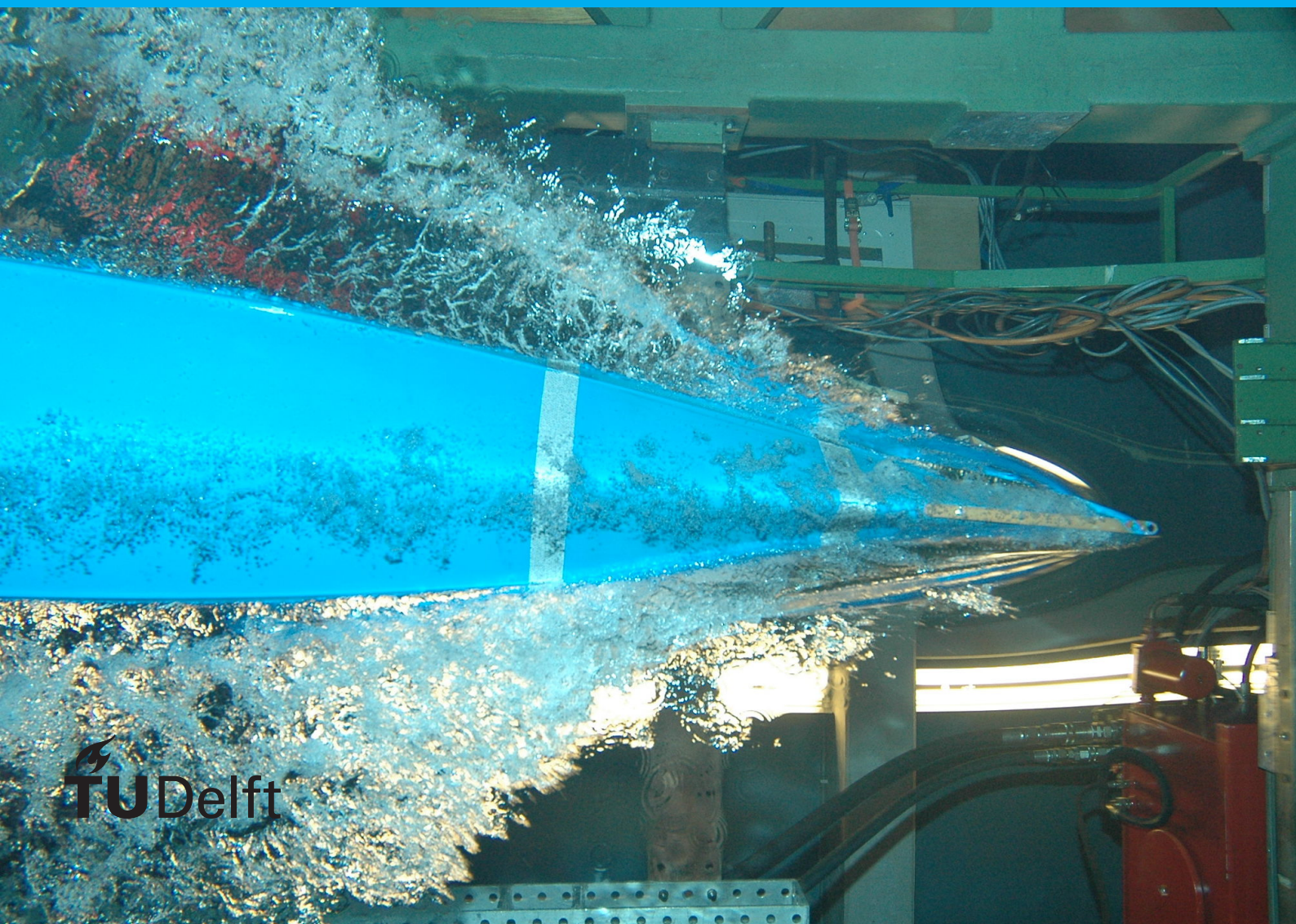
Model Predictive Control

with Application to Reticle Cooling

P. de Carvalho Monteiro



Model Predictive Control

with Application to Reticle Cooling

by

P. de Carvalho Monteiro

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday January 31, 2018 at 10:00 AM.

Student number: 4168453
Project duration: October 1, 2017 – January 31, 2019
Thesis committee:

Prof. dr. ir. N. van de Wouw,	TU Delft, supervisor
Dr. P. Mohajerin Esfahani,	TU Delft, supervisor
Prof. dr. ir. B. de Schutter,	TU Delft, extern committee member

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Contents

List of Figures

1

Introduction

1.1. Problem Description (Engineering Context)

A wafer scanner is an advanced lithography machine which prints images on a wafer at nano scale. Fig. 1.1 shows a schematic representation of the litography process. In this process, a laser creates a beam of light at a certain intensity w (W/m^2) which can be set by the customer before the start of the process. The light beam first passes through a reticle, then through the lens column (LC), and finally falls onto the wafer. The reticle is made of quartz and has a chrome layer at the bottom which contains the image to be projected on the wafer. The lens column consists of multiple passive and actively controlled lenses to reduce the reticle image by a factor of 4 and also to compensate for disturbances acting on the image. The wafer is a Silicium disc of 300 mm in diameter, on which the image is projected to create chips.

The area of interest (AOI) denotes the area underneath the reticle where the light beam passes through. During the exposure process, part of the dose intensity w is dissipated into heat within the reticle. This reticle heating has two consequences, which negatively influences the quality of the image created on the wafer, namely:

1. **Thermal Disturbances**: The most significant consequence of reticle heating is thermal disturbance of the air between the bottom of the reticle and the lens column. The environment is conditioned at a constant temperature of $22°C$, which is the desired working temperature. However, due to reticle heating, the average temperature at AOI increases, leading to temperature differences between AOI and the air underneath the reticle. This temperature differences cause a boundary layer underneath the reticle due to natural convection that affects the refractive index of light passing trough this layer. This causes unpredictable behaviour, which results in distorted images that cannot be corrected by the lens column and/or the wafer stage.

2. **Thermal Deformations**: Deformations of the reticle caused by reticle heating are called thermal deformations. When heat is absorbed on the reticle, the temperature starts to increase. Deviations in the reticle temperature with respect to a reference temperature, result in deformations of the reticle. Reticle deformation eventually leads to image errors due to the fact that the image created by the reticle will also deform with the shape of the reticle.

In conclusion, during the exposure process the consequences of reticle heating (thermal disturbances and deformations) result in undesired image aberrations created on the wafer. The next Section explains the current state of practice at ASML and the current state of the art in literature regarding the above mentioned problems.

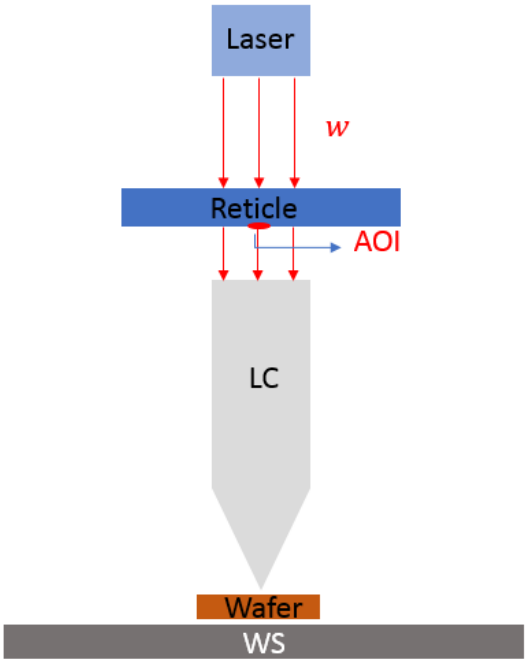


Figure 1.1: A wafer scanner.

