decreases flow through the boiler 4 rises, temperature falls down and valve V4 starts to close under control of TIC4. This causes increase of the flow through the boiler 3 and valve V3 continues to close. Because of this race condition the whole flow through the boilers decreases, boiler-house outlet temperature starts to fall down. TIC0 increases speed until achieves 100%. Saturated control loop cannot keep boiler-house outlet water temperature close enough to the setpoint. In addition, motor working with 100% speed consumes maximum electricity causing increase of own costs of the boiler-house production process. Race condition between TIC4 and UIC3 ended on approximately equal heat production in boilers 3 and 4. In case of DH network demand of 7MW it meant underload of boiler 4 and use of additional gas fuel in boiler 3 due to this.

B. *Fuzzy controller*

It would be possible to use additional logic to try to keep both process parameters—boiler 3 heat production and water flow through the boiler 4—on desired level, but at the same time it is a good opportunity to apply fuzzy controller that is natural for multiple input single output (MISO) systems.

Our target is to decrease heat production of boiler 3 when the needs of DH network decrease. We are doing it by closing valve V3 and decreasing the flow through the boiler 3. At the same time we would not like to increase flow through boiler 4. So, we have system with two inputs and one output—valve V3.

We decide to apply a simple fuzzy controller for this case using three triangular membership functions (MSF) for each input and five actions for output. There are nine rules used (see Table II).

Simple triangular MSF are selected for this application (see Figure 3).

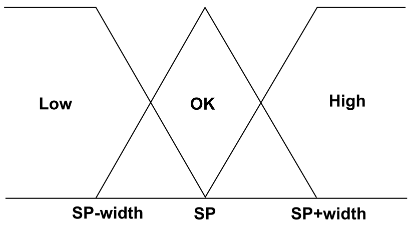


Fig. 3. Membership function

The first input is heat power of the boiler 3. It is changing in time, so MSF has to be changeable. To build the MSF we define two parameters—setpoint (SP) and MSF width. Using these parameters we build three MSFs. Middle one is "OK", left one for low values of measurement and right one for high values. Left MSF has value 1 from minus infinity to SP-width and then decreases linearly to 0 by achieving SP. "OK" MSF increases linearly from 0 to 1 while moving from SP-width to SP and decreases linearly from 1 to 0 while moving from SP to SP+width. Right MSF increases linearly from 0 to 1 while moving from SP to SP+width and then has value 1 until infinity. Setpoint value for the first input is calculated on the base of DH network needs:

$$U_3 = U_0 - U_4, \tag{2}$$

where U_4 is boiler 4 heat production, U_0 is DH network demand, U_3 is required boiler 3 heat production. MSF width is defined manually as 0.4.

Same mechanism is applied to the second input—water flow through the boiler 4. Required value is calculated on the base of other process parameters:

$$f_{SP} = \frac{U_4}{c_w(T_{out} - T_{in})}, \tag{3}$$

where U_4 is boiler 4 heat power, c_w is water heat capacity, T_{in} is boiler 4 input water temperature and T_{out} is boiler 4 outlet water temperature. So, setpoint value is equal to f_{SP} and MSF width we define manually as 14.

There are five possible actions defined: *remove much*, *remove little*, *no action*, *add little*, *add much*. Following nine rules are defined:

Table II. Fuzzy rules.

Rule	Input 1	Input 2	Action
1	Low	Low	No action
2	Low	OK	Add little
3	Low	High	Add much
4	OK	Low	Remove little
5	OK	OK	No action
6	OK	High	Add little
7	High	Low	Remove much
8	High	OK	Remove little
9	High	High	No action

As it was mentioned earlier, closing of the gas boiler outlet flow valve leads to reduction of water flow and heat production, because gas boiler's PLC control keeps outlet temperature constant. Less flow with the same temperature means less heat and vice versa. At the same time decreased flow through the boiler 3 while internal pipeline pump speed is constant causes increase of flow rate through the boiler 4. Taking these facts into account it is possible to describe following reasoning for definition of the fuzzy rules.

Rule 1. When boiler 3 produces less power than needed and flow through the boiler 4 is too low then no actions are needed. Due to low heat production boiler-house outlet water temperature will drop and TIC0 will increase internal water flow pump M0 speed. Flow through the boilers 3 and 4 will rise and boiler 3 will increase heat production as well.

Rule 2. If boiler 3 heat production is low and boiler 4 water flow is OK then it is possible to open valve V3 a bit to increase boiler 3 heat production while affecting boiler 4 water flow in a minor way.

Rule 3. If boiler 3 heat production is low and boiler 4 water flow is high then there is obvious need to open valve V3 much to increase boiler 3 heat production and decrease boiler 4 water flow.

Rule 4. If boiler 3 heat production is OK and boiler 4 water flow is low then valve V3 can be closed a bit to increase boiler 4 water flow without affecting boiler 3 heat production too much.

Rule 5. If both inputs are OK then no actions are needed.

Rule 6. If boiler 3 heat production is OK and boiler 4 water flow is high then valve V3 can be opened slightly to decrease boiler 4 water flow without affecting boiler 3 heat production too much.

Rule 7. If boiler 3 heat production is high and boiler 4 water flow is low then valve V3 needs to be closed much to shorten boiler 3 heat production and increase boiler 4 water flow.

Rule 8. If boiler 3 heat production is high, but boiler 4 water flow is OK, valve V3 can be slightly closed to reduce boiler 3 heat production without affecting boiler 4 water flow too much.

Rule 9. When boiler 3 produces more power than needed and flow through the boiler 4 is too fast, no actions are needed. Due to excessive heat generation boiler-house outlet water temperature will rise above the setpoint and TIC0 will decrease internal water flow pump M0 speed. Flow through the boilers 3 and 4 will be diminished and boiler 3 will decrease heat production as well.

Sugeno-type fuzzy inference [26] is selected to produce fuzzy controller output signal. "Product" AND function is used to calculate firing strengths s_i of each rule *i* (*i = 1, ... , N*, where *N* is number of rules). Rule table defines action a_i for each rule, where every action has its own output level $l(a_i)$. Final control *con* is calculated according to 4.

$$con = \frac{\sum_{i=1}^{N} w_i l(a_i)}{\sum_{i=1}^{N} w_i} \tag{4}$$

Controller functions in incremental mode, *con* defines change of controller output per second.