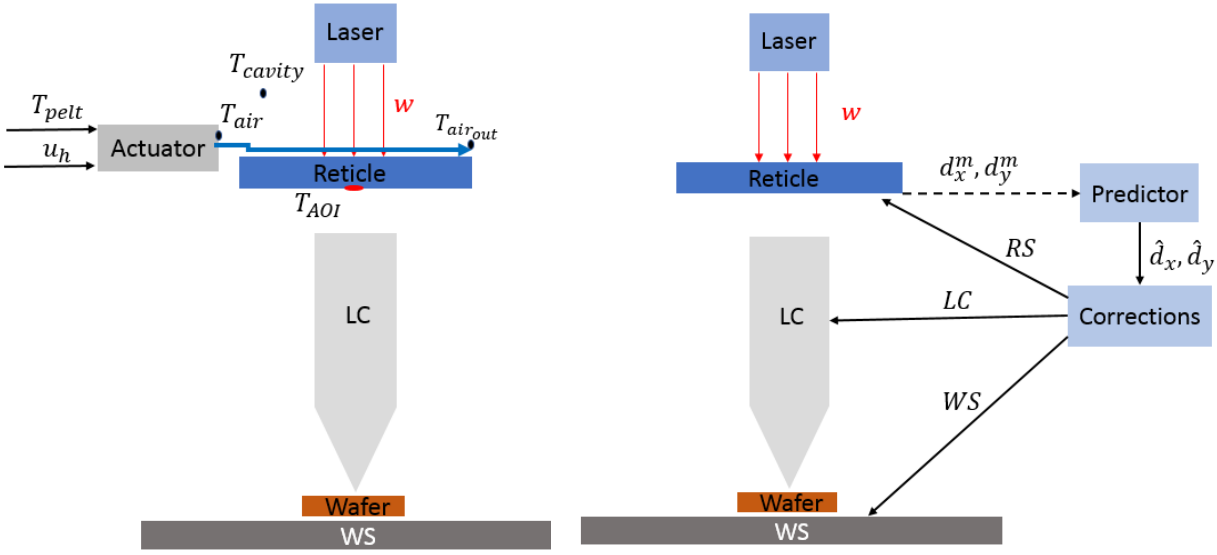
1.2. Current State of Practice

As solutions to the above-mentioned reticle heating problems, there are two current approaches installed:

1. **Open-Loop Cooling**: To reduce thermal disturbances to air due to reticle heating, wafer scanners use open-loop reticle cooling (see Figure 1.2a). The goal of this method is to blow cool air at a constant temperature $T_{air} = 21.8\ ^{\circ}C$ in order to reduce the temperature difference between AOI and the environment, which implies reducing thermal disturbances during the exposure process. In order to achieve control, actuators and sensors are needed. The actuator consists of a heating element and a nozzle. Conditioned air at $T_{pelt} = 18\ ^{\circ}C$ is heated by the heater to the desired reference temperature of $21.8\ ^{\circ}C$. The saturated input u_h to the heater is the output of the PID-controller, which reduces the error between the measurement of T_{air} and the desired reference temperature of $21.8\ ^{\circ}C$. This solution does not adapt to the change in the temperature difference between AOI and the environment, instead it only focuses on the temperature of the nozzle outflow (T_{air}), hence the reason for calling it open-loop control.

2. **Reticle Deformation Correction**: This method (see Figure 1.2b) involves the following steps:

   - At the start of every wafer exposure, the deformations (d_x^m, d_y^m) are being measured at 16 locations on the bottom surface of the reticle.
   - Since the deformation measurement is not available during exposure of a wafer, a predictor uses the measured deformation (d_x^m, d_y^m) to predict the deformation $(\hat{d}_x, \hat{d}_y)$ during the time interval of the wafer exposure. The predicted $\hat{d}_x, \hat{d}_y$ represent deformations that are correctable by the Reticle Stage (RS), the Wafer Stage (WS) and the Lens Column (LC).



(a) Open-Loop Cooling method. (b) Reticle Deformation Correction method.

Figure 1.2: The current state of practice.

1.3. Problem Formulation and Current State of the Art

Instead of designing a controller which keeps the nozzle outflow temperature T_{air} constant, we aim to investigate the design of a controller which adapts to the change in temperature difference between AOI and the environment. The goal of this controller is to reduce the effect of thermal disturbances with respect to the currently used Open-Loop Cooling approach, which implies reducing the temperature difference. However, when the flow temperature changes, it also affects the deformation of the reticle. In a worst case scenario, a controller could introduce reticle deformation shapes that are not correctable by the Reticle Deformation Correction method. It is thus very important that the feedback controller does not introduce non-correctable deformations. This is achieved by constraining the non-correctable deformations such that it does not worsen with respect to Open-Loop Control. The controller should be able to handle input/output constraints, since Section 2 will show the non-correctable deformations to be an output of the reticle model. Incorporation of the sensor output T_{cavity} (see Figure 1.2a) is also a requirement for the controller due to the inability to measure the reticle states directly since placing sensors on the reticle would lead to inevitable image errors.

A controller which satisfies the above criteria is found in Model Predictive Control (MPC). MPC distincts itself from other controller methods such as PID, LQ, and H_∞ due to the fact that it can systematically cope with time domain input/output constraints. There are several publications on MPC available in literature that provide a good introduction to the theory. Rawlings [1] provides a tutorial overview of the MPC aimed at control practitioners. Also, Qin and Badgwell [2] provide an overview of the general MPC control algorithm along with an overview of the available MPC technology in industry. For a more detailed description of the MPC algorithm, Van Den Boom and Backs [3] provided lecture notes describing the basics of MPC, stability, and more. It is however a well known fact that the MPC algorithm struggles with hard output/state constraints, which is the reason why Zafiriou and Marchal [4] provided an overview for why instability may occur for SISO systems when implementing MPC (with hard output/state constraints) in a closed-loop manner. Oliveira and Biegler [5] showed that the presence of hard output/state constraints can introduce extra feedback terms in the predictive controller, which can lead to instability/infeasibility of the constraint closed-loop system. The presence of hard input and output constraint constraint results in an online optimization that produces a nonlinear controller even when the plant and model are assumed linear (Zafiriou [6]). In all these papers, the use of soft output/state constraints

is recommended. For MPC problems with non-minimum phase plants there are also papers describing how to choose the soft-constraint weights in order to improve closed-loop behaviour [7, 8]. In order to implement MPC in practice, it is necessary to incorporate output feedback. Similar to the LQG-control case, it is possible to separately design a linear state observer, which estimates the current state of the plant for use in the MPC algorithm (Rawlings and Mayne [9]).

1.4. Contribution

From the perspective of wafer scanners, this project, to the best of our knowledge, for the first time develops an optimization-based framework to account for thermal disturbances and the non-correctable deformations for an MPC controller equipped with partial temperature sensors located away from the *AOI*. A particular contribution of this project is to investigate possible existing tuning guidelines in order to choose the optimal control and prediction horizon for the MPC-controller in the context of the reticle cooling application. Deriving (sub) models to be used for MPC using system identification and model reduction techniques, were also one of the challenges in this project. The process model, as will be introduced in Section 2, is also a contribution of this project.

1.5. Outline

Section 2 discusses the mathematical derivation of the actuator, the reticle and the sensor thermal dynamics. Section 3 uses the derived model to design the MPC using full state information along with an overview of the tuning guidelines. However, since full state information is not realistic, Section 4 discusses the incorporation of the output feedback using the available sensor.

2

Thermal Process Modeling

This Section discusses the derivation of the process model (see Figure 2.1), which consists of the following components, related to system components in Figure 1.2a:

- a sub-model P_A of the actuator having inputs $u_h(t)$ and T_{pelt} and output $T_{air}(t)$. The actuator consists of a heater and a nozzle where conditioned air at a certain temperature T_{pelt} is heated by the heater, with heating power u_h, to the temperature $T_{air}(t)$ out of the nozzle.

- a sub-model P_R of the reticle with inputs $w(t)$ and $T_{air}(t)$ and outputs $\epsilon_x(t)$, $\epsilon_y(t)$, $z(t)$, and $T_{air_{out}}$. The dose with intensity $w(t)$ falls onto the reticle causing the non-correctable deformations $\epsilon_x(t)$, $\epsilon_y(t)$, the temperature difference $z(t)$, and the airflow temperature $T_{air_{out}}$ to increase.

- a sub-model P_S of the sensor which relates the flow temperature $T_{air_{out}}(t)$ to the sensor output $y(t) = T_{cavity}(t)$.

The three sub-models are derived in Section 2.1, 2.2, and 2.3, respectively. Section 2.4 shows how the dose disturbance $w(t)$ is modeled and Section 2.5 shows how a series interconnection of the three sub-models results in the total process model P that will be used throughout this report.

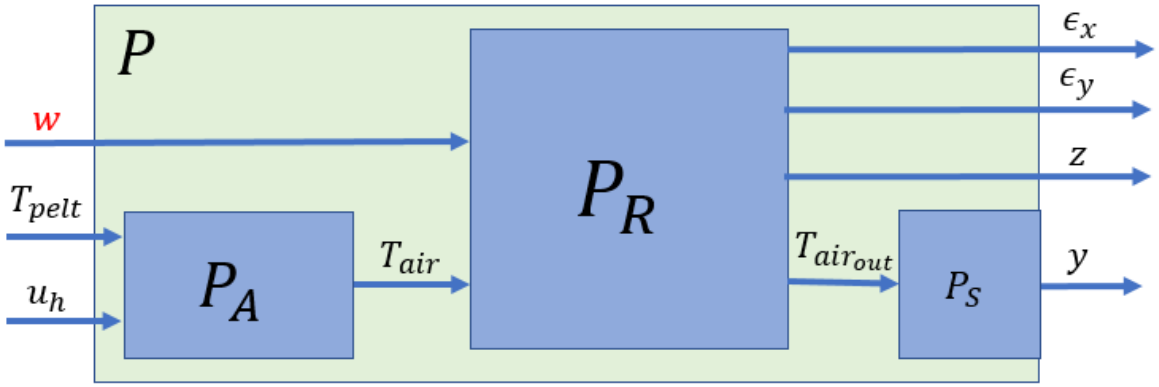


Figure 2.1: Blok diagram of the process model P with the corresponding sub plants.

2.1. Actuator Dynamics

As mentioned earlier, conditioned air flows out of the nozzle into the reticle stage at a certain temperature $T_{air}(t)$, for which a peltier element is used to deliver cool air at a constant temperature T_{pelt}. Thereafter, it is heated by a heating element with power $u_h(t)$. The actuator dynamics gives a relation between the inputs T_{pelt}, $u_h(s)$ and the output $T_{air}(s)$ in the Laplace-domain ($s \in \mathbb{C}$) that can be written as:

$$T_{air}(s) = \underbrace{\begin{bmatrix} 1 & P_{u_h 2 T_{air}}(s) \end{bmatrix}}_{P_A(s)} \begin{bmatrix} T_{pelt} \\ u_h(s) \end{bmatrix}, \tag{2.1}$$

where $P_{u_h,T_{air}}(s)$ denotes the dynamics from the heating element $u_h(s)$ to the inflow temperature $T_{air}(s)$. $P_{u_h,T_{air}}(s)$ can be written as a third-order transfer function

$$P_{u_h,T_{air}}(s) = \frac{T_{air}(s)}{u_h(s)} = \frac{K}{as^3 + bs^2 + cs + 1}, \tag{2.2}$$

where the parameters $K = 88.81\ mK/W$, $a = 3.1162 \cdot 10^4\ s^3$, $b = 3.244 \cdot 10^3\ s^2$, and $c = 175.69\ s$ are estimated using measured step response data to a heater input of $u_h = 40\ W$. Figure 2.2 shows the step response of the derived model and the measurement data, where the maximum absolute error equals 0.035 K, which is sufficiently small. For control design purposes, the actuator dynamics (2.1) are written in state-space form as follows:

$$P_A \begin{cases} \dot{x}_A(t) = A_A x_A(t) + B_A \begin{bmatrix} T_{pelt} \\ u_h(t) \end{bmatrix} \\ T_{air}(t) = C_A \cdot x_A(t) + D_A \begin{bmatrix} T_{pelt} \\ u_h(t) \end{bmatrix}, \end{cases} \tag{2.3}$$

where $x_A \in \mathbb{R}^{N_A \times 1}$, $A_A \in \mathbb{R}^{N_A \times N_A}$, $B_A \in \mathbb{R}^{N_A \times 2}$, $C_A \in \mathbb{R}^{1 \times N_A}$, $D_A \in \mathbb{R}^{1 \times 2}$, $T_{air} \in \mathbb{R}$ and $N_A = 3$. The matrix A_A is Hurwitz with the eigenvalues $\lambda = -0.0489 \pm 0.0512i$ and $\lambda = -0.0064$. The matrices of the actuator model can be found in Appendix A.