Output levels are selected by try & error method. Their values are shown in Table III.

Table III. Output levels.

	Action	Output level
1	Reduce much	-0.12
2	Reduce little	-0.05
3	No action	0
4	Add little	0.05
5	Add much	0.12

C. Results

As it was written earlier before implementation of fuzzy control there was a PI controller for boiler 3 heat power control. It controlled heat production without paying attention to water flow to boiler 4. As boilers 3 and 4 are hydraulically dependant, manipulation of boiler 3 valve caused increase of boiler 4 water flow leading to decrease of boiler 4 outlet temperature. Additional logic of boiler 4 heat power controller prevents it from decreasing fuel feed rate, if outlet temperature is low. Due to this reason boiler 4 produced 6MW of heat power for a few hours (Figure 4a), which is not allowed from technological point of view. Maximum heat production can be 5.5MW for short period of time. Because of long overload boiler 3 heat power controller was turned into manual mode. Heat production of boiler 3 was controlled by the operator in manual mode for few days until implementation of fuzzy controller.

After implementation of fuzzy logic boiler 3 heat power controller started to take into account flow through the boiler 4 keeping heat production as close as possible to the setpoint and supplying suitable amount of water to boiler 4 for its proper operation on maximum load. Since this moment all the controllers were switched to auto mode keeping all the process parameters close to their setpoints. Boiler 4 heat production was close to 5MW and outlet temperature close to 115°C, see Figure 4b.

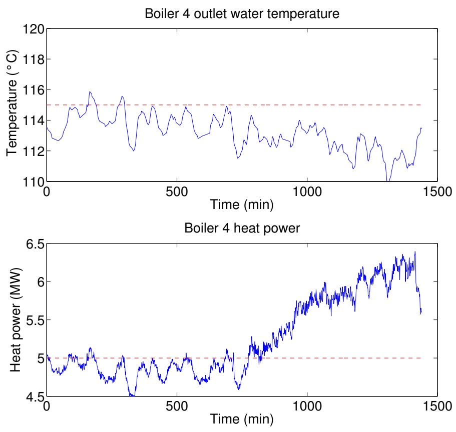
In the bottom trend of Figure 4b it can be seen that boiler 4 heat production variation frequency reduced since some moment. This was caused by decreasing firing strengths of fuzzy controller rules. Probably it is worth to lower these strengths more to make variations smoother. Unfortunately changed weather conditions do not allow to test modifications immediately. DH network demand is less than 5MW and only boiler 4 is in operation nowadays. Next tuning possibility will apparently be available next season.

IV. Conclusions

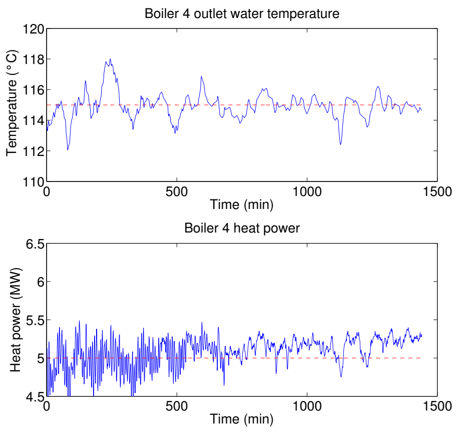
In this paper, we proposed an initial structure of an ICS designed for creating and modifying control loops in district heating plants. The main contribution of the paper was the design and practical implementation of an industrial fuzzy controller for a heating plant which is a valuable addition to the fuzzy control module of the ICS. The industrial application was tested and was found to improve the performance of the underlying control loop which is supported by experimental evidence. In particular, the new controller allows to properly fulfill the control task, keep the equipment in the correct operating point thus prolonging its lifespan, and prevent the necessity for frequent manual control override. Therefore, the requirements for the controller put forth by a specific heating element and described in this paper are satisfied, and the knowledge is preserved to form a part of the prospective ICS.

Acknowledgements

We would like to thank AS Erakute for allowing to use the data collected from the Rapla district heating plant for the purposes of this research.



(a) Heat production before implementation of fuzzy control.



(b) Heat production after implementation of fuzzy control.

Fig. 4. Comparison of boiler 4 control loop performance.

REFERENCES

[1] V. Vyatkin, "Software engineering in industrial automation: State-of-the-art review," *IEEE Trans. Ind. Informat.*, vol. 9, no. 3, pp. 1234–1249, 2013.

[2] "Directive 2010/31/EU of the European Parliament and of the Council of 19 May 2010 on the energy performance of buildings," 2010.

[3] "Directive 2012/27/EU of the European Parliament and of the Council of 25 October 2012 on energy efficiency," 2012.

[4] C. D. Tebbutt, *Expert aided control system design*. Springer-Verlag London, 1994.

[5] A. Zilouchian and M. Jamshidi, *Intelligent control systems using soft computing methodologies*. CRC Press, Inc., 2000.

[6] W. Yu, *Recent advances in intelligent control systems*. Springer, 2009.

[7] S. Mitra, R. Singh, and A. Mondal, "An expert system based process control system for silicon steel mill furnace of rourkela steel plant," in *Emerging Applications of Information Technology (EAIT), 2014 Fourth International Conference of*, 2014, pp. 29–33.

[8] L. Ma, Z. Wang, M. Zhang, and K. Lee, "Neural network inverse models of supercritical boiler unit for intelligent coordinated controller design," in *Intelligent Control and Automation (WCICA), 2014 11th World Congress on*, June 2014, pp. 1201–1206.

[9] V. Vansovits, E. Petlenkov, K. Vassiljeva, and A. Guljajev, "Identification of industrial water boiler for model predictive control of district heat plant," in *Electronics Conference (BEC), 2012 13th Biennial Baltic*, Oct 2012, pp. 315–318.

[10] V. Vansovits, E. Petlenkov, K. Vassiljeva, A. Tepljakov, and J. Belikov, "Application of MPC to industrial water boiler control system in district heat plant," in *Control Automation Robotics Vision (ICARCV), 2014 13th International Conference on*, Dec 2014, pp. 1609–1614.

[11] K. Passino and S. Yurkovich, *Fuzzy Control*. Addison-Wesley, 1998.

[12] B. Bose, "Expert system, fuzzy logic, and neural network applications in power electronics and motion control," *Proc. IEEE*, vol. 82, no. 8, pp. 1303–1323, 1994.

[13] W. Tang, Z. Wang, Q. Feng, and M. Wang, "Application of fuzzy expert control to APMP pulping wastewater treatment process of aerobic," in *Mechatronics and Automation (ICMA), 2010 International Conference on*, Aug 2010, pp. 339–344.

[14] A.-V. Duka and S.-E. Oltean, "Fuzzy control of a heat exchanger," in *Automation Quality and Testing Robotics (AQTR), 2012 IEEE International Conference on*. IEEE, 2012, pp. 135–139.

[15] A. Vasickaninova, M. Bakosova, A. Meszaros, and J. Oravec, "Fuzzy controller design for a heat exchanger," in *Intelligent Engineering Systems (INES), 2015 IEEE 19th International Conference on*, 2015, pp. 225–230.

[16] W. Wenbo and G. Na, "Design and research of intelligent control system for heating network flows," in *Mechatronic Sciences, Electric Engineering and Computer (MEC), Proceedings 2013 International Conference on*, 2013, pp. 318–321.

[17] A. Tepljakov. (2015) Official website of Alpha Control Laboratory. [Last access time: 12.05.2015]. [Online]. Available: http://www.a-lab.ee/

[18] J. Belikov and E. Petlenkov, "NN-SANARX model based control of a multi tank liquid-level system," *International Journal of Computational Intelligence Systems*, vol. 8, no. 2, pp. 265–277, 2015.

[19] K. Vassiljeva, A. Tepljakov, and E. Petlenkov, "NN-ANARX model based control of liquid level using visual feedback," in *Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on*, July 2015, pp. 133–138.

[20] A. Tepljakov, E. Petlenkov, J. Belikov, and E. A. Gonzalez, "Design of retuning fractional PID controllers for a closed-loop magnetic levitation control system," in *ICARCV 2014 : The 13th International Conference on Control, Automation, Robotics & Vision*, 2014, pp. 1345–1350.

[21] A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: a MATLAB toolbox for fractional-order system identification and control," *International Journal of Microelectronics and Computer Science*, vol. 2, no. 2, pp. 51–62, 2011.

[22] ——. (2011) FOMCON toolbox. [Online]. Available: http://www.fomcon.net/

[23] H. Yali and G. Ruikun, "Application of fractional-order model reference adaptive control on industry boiler burning system," in *Intelligent Computation Technology and Automation (ICICTA), 2010 International Conference on*, vol. 1, 2010, pp. 750–753.

[24] K. Vassiljeva, E. Petlenkov, and J. Belikov, "GA based optimization of NN-SANARX model for adaptive control of nonlinear systems," in *Neural Networks (IJCNN), The 2012 International Joint Conference on*, June 2012, pp. 1–8.

[25] M. Hermann, T. Pentek, and B. Otto, "Design principles for Industrie 4.0 scenarios: a literature review," *Technische Universität Dortmund, Dortmund*, 2015.

[26] M. Sugeno, *Industrial applications of fuzzy control*. Elsevier Science Inc., 1985.

Appendix 4

Publication 4

Reference

V. Vansovits, B. I. Godoy, A. Tepljakov, K. Vassiljeva, and E. Petlenkov, "Model-based control design for a district heating plant," in Industrial Informatics (INDIN), 2017 IEEE 15th International Conference on. IEEE, 2017, pp. 615–620.

Abstract

This paper describes the design of unconstrained model predictive controller (MPC) and the corresponding simulation environment. The resulting controller is developed in Java programming language and implemented as a standalone application. The idea of this work is to test an MPC application in a distributed control system (DCS) run-time environment in a simulation mode before it can be implemented in the real process. Results show that developed MPC application is capable of controlling the simulated process in a stable manner and perform better than a PI controller under same conditions.

Model-based Control Design for a District Heating Plant

Vitali Vansovits, Boris I. Godoy, Aleksei Tepljakov, Kristina Vassiljeva, and Eduard Petlenkov
Centre for Intelligent Systems
Department of Computer Systems
Tallinn University of Technology
Ehitajate tee 5, 19086 Tallinn, Estonia
E-mail: vitali.vansovits at a-lab.ee, bgodoyt at gmail.com and
{aleksei.tepljakov, kristina.vassiljeva, eduard.petlenkov} at ttu.ee

***Abstract*—This paper describes the design of unconstrained model predictive controller (MPC) and the corresponding simulation environment. The resulting controller is developed in Java programming language and implemented as a standalone application. The idea of this work is to test an MPC application in a distributed control system (DCS) run-time environment in a simulation mode before it can be implemented in the real process. Results show that developed MPC application is capable of controlling the simulated process in a stable manner and perform better than a PI controller under same conditions.**

I. Introduction

MPC is a well-known control strategy that has had an incredible development since the late 80s, see e.g. [2], [5], [6], [11]. A good overview of the whole topic is given in [14]. MPC can be found in diverse industries such as: automotive [3], chemical [4], [12], [16], and in power electronics and drives [19], just to mention a few. Nevertheless the use of this kind of control is not yet widely spread giving priority to classical PI(D) controllers in many industrial applications. In this paper, we consider the development of a standalone MPC application and its implementation in a real process control on its first step—testing in simulation environment built in real DCS.

One of the processes controlled by PI control loops is a water boiler of a combined heating plant (CHP) near the city of Tallinn, Estonia. Plant production facilities include a waste to energy (WtE) power unit that produces heat and electricity, and a gas-fueled water boiler that produces only heat. Power plant is controlled by means of DCS that can maintain a high level of automation. WtE unit produces 50 MW of heat power at its maximum capacity. When this heat is below city demand, a water boiler is started. Water boiler produces up to 116 MW of heat. Power unit is normally operated at a constant load and water boiler produces variable heat depending on the district heat (DH) network's demand. The biggest challenge is to control the water boiler so that it could change the power production as fast as DH network load requires. At the moment, boiler heat production is controlled with a cascade PI-controller consisting of three cascades: plant outlet water temperature control, water boiler outlet temperature control and boiler gas flow control. This loop has a very slow response and it cannot handle process disturbances of the plant.