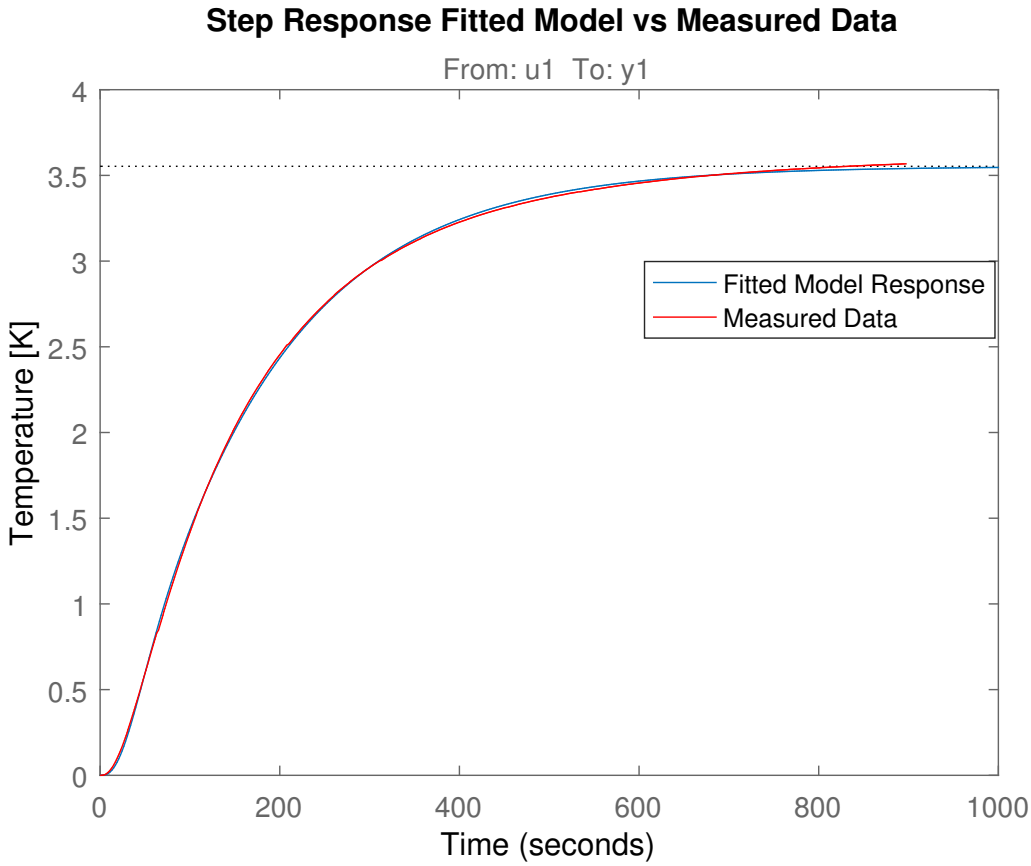


Figure 2.2: Fitted model response vs measured data.

2.2. Reticle Dynamics

After the air exits the nozzle, it flows over the reticle in the reticle stage. The flow above the reticle influences the temperature of the reticle and vice versa. The model is derived using a FEM discretization of the heat equation, which resulted in an LTI system with 2529 elements/ states and 34 outputs [11]. The first 32 outputs represent the constraint variables for MPC, which are the non-correctable deformations ϵ_x, ϵ_y. The non-correctable deformation is denoted by

$$\epsilon_x(t) = d_x^m(t) - \hat{d}_x(t) \tag{2.4}$$

$$\epsilon_y(t) = d_y^m(t) - \hat{d}_y(t), \tag{2.5}$$

where $(d_x^m, d_y^m) \in \mathbb{R}^{16}$ represent the measured deformation at 16 locations at the bottom of the reticle in the x and y direction, respectively. Assuming the reticle model to be accurate, (d_x^m, d_y^m) becomes a linear combination of the states of the reticle model. The correctable deformations $(\hat{d}_x, \hat{d}_y) \in \mathbb{R}^{16}$ are derived using polynomials representing the basic correctable reticle shapes shown in Figure 2.3a and 2.3b. The correctable deformations are derived as

follows:

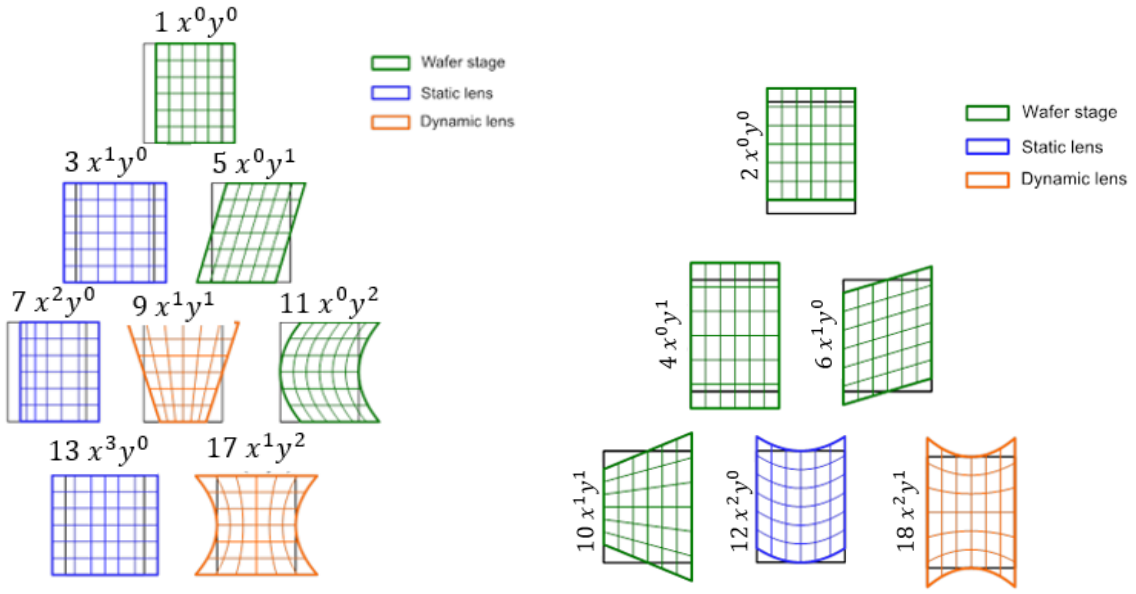
$$\hat{d}_x(t) = \underbrace{\begin{bmatrix} f_x(x_1, y_1) \\ f_x(x_2, y_2) \\ \vdots \\ f_x(x_{16}, y_{16}) \end{bmatrix}}_{F_x} \cdot k_x(t)$$
$$= F_x \cdot k_x(t) \tag{2.6}$$

$$\hat{d}_y(t) = \underbrace{\begin{bmatrix} f_y(x_1, y_1) \\ f_y(x_2, y_2) \\ \vdots \\ f_y(x_{16}, y_{16}) \end{bmatrix}}_{F_y} \cdot k_y(t)$$
$$= F_y \cdot k_y(t) \tag{2.7}$$

where $f_x(x_i, y_i) \in \mathbb{R}^{8\times1}$ contains the 8 polynomials describing the mode shapes schematically depicted in Figure 2.3a and $f_y(x_i, y_i) \in \mathbb{R}^{6\times1}$ contains the 6 polynomials of Figure 2.3b evaluated at the positions (x_i, y_i) on the bottom of the reticle. The terms $k_x \in \mathbb{R}^{8\times1}$ and $k_y \in \mathbb{R}^{6\times1}$ contains the coefficients belonging to the specified polynomials in f_x and f_y. The coefficients (k-parameters) are derived based on a linear least-squares fit between the correctable deformations (2.6), (2.7) and the measured deformations d_x^m, d_y^m. The formula for the k-parameters becomes

$$k_x(t) = (F_x^T F_x)^{-1} F_x^T \cdot d_x^m(t) \tag{2.8}$$
$$k_y(t) = (F_y^T F_y)^{-1} F_y^T \cdot d_y^m(t) \tag{2.9}$$

Since the measured deformations d_x^m, d_y^m are linear combinations of the states of the reticle model P_R, this implies that k_x, k_y are also linear combinations of the states (see (2.8) and (2.9)). In turn this makes the correctables $\hat{d}_x, \hat{d}_y$ a linear combination of the reticle states (see (2.6) and (2.7)), which shows that the non correctable deformations ϵ_x, ϵ_y are outputs of the reticle model with linear relations to the states.



(a) Correctable mode shapes for $\hat{d}_x$.

(b) Correctable mode shapes for $\hat{d}_y$.

Figure 2.3: Correctable mode shapes for the reticle's x and y direction.

The performance output z is the term that has to be minimized by the MPC-controller. As discussed in Section 1, the goal of the controller is to reduce thermal disturbances which implies reducing the temperature difference between the reticle bottom and the environment. The performance variable z thus becomes

$$z(t) = T_{AOI}(t) - T_{env} \tag{2.10}$$

where T_{AOI} represents the average temperature at the reticle's area of interest AOI (see Figure 1.1) and T_{env} is the temperature of the environment underneath the reticle. The performance z is also a linear combination of the reticle state.

The output $T_{air_{out}}$ denotes the average temperature of the airflow out of the nozzle after passing over the reticle (see Figure 1.2a). This temperature is related to the sensor output y via the sensor dynamics P_S, which is explained in Section 2.3

For control design purposes, the FEM model is reduced using the Balanced Residualization technique, which eliminates states that have lower Hankel singular values than that of the specified model order [11]. The remaining states are altered in order to preserve the DC-gain. The model order is chosen such that the resulting state-space model (2.11) captures most of the dynamics excited by the disturbance with a fundamental frequency of $f_w = \frac{1}{T_w} = \frac{1}{17}$ Hz, where T_w is the fundamental period of the dose $w(t)$ as explained in Section 2.4.

The resulting model has as inputs the dose $w(t)$ and the nozzle outflow temperature $T_{air}(t)$. The state-space realization of the reduced reticle model is given by

$$P_R \begin{cases} \dot{x}_R(t) = A_R x_R(t) + B_R \begin{bmatrix} w(t) \\ T_{air}(t) \end{bmatrix} \\ \begin{bmatrix} \epsilon_x(t) \\ \epsilon_y(t) \\ z(t) \\ T_{air_{out}}(t) \end{bmatrix} = \begin{bmatrix} C^R_{\epsilon_x} \\ C^R_{\epsilon_y} \\ C^R_z \\ C^R_{T_{air_{out}}} \end{bmatrix} x_R(t) + \begin{bmatrix} D^R_{\epsilon_x} \\ D^R_{\epsilon_y} \\ D^R_z \\ D^R_{T_{air_{out}}} \end{bmatrix} \begin{bmatrix} w(t) \\ T_{air}(t) \end{bmatrix} \end{cases}, \tag{2.11}$$

where $x_R \in \mathbb{R}^{N_R \times 1}$, $w \in \mathbb{R}$, $(\epsilon_x, \epsilon_y) \in \mathbb{R}^{16\times 1}$, $(z, T_{air_{out}}) \in \mathbb{R}$ and $N_R = 25$.

2.3. Sensor Dynamics

In order to have a complete model for control design, it is necessary to incorporate a sensor which measures the state of the reticle in real-time. The difficulty for both modeling and control lies in the fact that it is not possible to mount a sensor directly on the reticle instead, the location of the sensor $y(t) = T_{cavity}(t)$ is chosen just above the reticle. The derived sensor dynamics $P_S(s)$ should provide a relation between the average air temperature after passing the reticle $T_{air_{out}}(t)$ and the sensor output $y(t) = T_{cavity}(t)$. The sensor dynamics are derived using a second-order transfer function P_{cavity}, which was derived using measurement data available. $P_{cavity}(s)$ gives the transfer function from the inflow temperature $T_{air}(s)$ to the measured sensor output $T_{cavity}(s)$ (see Figure 2.4 and 2.5). The red line in Figure 2.6 gives the response of $T_{cavity}(t)$ (measured) to a step input of $T_{air} = -1\ ^\circ C$.

Figure 2.4: Transfer from T_{air} to T_{cavity} derived from measurements.

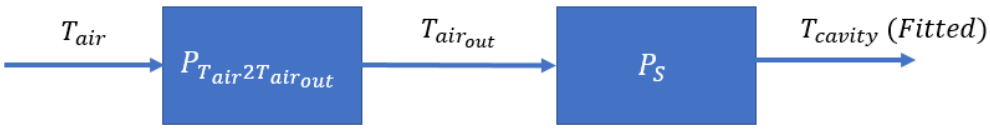


Figure 2.5: Transfer from T_{air} to T_{cavity} based on a series interconnection.

The blue line in Figure 2.6 gives the response of $T_{air_{out}}(t)$ to the same step input of $T_{air} = -1$ $^{\circ}C$. As expected, the response shows that there is indeed missing dynamics between the measured sensor output $T_{cavity}(t)$ and $T_{air_{out}}$. However, by using knowledge of the acquired time domain data $T_{air_{out}}(t)$ and $T_{cavity}(t)$ (measured) and the assumption that the following relation holds:

$$T_{cavity}(s) = P_S(s) \cdot T_{air_{out}}(s), \tag{2.12}$$

it is possible to determine the sensor dynamics P_S by using system identification techniques. The resulting transfer function structure is derived as follows

$$P_S(s) = \frac{K_1}{\tau_1 s + 1} + \frac{K_2}{\tau_2 s + 1}, \tag{2.13}$$

with the dimensionless parameters $K_1 = 2.03$, $K_2 = -1.17$ and the time constants being $\tau_1 = 27.03\ s$ and $\tau_2 = 48.08\ s$.

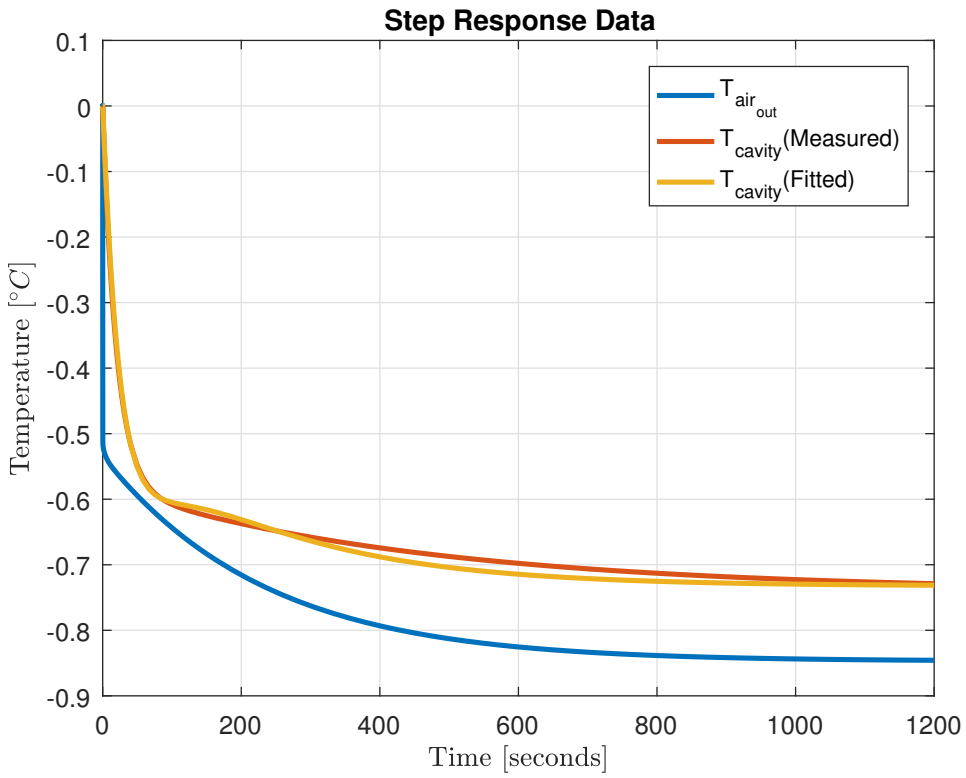


Figure 2.6: Response of both H_{cavity} and H_{FEM} to a step of $T_{air} = -1$ °C.