We propose to use an MPC strategy to control the outlet temperature of the plant. The manipulated variable is the gas flow. Using the gas flow as the manipulated variable is not a new approach, since it was previously proposed in [17]. However, in this work, we extend this previous research by making the following contributions: (i) we prepare a real implementation of the proposed control strategy using MPC and, (ii) control the whole plant outlet temperature, not only the boiler temperature as originally proposed in [17].

The structure of the paper is as follows. In Section II, we give motivation to this work. In Section III, the process is explained and a model is obtained. In Section IV, we design an MPC strategy for the particular application. In Section V, we consider aspects of simulation and the real implementation. In Section VI and VII, conclusions and discussion are given, respectively.

II. Motivation

Natural gas is an extremely convenient fuel with a stable calorific value. The gas boiler process is nonlinear, however the non-linearity does not depend on the fuel quality. In contrast, bio-fuel boiler is a nonlinear process which does depend on the type of fuel being used. Moisture and raw material affect calorific value of the fuel [13]. Varying non-measured calorific value makes bio-fuel boiler modeling complicated. MPC would not perform appropriate in such a boiler control because a linear process model can be too different from the actual process when fuel is good (high calorific value) or when fuel is bad (low calorific value). Calorific value of the fuel is not measured on-line, so it is not possible to use it in control. That is why we considered a fuzzy control strategy for the bio-fuel boiler in one of our previous works [18].

On the contrary, if gas boiler fuel does not influences process nonlinearities, and if we are confident that the identified model is accurate enough within a certain range of operation, then we can consider applying an MPC strategy.

In a previous work [17], we have shown, through simulations in Matlab, that there may be some improved performance

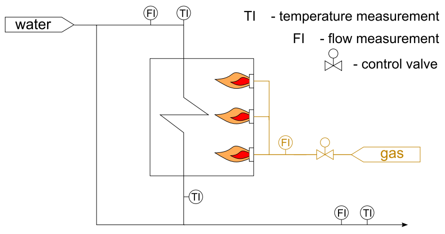


Fig. 1. Process diagram.

TABLE I
MODEL INPUTS AND OUTPUTS.

Input/output	Description
Input 1	Gas flow
Input 2	Boiler water flow
Input 3	Plant water flow
Input 4	Inlet water temperature
Output 1	Boiler outlet temperature
Output 2	Plant outlet temperature

of the process when an MPC strategy is considered. This advantage comes from comparing performance of MPC with a standard PI controller. In this work, we want to expand on these findings, and propose a real implementation of the control strategy suitable for the real-life process.

III. PROCESS MODEL

Process layout is shown in Fig. 1. Only devices related to the considered control loop are shown.

After passing through the power unit (not shown in the layout), part of the water flow with measured temperature is delivered to the water boiler, where it is heated with natural gas. Remaining water flows through a by-pass line. After exiting the boiler both flows are mixed and resulting flow and temperature are measured on the plant output. Inputs and outputs listed in Table I are used for deriving the dynamic model of the process.

For testing purposes, two different models of the process at different operating points are created. One model is for process simulation and the other one for MPC design. This is with the purpose of having a simulation closer to reality, where the process model normally is not able to reflect exactly the real process behavior. If we succeed in such a challenging simulation, then it is highly probable that MPC will also work for the real process.

We propose to identify a model in state-space:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) + Du(k) \end{aligned}, \tag{1}$$

where $u(k) \in \mathbb{R}^{n_u}$, $x(k) \in \mathbb{R}^{n_x}$, and $y(k) \in \mathbb{R}^{n_y}$ are the input, state, and the output of the system, respectively.

Using the model equations given in (1), we propose to identify two models of the process at different operating points.

TABLE II
MODELS OPERATING POINTS COMPARISON.

Input/output	Process		MPC	
	Min	Max	Min	Max
Gas flow, Nm^3/h	3000	11500	5500	13000
Boiler water flow, m^3/h	1220	1330	1295	1370
Plant water flow, m^3/h	1500	4000	2900	4600
Inlet water temperature, $^\circ C$	58	75	55	64
Boiler outlet temperature, $^\circ C$	80	128	95	135
Plant outlet temperature, $^\circ C$	74	86	76	94

Upper and lower limits of the process data used for identification are listed in Table II. This table shows some overlapping in the operation points.

Models were identified with the Matlab System Identification Toolbox [8] using state-space models by means of Predictive Error Method (PEM) estimation.

Here, we distinguish two cases in the modeling.

A. Process model for simulation

Process model was identified using data sets collected from the process in the periods 2014.02.27 16:53 – 2014.03.06 22:48 and 2015.10.30 0:00 – 2015.11.06 12:52, 7634 and 6199 samples, respectively. Data sampling interval is 1 minute which is enough to obtain a reliable representation of such a slow process. Resulting model is:

$$A = \begin{pmatrix} 0.3589 & -0.07221 & -0.1482 & 0.05587 \\ 0.2856 & 0.9659 & -0.1358 & -0.2506 \\ -0.7532 & 0.1398 & 0.4404 & -0.08773 \\ -0.03067 & 0.2475 & -0.207 & 0.7836 \end{pmatrix} \tag{2}$$

$$B = \begin{pmatrix} -1.266 & -0.9273 & 3.994 & -728.6 \\ 2.514 & -2.471 & 1.859 & 676.6 \\ 1.853 & 3.958 & 7.8 & 47.22 \\ 1.797 & 14.56 & 0.7359 & 714.9 \end{pmatrix} \cdot 10^{-5} \tag{3}$$

$$C = \begin{pmatrix} -64.75 & -18.78 & 46.79 & -31.42 \\ -29.62 & -37.28 & 13.03 & -5.59 \end{pmatrix} \tag{4}$$

$$D = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{5}$$

B. Process model for MPC

Model for MPC is identified from the data collected during the period of 2015.02.11 0:00 – 2015.02.24 23:59. 13502 samples were used. Sampling time is also 1 minute. Resulting model is as follows:

$$A = \begin{pmatrix} 0.7598 & 0.01881 & 0.3164 \\ -0.02674 & 0.86 & 0.2066 \\ 0.3779 & -0.1196 & -0.1004 \end{pmatrix} \tag{6}$$

$$B = \begin{pmatrix} 5.804 & -5.332 & 0.9552 & 605.7 \\ 2.572 & 7.023 & -3.891 & 557.3 \\ -14.19 & 17.86 & -1.721 & -1161 \end{pmatrix} \cdot 10^{5} \tag{7}$$

$$C = \begin{pmatrix} 54.09 & 2.848 & 19.75 \\ 16.54 & 27.72 & 8.731 \end{pmatrix} \tag{8}$$

TABLE III
MPC JAVA APPLICATION THREADS.