The yellow line of Figure 2.6 shows the responses of $T_{cavity}(t)$ due to a step input of $T_{air} = -1\ ^{\circ}C$ using the series interconnection between $P_{T_{air},T_{air_{out}}}$ and the fitted sensor dynamics P_S. As desired, the result of the fitted sensor output gives a reasonable representation of the measured sensor output ($T_{cavity}(t)$), which means that the sensor dynamics P_S can be used as the missing dynamics between $T_{air_{out}}$ and T_{cavity}. For control design purposes, the sensor dynamics will be represented by the state-space representation

$$P_S \begin{cases} \dot{x}_S(t) = A_S x_S(t) + B_S T_{air_{out}}(t) \\ y(t) = C_S \cdot x_S(t) + D_S T_{air_{out}}(t) \end{cases} \tag{2.14}$$

where $x_S \in \mathbb{R}^{N_S \times 1}$, $A_S \in \mathbb{R}^{N_S \times N_S}$, $B_S \in \mathbb{R}^{N_S \times 1}$, $C_S \in \mathbb{R}^{1 \times N_S}$, $D_S \in \mathbb{R}^{1 \times 1}$, and $y = T_{cavity}$ where $N_S = 2$. A_S is Hurwitz with the eigenvalues being $\lambda = -0.0370$ and $\lambda = -0.0208$. The sensor dynamics has a stable zero at $z = -0.0131$. The matrices of the sensor model can be found in Appendix B.

2.4. Dose Modeling

During wafer exposure, the reticle starts to heat up due to the laser dose $w(t)$ acting on the reticle. This dose, which represents the input disturbance in the model, is what causes the undesired nonzero performance variable z (see (2.10)). For this reason, this Section discusses the modeling of the dose/disturbance along with the values of the corresponding parameters. These parameters are derived based on average values during exposure of ± 70 wafers. The dose $w(t)$ is modeled as a pulse-train (see Figure 2.7), where $\bar{w} = 128.8\ W/m^2$ and $T_{\bar{w}} = 14$ seconds denote the exposure intensity and exposure duration for one wafer respectively. After exposure of a wafer, there is an average waiting time of three seconds for the next wafer in which there is no exposure ($w(t) = 0$). When the next wafer is in place, the exposure starts again resulting in a pulse train period of $T_w = 17$ seconds. The time T_{w_0} at which the exposure starts is set at 50 seconds. The model of the dose will be used throughout this

report to represent the real disturbance on the system. Note that this representation is a mere approximation of a much more complicated disturbance behaviour as seen in practice. Section 4 however, will consider an uncertainty on the intensity $\bar{w}$ in order to investigate the performance loss of MPC in a Kalman-Filter framework.

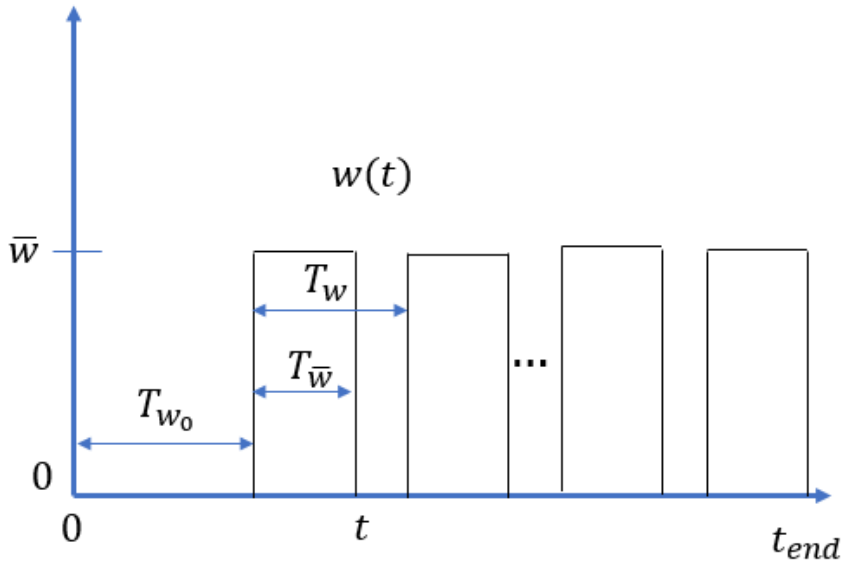


Figure 2.7: Model of the dose $w(t)$.

2.5. Total Thermal Model

By making a series interconnection of all the sub plants P_A, P_R, and P_S according to Figure 2.1, will result in the following total plant P given by

$$P\begin{cases} \dot{x}(t) = Ax(t) + B_u u_h(t) + B_d \begin{bmatrix} T_{pelt} \\ w(t) \end{bmatrix} \\ \begin{bmatrix} \epsilon_x(t) \\ \epsilon_y(t) \\ z(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} C_{\epsilon_x} \\ C_{\epsilon_y} \\ C_z \\ C_y \end{bmatrix} x(t) + \begin{bmatrix} D^u_{\epsilon_x} \\ D^u_{\epsilon_y} \\ D^u_z \\ D^u_y \end{bmatrix} u_h(t) + \begin{bmatrix} D^d_{\epsilon_x} \\ D^d_{\epsilon_y} \\ D^d_z \\ D^d_y \end{bmatrix} \begin{bmatrix} T_{pelt} \\ w(t) \end{bmatrix} \end{cases} \tag{2.15}$$

where $x = \begin{bmatrix} x_A \\ x_R \\ x_S \end{bmatrix} \in \mathbb{R}^{N_t}$, $A \in \mathbb{R}^{N_t \times N_t}$, $B_u \in \mathbb{R}^{N_t \times 1}$, $B_d \in \mathbb{R}^{N_t \times 2}$, $N_t = N_A + N_R + N_S$, and the matrices is given by Appendix C.The total matrix A is Hurwitz with the fastest eigenvalue being $\lambda = -200$ and the slowest eigenvalue being $\lambda = -0.00487$. The system contains right-half plain (RHP) zeros, which results in a non-minimum phase system. The RHP-zeros stem from the reticle dynamics derived in Section 2.2.

3

Model Predictive Control with Full State Feedback

This Section discusses the mathematical representation of the control problem in the framework that is solvable by using a Model predictive Controller. In Section 3.1, the original problem is formulated using hard state constraints, but the implementation showed that the solver was having infeasibility problems. For this reason Section 3.2 introduces soft state constraints where the state constraints are violated whenever the problem becomes infeasible. Section 3.3 ends with a guideline on how to choose the control and prediction horizon based on some defined performance variables.

3.1. Control Problem Formulation with Hard Constraints

Before formulating the control problem, the plant P given by (2.15) is discretized using a sampling time of $T_s = 1$ seconds. The resulting discrete-time system model becomes

$$P\begin{cases} x(k+1) = \Phi x(k) + \Gamma_u u_h(k) + \Gamma_d \begin{bmatrix} T_{pelt} \\ w(k) \end{bmatrix} \\ \begin{bmatrix} \epsilon_x(k) \\ \epsilon_y(k) \\ z(k) \\ y(k) \end{bmatrix} = \begin{bmatrix} C_{\epsilon_x} \\ C_{\epsilon_y} \\ C_z \\ C_y \end{bmatrix} x(k) + \begin{bmatrix} D^u_{\epsilon_x} \\ D^u_{\epsilon_y} \\ D^u_z \\ D^u_y \end{bmatrix} u_h(k) + \begin{bmatrix} D^d_{\epsilon_x} \\ D^d_{\epsilon_y} \\ D^d_z \\ D^d_y \end{bmatrix} \begin{bmatrix} T_{pelt} \\ w(k) \end{bmatrix} \end{cases} \tag{3.1}$$

with $\Phi = e^{AT_s}$, $\Gamma_u = \int_0^{T_s} e^{A\tau} d\tau B_u$ and $\Gamma_d = \int_0^{T_s} e^{A\tau} d\tau B_d$. The control problem is written such that an objective function, related to minimizing the performance variable z, is minimized subject to input- and state constraints at every time step k according to

$$\min_{u_h(\cdot)} \sum_{j=1}^{N} |z(k+j)| \tag{3.2}$$

subject to the hard state constraints

$$\begin{aligned} \left\|\epsilon_x(k+j)\right\|_1 &\leq \delta_1 \quad \forall j = 1, \cdots, N, \\ \left\|\epsilon_y(k+j)\right\|_1 &\leq \delta_2 \quad \forall j = 1, \cdots, N, \end{aligned} \tag{3.3}$$

input constraints

$$u_{min} \leq u_h(k+j) \leq u_{max} \quad \forall j = 0, \cdots, N_c - 1, \tag{3.4}$$

and subjected to model constraints given by (3.1) where the following variables are known

$$\begin{aligned} &w(k+j), && \forall j = 1, \cdots, N \\ &T_{pelt}(k+j) = c, && \forall j = 1, \cdots N \\ &x(k), && \forall k \end{aligned} \tag{3.5}$$

The objective function (3.2) is aimed at minimizing the absolute value of the difference between the future reticle temperature at the area of interest $T_{AOI}(k+j)$ and the constant reference environment temperature T_{env} denoted by $|z(k+j)| = |T_{AOI}(k+j) - T_{env}|$ within the prediction horizon window N. In minimizing the objective function it is nescecarry that all future constraint variables $||\epsilon_x(k+j)||_1$ and $||\epsilon_y(k+j)||_1$ satisfy the constraints δ_1 and δ_2 respectively. These constraints are derived based on the maximum allowable non-correctable deformations of the current state of practice. The result of the optimization is a sequence of control commands $u_h(\cdot) = [u_h(k), u_h(k+1), \dots, u_h(k+N_c-1), \dots, u_h(k+N)]$, where N_c is the control horizon. The control input is free to move for the first N_c moves and is kept constant after that according to the following constraint

$$u_h(k+j) = u_h(k+N_c-1) \quad \forall j \geq N_c \tag{3.6}$$

Introduction of a control horizon is usually beneficial for the computation time of the optimization problem, while still allowing the prediction horizon to be sufficiently large.

MPC is implemented using the receding horizon principle where after computing the optimal control sequence $u_h(\cdot)$, only the first control sample $u_h(k)$ is implemented. For the implementation of the MPC in this report, the assumption (3.5) is made that future disturbances of the dose $w(k+j)$ is known with the values as discussed in Section 2.4. Furthermore, the following parameters are used in order to completely define the MPC-algorithm in this Section:

$$T_{pelt} = -4\ ^\circ C \tag{3.7}$$
$$\delta_1 = 1.051\ nm \tag{3.8}$$
$$\delta_2 = 946\ nm \tag{3.9}$$
$$u_{min} = 0\ W \tag{3.10}$$
$$u_{max} = 45\ W. \tag{3.11}$$

The above mentioned parameters are used for all the simulations throughout this report. The state $x(k)$ is also known for every time step k, where the initial condition $x(0)$ is always chosen such that the overall reticle stage temperature starts at 22 $^\circ C$. However, during the closed-loop simulations with MPC, infeasibility problems occurred due to the fact that at a certain point in time the solver was not able to solve the optimization problem given by (3.2)-(3.6). The only explanation is that the solver could not find an input sequence $u_h(\cdot)$, which satisfied the constraints in (3.3) and (3.4). In order to find out which constraint is violated, Figure 3.1 shows a result of the three constrained signals $u_h(k)$, $\|\epsilon_x(k)\|_1$, $\|\epsilon_y(k)\|_1$ along with their constraints u_{min}, u_{max}, δ_1, and δ_2, respectively in case of when $N = 30$ and $N_c = 1$. This specific simulation is terminated at 253 seconds due to infeasibility, which is also the time at which $\|\epsilon_x(k)\|_1$ reaches the constraint δ_1. The rest of the signals $\|\epsilon_y(k)\|_1$ and $u_h(k)$ do satisfy their constraints. Figure 3.1 thus suggests that infeasibility is caused by the hard state constraint $\|\epsilon_x(k)\|_1$. Infeasibility due to state constraints is a commonly recognized problem when using MPC [4], [5], [6]. An approach to avoid of this infeasibility is by using soft state constraints, which is discussed in the next Section.

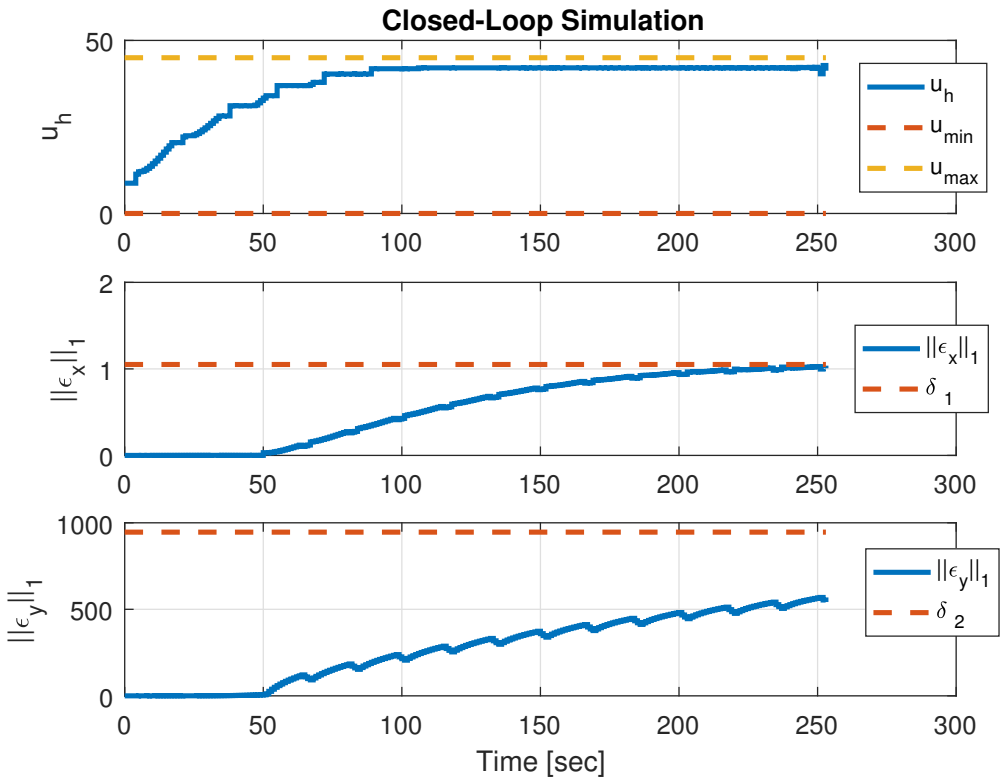


Figure 3.1: Closed-loop simulation results of the three constraint signals until infeasibility occurs in case of $N = 30$ and $N_c = 1$.