Thread	Function
1	Command line scanner to read user's commands
2	Modbus/TCP interface to communicate with DCS
3	MPC to calculate optimal process inputs

$$D = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{9}$$

This model will be used for the design of an MPC described in the next section.

IV. Model Predictive Controller application

A. Java application

MPC is implemented in Java programming language as a command line application. As a first stage, we have chosen to implement the unconstrained case. The constrained case requires further programming in Java, and the implementation of an interior-point method, see e. g. [15], which makes the Java programming more complex. It is considered to implement constrained MPC in a future development of this work.

Java is not the most efficient programming language [10], [20], but it is convenient, reliable and supports cross-platform execution. The latter characteristic can be useful in the future, providing more freedom for a real process control implementation. Java program is compiled to the platform independent byte code that is executed in platform dependent Java Virtual Machine (JVM). This type of execution requires compilation from byte code to machine code in real-time. Each application method is compiled on the fly when it is called by the program. This can cause processing overhead for real-time compilation. Java uses Jast-In-Time (JIT) compilation to avoid it. JIT compiles program methods once during program execution when they are called and stores them in memory for later use. As our program is cyclical, processing overhead for online compilation should be minimal.

We need an MPC application that supports the following functions:

- read configuration from the file,
- communicate with the process via Modbus/TCP,
- calculate optimal inputs for the process,
- interact with the user via command line interface (CLI).

As Java is multi-threading programming language by its nature, it is possible to build these functions as separate threads. Application structure is shown in Fig. 2. The main thread that starts all other functions is created first. It starts separated threads for the functions listed in Table III.

MPC thread reads process model and MPC parameters from an XML configuration file.

Modbus thread reads its configuration parameters from another XML file. Modbus implementation is based on Jamod library [7].

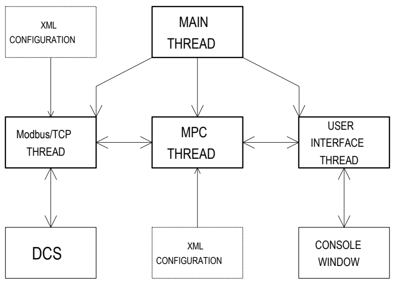


Fig. 2. MPC application structure

The CLI shows model, MPC parameters and Modbus parameters loading status after the application has started up. By default it does not provide any other information, but it accepts a variety of commands from the user to force the application to write useful information to CLI, such as calculated optimal process input, prediction vector, process input moves, etc.

When the application is running, it reads process values from the DCS via Modbus interface, calculates optimal process input and writes it back to the DCS via Modbus.

B. MPC algorithm

In MPC we aim to optimize a cost function defined as [9]:

$$V(k) = ||Z(k) - T(k)||^2_Q + ||\triangle U(k)||^2_R, \tag{10}$$

where $Z(k)$ is the outputs prediction vector within prediction horizon H_p, $T(k)$ is set points trajectory within H_p and $\triangle U(k)$ is vector of process input moves (changes) within control horizon H_c. $Z(k)$ is calculated as:

$$Z(k) = \varPhi X(k), \tag{11}$$

where

$$\varPhi = \begin{pmatrix} C & 0 & \cdots & 0 \\ 0 & C & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & C \end{pmatrix}.$$

The predicted states can be written as:

$$X(k) = \hat{\Psi} x(k) + \hat{\Upsilon} u(k-1) + \hat{\Theta} \triangle U(k), \tag{12}$$

where

$$X(k) = \begin{pmatrix} x(k+1|k) \\ \vdots \\ x(k+H_p|k) \end{pmatrix}, \quad \triangle U(k) = \begin{pmatrix} \triangle u(k|k) \\ \vdots \\ \triangle u(k+H_u-1|k) \end{pmatrix},$$

$$\hat{\Psi} = \begin{pmatrix} A \\ \vdots \\ A^{H_p} \end{pmatrix}, \quad \hat{\Upsilon} = \begin{pmatrix} B \\ \vdots \\ \sum_{i=0}^{H_p-1} A^i B \end{pmatrix},$$

$$\hat{\Theta} = \begin{pmatrix} B & \cdots & 0 \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{H_u-1} A^i B & \cdots & B \\ \sum_{i=0}^{H_u} A^i B & \cdots & AB+B \\ \vdots & \vdots & \vdots \\ \sum_{i=0}^{H_p-1} A^i B & \cdots & \sum_{i=0}^{H_p-H_u} A^i B \end{pmatrix}.$$

$\hat{\Psi}$ is the matrix used to calculate the effect of current states to the states in the future within the prediction horizon H_p, $\hat{\Upsilon}$ is the matrix used to calculate the effect of the latest process inputs to the future states and $\hat{\Theta}$ is the matrix used to calculate the effect of future input changes to future states. H_u is control horizon (number of control actions into the future),. $u(k+i|k) = 0$ for each $i > H_u$ with $H_u \leq H_p$. Thus:

$$Z = \Psi x(k) + \Upsilon u(k-1) + \Theta \Delta U(k), \quad (13)$$

where $\Psi = \Phi\hat{\Psi}$, $\Upsilon = \Phi\hat{\Upsilon}$ and $\Theta = \Phi\hat{\Theta}$.

Tracking error between free response and tracking trajectory in this case can be written as:

$$E(k) = T(k) - \Psi x(k) - \Upsilon u(k-1), \quad (14)$$

where free response means the prediction of model outputs for the whole prediction horizon, if $\triangle U$ is always zero vector—no input moves assumed.

After this the cost function can be expressed:

$$V(k) = ||\Theta \Delta U(k) - E(k)||_Q^2 + ||\triangle U(k)||_R^2. \quad (15)$$

This can be brought to the form (refer to [9] for details):

$$V(k) = const - \triangle U(k)^T G + \triangle U(k)^T H \triangle U(k), \quad (16)$$

where $G = 2\Theta^T Q E(k)$ and $H = \Theta^T Q \Theta + R$.

To find the optimal $\triangle U(k)$, we set the gradient of $V(k)$ equal to zero, thus obtaining:

$$\nabla_{\triangle U(k)} V = -G + 2H\triangle U(k) = 0. \quad (17)$$

Then the optimal future input changes are given by:

$$\triangle U(k)_{opt} = \frac{1}{2} H^{-1} G. \quad (18)$$

Once the optimal process inputs are computed, we only take the first element, apply the control action, and send it to the DCS. Once this step is finalized, we apply the same concept in a receding horizon strategy.

As we have different process model for simulation and for the MPC design, it is not possible to guarantee zero tracking error. If we do not measure the real output and do not restore model states respectively, then an offset appears between MPC model output estimation and the real process outputs. When model output is close to the set point, real process output value can differ significantly. As models are too different, MPC finds a steady-state point where it estimates that no updates to the control law are necessary and the process output should reach the set point without any additional actions. However, this is not the case, and the process does not converge to the set point because it has a slightly different behavior than the one predicted by the model of MPC. As a result, a static control error appears.

We decided to try a simple approach to solve the problem. We augment the model with extra states that will compensate the offset in each cycle of process states and outputs estimation. This augmented model has the following form:

$$\hat{A} = \begin{pmatrix} A & \mathbf{0} \\ C & \mathbf{0} \end{pmatrix}, \hat{B} = \begin{pmatrix} B & \mathbf{0} \\ \mathbf{0} & -I \end{pmatrix}, \hat{C} = \begin{pmatrix} C & -I \end{pmatrix}, \quad (19)$$

where I is identity matrix and $\mathbf{0}$ is zero matrix of suitable dimension.

Extra states and extra inputs are added to the model. Extra input vector U_{ext} consists of measured process outputs. Extra states are calculated as follows:

$$X_{ext} = CX - U_{ext}, \quad (20)$$

where X is a vector of original states, X_{ext} is a vector of offsets between model outputs and real measured process outputs. It is calculated every execution cycle and subtracted from the original model output vector using $-I$ in matrix $\hat{C}$.

Unfortunately the same model cannot be used to calculate the optimal process input due to problems with scaling. Coefficients of augmented states (matrix C) and inputs (identity matrix) are significantly higher than coefficients of original states (matrix A) and inputs (matrix B). While making prediction MPC multiplies states with matrix $\hat{A}$ and inputs with matrices $\hat{A}$ and $\hat{B}$ many times. This leads to domination of augmented states in prediction calculation thus masking the effect of original states and inputs. To avoid this, we use another approach. For prediction, we augment the model in a different way:

$$\tilde{A} = \begin{pmatrix} A & \mathbf{0} \\ \mathbf{0} & I \end{pmatrix}, \tilde{B} = \begin{pmatrix} B & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}, \tilde{C} = \begin{pmatrix} C & -I \end{pmatrix}. \quad (21)$$

In this case offset is fixed for all prediction steps and any effect of the original input change will be correctly transferred to the model output. This model is much better for prediction than the original one, because it does not suffer from output offset, maintaining the original model behavior.

V. MPC IN SIMULATION ENVIRONMENT

After process model has been obtained, a simulation environment is developed to transfer the results to a real-life application. Simulation is built with Java function block of DCS software. A simple Java code block is created. It reads matrices A, B and C of the state-space model as well as process inputs values and calculates model states and outputs using (1).

Without loss of generality, we initialize the model with $x(0) = 0$. Model is run in DCS run-time environment in a 4-second cycle. Simulation is carried out 15 times faster than real process intentionally to make testing faster without need to wait for long process transients.

MPC application is developed in Java using Eclipse development environment in Debian Linux. It uses TCP/IP network to communicate with DCS and can be located in the same computer with the process simulation or in any other accessible through the network. DCS process simulation runs under Windows OS. After a functional version of MPC application was released, it was used on the same computer with DCS process simulation.

To make simulation environment ready for testing we need to only prepare communication with the MPC application using Modbus/TCP interface that is supported by DCS. Related configuration was prepared with DCS role in communication as slave and MPC application as master.

After MPC application has started, it shows to the user that all the parameters (Modbus/TCP settings, model, MPC settings) have been loaded. As the applications acts as Modbus master, it sends requests to the configured slave. In case of no response, the application quits informing the problem to the user. If slave (DCS process simulation) is also configured correctly and is on-line, then communication is established.

It is also important to secure the control system from communication problems or any MPC application functionality issue. For these purposes keepalives are used, binary signals that switch their values after a predefined period of time. One signal is generated by the Modbus thread every 2 seconds, another one by the MPC thread in each execution cycle. These signals are also sent through Modbus interface to DCS. The later has a timeout 5 seconds for Modbus thread signal and 10 seconds for the MPC thread signal. Time counter of each signal is reset when related signal changes its value. If timeout period is exceeded in any counter, DCS diagnoses MPC application fault, freezes manipulated input value and gives an alarm to the operator.

MPC application reads all relevant process variables listed in Table I as well as output set points. All of these variables are used to calculate current states of the model (including offset compensation in current step) and optimal process input for the next cycle.

In this case only *Input 1* is controllable. All other process inputs are measurable disturbances. MPC application calculates the optimal value for the gas flow only and sends it to DCS via Modbus interface.

DCS graphical user interface is shown in Fig. 3. Basic process layout is depicted on the display. All inputs and outputs are placed on the positions where they are measured in the process. Process inputs that are not controlled—water flow through the boiler, water flow through the plant and boiler inlet temperature—can be modified by the user directly simulating process disturbance (in reality these are modified by other parts of the process). Process outputs—boiler outlet temperature and plant outlet temperature—are calculated by the simulation model in DCS. These are shown at the bottom of the corresponding value boxes. Upper parts are relative set points that can be modified by the user.

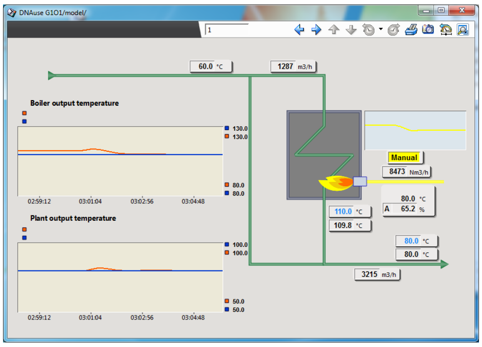


Fig. 3. DCS user interface.

Trends for the outputs and for the controllable input are added to better understand simulation results. These are updated in real time.

Process simulation can be operated in three modes. Mode 1 is manual. Operator can modify process inputs (including gas flow) and observe resulting outputs values. For mode 2 PI controller is used in DCS. Related data box is right under the gas pipe on the display. PI controller measures plant output temperature and controls gas flow to keep temperature close to set point. All other inputs can be modified by the user. The controller is made only for simulation purposes. This loop does not exist in the real process control system. PI controller was tuned using direct synthesis method [1]: $k_p = 1.24$, $t_i = 10$. In mode 3 gas flow value is calculated by the MPC application. User can also modify other inputs of the process model.

In the current work, only the plant output temperature is of interest. There is no need to control boiler output because we assume that this output is maintained within the normal range of operation (between 80 to $140°C$). In the real world, the boiler is operated in conditions where its output is in the range of 85 to $125°C$. Because we control a slow process, it is not possible that the output goes outside of the normal operating condition without the operator seeing it. If this happens, then the operator can take corrective actions. Of course, MPC including constraints would be ideal in order to take these limitations into account. Nonetheless, unconstrained MPC can also operate in these circumstances. We keep boiler output temperature in the model for later use in the future continuation of this work.

To tune the MPC strategy, we consider that the plant temperature output is the main point of interest. Weights for *Output 1* (boiler output temperature) and for *Output 2* (plant output temperature) are set to 0.1 and 350, respectively. These values are found by trial and error.

We also need to penalize input changes to prevent overreaction in the gas flow, making the whole system unstable. We find that a suitable weight for this case is 0.1.

Testing of MPC (in the simulation framework) will be performed by giving an extreme disturbance that cannot happen

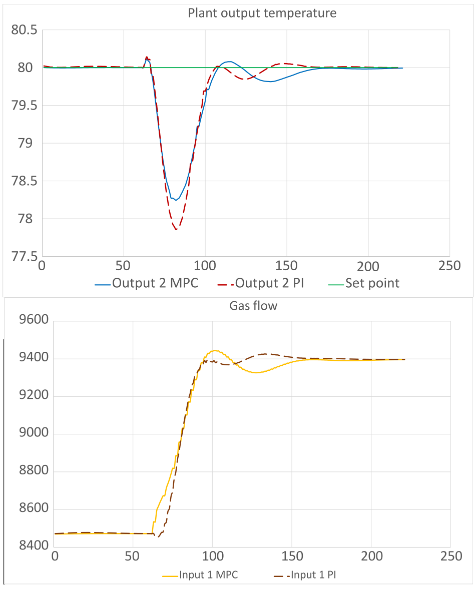


Fig. 4. Simulation results.

in the real world. By doing this, we want to assess whether the MPC application is able to stabilize the process in such extreme conditions. In case of success, there is a high chance of having a stable control in the real process. The disturbance is chosen as a instant decrease in the boiler inlet temperature by $2°C$. In real operations, this value only changes around $\pm 0.2°C/min$.

Simulation results are shown in Fig. 4. Also DCS PI-controller result is added to the trend. We can see that PI-controller behavior is similar to MPC result, but MPC shows a quicker reaction.

The main purpose of this work is to assess whether MPC is capable of making the closed loop stable, and to overcome the possible difficulties in the implementation of the MPC strategy in the real world. We also performed other tests with other large-scale disturbances. In all those cases, we obtained a stable closed-loop.

VI. Conclusion

An MPC application was developed in Java as a stand-alone software application. It was tested in a simulation environment close to real-life conditions. Real DCS run-time environment was used for this purpose. Modbus/TCP protocol was applied for communication between MPC and DCS. Similarly, this MPC application can be applied to any process controlled by any DCS or PLC, since Modbus is one of the most widely supported industrial protocols.

From simulation results we can observe that MPC is able to control system's behavior adequately. Next step will be to implement this MPC strategy to the real-life process control.

VII. Discussion

Unconstrained MPC is straightforward to implement, but it is not sufficient for controlling the process in the whole range of operation, because it does not take into account any physical limitations of the process. In some circumstances its control action can be inadequate. In the future it would be of interest to implement constrained MPC with an improved augmented model.

Also command line interface cannot be considered as a modern way to interact with the user. In a later stage, it is planned to build a web-based human-machine interface.

References

[1] B Wayne Bequette. *Process control: modeling, design, and simulation*. Prentice Hall Professional, 2003.

[2] E.F. Camacho and C.A. Bordons. *Model Predictive Control*. Springer-Verlag, London, 2004.

[3] L. del Re, F. Allgöwer, L. Glielmo, C. Guardiola, and I. Kolmanovsky, editors. *Automotive Model Predictive Control*. Springer-Verlag, London, 2010.

[4] J.W. Eaton and J.B. Rawlings. Model predictive control of chemical processes. *Chemical Engineering Science*, 47:705–720, 1992.

[5] C.E. Garcia, D.M. Prett, and M. Morari. Model predictive control: Theory and practice–a survey. *Automatica*, 25:335–348, 1989.

[6] B. I. Godoy, J. H. Braslavsky, and J. C. Aguero. A model-based feedback control strategy for heap bioleaching processes. In *Proc. 46th IEEE Conf. Decision and Control*, pages 1850–1855, December 2007.

[7] Jamod. Java modbus library. [Accessed: February, 2017]. [Online]. Available: http://jamod.sourceforge.net/.

[8] L. Ljung. *System Identification: Theory for the User*. Prentice-Hall, 2nd edition, 1999.

[9] J. Maciejowski. *Predictive control: with constraints*. Prentice Hall, 2002.

[10] G. G. Magalhaes, A. L. Sartor, A. F. Lorenzon, P. O. A. Navaux, and A. C. S. Beck. How programming languages and paradigms affect performance and energy in multithreaded applications. In *Proc. VI Brazilian Symp. Computing Systems Engineering (SBESC)*, pages 71–78, November 2016.

[11] D.Q. Mayne and H. Michalska. Receding horizon control of nonlinear systems. *IEEE Trans. on Aut. Control*, 35:814–824, 1990.

[12] K.R. Muske and J.B. Rawlings. Model predictive control of with linear models. *AIChE Journal*, 39:262–287, 1993.

[13] Krzysztof J. Ptasinski. *Efficiency of Biomass Energy*. Wiley, [s.l.], 1. aufl. edition, 2016.

[14] S Joe Qin and Thomas A Badgwell. A survey of industrial model predictive control technology. *Control engineering practice*, 11(7):733–764, 2003.

[15] Christopher V Rao, Stephen J Wright, and James B Rawlings. Application of interior-point methods to model predictive control. *Journal of optimization theory and applications*, 99(3):723–757, 1998.

[16] P.B. Sistu and B.W. Bequette. Nonlinear predictive control of uncertain processes: Application to a cstr. *AIChE Journal*, 37:1711–1723, 1991.

[17] V. Vansovits, E. Petlenkov, K. Vassiljeva, A. Tepljakov, and J. Belikov. Application of MPC to industrial water boiler control system in district heat plant. In *Proc. 13th Int. Conf. Control Automation Robotics Vision (ICARCV)*, pages 1609–1614, December 2014.

[18] V. Vansovits, A. Tepljakov, K. Vassiljeva, and E. Petlenkov. Towards an intelligent control system for district heating plants: Design and implementation of a fuzzy logic based control loop. In *Proc. IEEE 14th Int. Conf. Industrial Informatics (INDIN)*, pages 405–410, July 2016.

[19] S. Vazquez, J.I. Leon, and L.G. Franquello. Model predictive control: A review of its applications in power electronics. *IEEE Industrial Electronics Magazine*, 8:16–31, 2014.

[20] R. Vivanco and N. Pizzi. Computational performance of java and c++ in processing large biomedical datasets. In *Proc. (Cat. No.02CH37373) IEEE CCECE2002 Canadian Conf. Electrical and Computer Engineering*, volume 2, pages 691–696 vol.2, 2002.

Curriculum Vitae

1. Personal Data

Name	Vitali Vansovitš
Date and place of birth	03.12.1981, Russia
E-mail address	Vitali.Vansovich@gmail.com

2. Education

Educational institution	**Graduation year**	**Education (field of study/degree)**
Tallinn University of Technology	2005	Computer and Systems Engineering, B.Sc.
Tallinn University of Technology	2008	Computer and Systems Engineering, M.Sc.

3. Language competence/skills (fluent, average, basic skills)

Language	**Level**
Estonian	fluent
English	fluent
Russian	native

4. Professional employment

Period	**Organization**	**Position**
14.10.2004	ÅF-Automaatika OÜ	Automation Engineer
01.09.2008	Metso Automation Oy	Service Engineer
01.05.2013	Metso Automation Oy	Service Manager
01.04.2015–...	Valmet Automation Oy	Service Manager

5. Scientific work

    1. V. Vansovits, E. Petlenkov, K. Vassiljeva, and A. Guljajev, "Identification of industrial water boiler for model predictive control of district heat plant," in Proc. 13th Biennial Baltic Electronics Conf, Oct. 2012, pp. 315–318.
    2. V. Vansovits, E. Petlenkov, K. Vassiljeva, A. Tepljakov, and J. Belikov, "Application of MPC to industrial water boiler control system in district heat plant," in Proc. 13th Int. Conf. Control Automation Robotics Vision (ICARCV), Dec. 2014, pp. 1609–1614.
    3. V. Vansovits, A. Tepljakov, K. Vassiljeva, and E. Petlenkov, "Towards an intelligent control system for district heating plants: Design and implementation of a fuzzy logic based control loop," in Proc. IEEE 14th Int. Conf. Industrial Informatics (INDIN), Jul. 2016, pp. 405–410.
    4. V. Vansovits, B. I. Godoy, A. Tepljakov, K. Vassiljeva, and E. Petlenkov, "Model-based control design for a district heating plant," in Industrial Informatics (INDIN), 2017 IEEE 15th International Conference on. IEEE, 2017, pp. 615–620.

6. Defended theses

    Using "metsoDNA" techology in control courses, B.Sc., Tallinn University of Technology, 2005.

    Reducing the control loop oscillations in the heat exchanger for liquid fuel heating, M.Sc., Tallinn University of Technology, 2008.

7. Main areas of scientific work

    Model based control of industrial processes.

Elulookirjeldus

1. Isikuandmed

Ees- ja perekonnanimi	Vitali Vansovitš
Sünniaeg ja -koht	03.12.1981, Venemaa
Kodakondsus	Eesti
E-posti aadress	Vitali.Vansovich@gmail.com

2. Hariduskäik

Õppeasutus (nimetus lõpetamise ajal)	**Lõpetamise aeg**	**Haridus (eriala/kraad)**
Tallinna Tehnikaülikool	2005	Arvuti- ja süsteemitehnika, B.Sc.
Tallinna Tehnikaülikool	2008	Arvuti- ja süsteemitehnika, M.Sc.

3. Keelteoskus (alg-, kesk- või kõrgtase)

Keel	**Tase**
Eesti	kõrgtase
Inglise	kõrgtase
Vene	emakeel

4. Teenistuskäik

Töötamise aeg	**Tööandja nimetus**	**Ametikoht**
14.10.2004	ÅF-Automaatika OÜ	Automaatika Insener
01.09.2008	Metso Automation Oy	Hooldusinsener
01.05.2013	Metso Automation Oy	Hooldusjuht
01.04.2015–...	Valmet Automation Oy	Hooldusjuht

5. Teadustegevus

   Ajakirja- ja konverentsiartiklite loetelu on toodud ingliskeelse CV juures.

6. Kaitstud lõputööd

   "MetsoDNA" tehnoloogia kasutamine õppetöös, B.Sc., Tallinna Tehnikaülikool, 2005.

   Võnkumiste vältimine vedelkütuse soojusvaheti töös, M.Sc., Tallinna Tehnikaülikool, 2008.

7. Teadustöö põhisuunad

   Mudelil põhinev tööstusprotsessi juhtimine.