3.2. Control Problem Formulation with Soft Constraints

Section 3.1 showed that implementation of hard state constraints can cause infeasibility of the optimization problem. This is a common problem in MPC, which is usually solved by relaxing the state constraints with a slack variable α. This variable equals the maximum constraint violation within the prediction horizon window N. The slack is added in the cost-function in order to keep the term as small as possible. The objective function and the state constraint given by (3.2) and (3.3) are modified as

$$\min_{u_h(\cdot),\alpha} \sum_{j=1}^{N} |z(k+j)| + p \cdot \alpha, \tag{3.12}$$

subjected to the soft state constraints given by

$$\begin{aligned} \left\|\epsilon_x(k+j)\right\|_1 &\leq \delta_1 + W_x \cdot \alpha \quad \forall j = 1, \dots, N, \\ \left\|\epsilon_y(k+j)\right\|_1 &\leq \delta_2 + W_y \cdot \alpha \quad \forall j = 1, \dots, N, \end{aligned} \tag{3.13}$$

and the nonzero slack variable

$$\alpha \geq 0, \tag{3.14}$$

where p is a constant penalty term, which adds importance to minimizing the maximum constraint violation α in the objective function (3.12). The weights W_x and W_y represent the concern for relaxing the corresponding state constraints. The larger W_x and W_y, the softer the state constraints. The input constraints denoted by (3.4), the model constraints given by (2.3) along with the assumptions of (3.5), and the control horizon constraints remains unchanged in the new framework.

For the implementation of the MPC with soft constraints, the same disturbances parameters as in the previous Section are used to define the MPC-algorithm. The algorithm is now

expanded with the following parameters:

$$W_x = 1, W_y = 1, p = 500 \tag{3.15}$$

Figure 3.2 shows the result of the three constraint signals in closed-loop in case of a soft-constraint implementation. The value of N and N_c used in this figure is the same as in Figure 3.1. The difference between the two is that, it is now possible for MPC to find feasible solutions within the specified simulation time. Figure 3.2 also shows that MPC now allows $\epsilon_x(t)$ to violate the constraint δ_1 by 43 %. However, a constraint violation of 43 % is quite high and thus different combinations of N and N_c have to be considered in order to reduce this violation. Since there are infinite possible combinations of choosing N and N_c, Section 3.3 gives tuning guidelines for choosing a combination which satisfies i.e. the desired constraint violation.

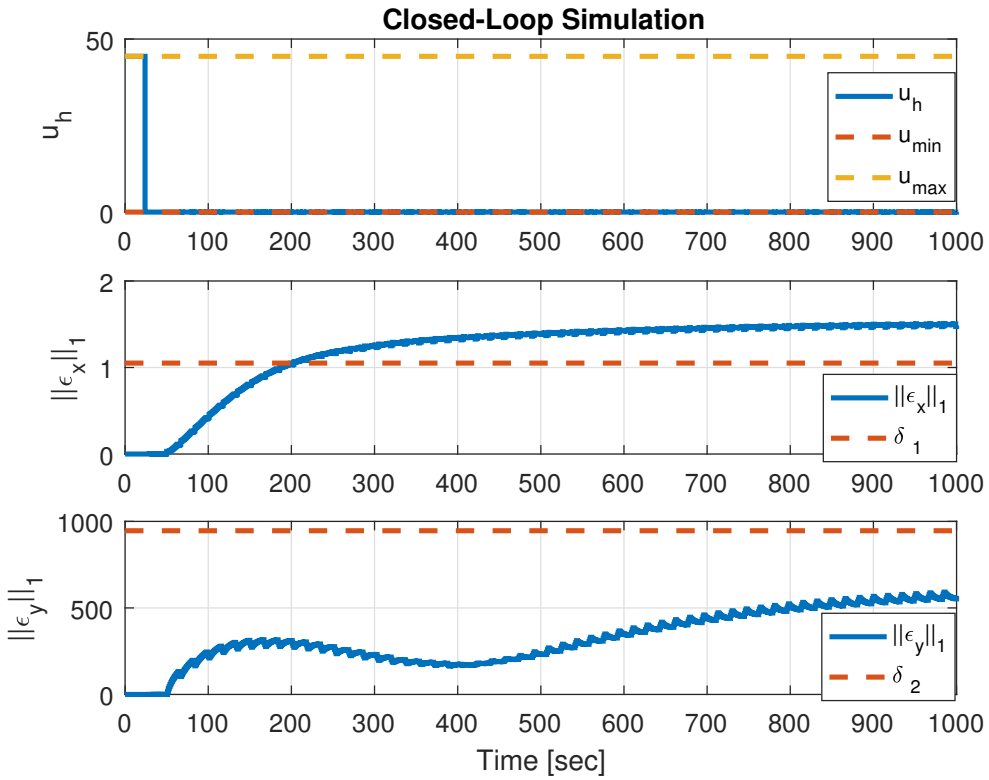


Figure 3.2: Closed-loop simulation results of the three constraint signals in case of MPC with soft constraints for $N = 30$ and $N_c = 1$.

3.3. Tuning Guidelines

Before comparing the result of MPC with the current state of practice, it is important to investigate which combination of N and N_c gives best performance. The closed-loop performance outputs that will be used to determine the optimal prediction $N*$ and- control horizon N_c* are calculated for each combination of N and N_c according to

$$z_{cl} = \frac{1}{N_{cl}} \sum_{j=1}^{N_{cl}} |z(k+j)|, \tag{3.16}$$

$$\alpha_{cl} = \frac{\max\left(\left\|\epsilon_x(k+1)\right\|_1 - \delta_1, \cdots, \left\|\epsilon_x(k+N_{cl})\right\|_1 - \delta_1\right)}{\delta_1} \cdot 100, \tag{3.17}$$

where z_{cl} denotes the average temperature deviation in K (or $^\circ C$), $N_{cl} = 1000$ denotes the amount of closed-loop iterations and α_{cl} denotes the maximum constraint violation within

the closed-loop window in %. Another important closed-loop performance measure is the computation time for every iteration, or $Comp$, which basically denotes the complexity of the optimization problem for every iteration. Note that the computation time could depend on the type of optimization solver and the type of processor. The solver used in this project is *Sedumi* and the processor is *Intel Xeon CPU E3 1505M v5*. The calculated performances for different combinations of N and N_c are given by Figure 3.3, where N varies between 1 and 600 and for each combination $N_c \leq N$. The figure illustrates 6 different cases for the control horizon, namely $N_c = N$ (maximum control horizon), $N_c = 1$ (minimum control horizon), $N_c = 3$, $N_c = 5$, $N_c = 10$, and $N_c = 20$. When increasing the control horizon from $N_c = 1$ to $N_c = 20$, it is evident that the objectives z_{cl} and α_{cl} for these settings start to resemble the objective for the setting of $N = N_c$. It can be concluded that the computation time $Comp$ decreases with decreasing prediction horizon N. When choosing the right setting of (N, N_c) it is not only important to make the average temperature deviation z_{cl} as small as possible, but also to know how much constraint violation α_{cl} is allowed and what is the sampling time of the system since preferably $Comp << T_s = 1\ s$. Consider for example the following cases:

- **Minimum temperature objective** z_{cl}: The curves in Figure 3.3 show that z_{cl} is the lowest when $N \geq 550$ for $N_c = 1$ and $N \geq 250$ for the remaining control horizons. The corresponding value of z_{cl} is around 0.27 K for the case of $N_c = 1$ and 0.13 K for the remaining control horizons. However, choosing such high prediction horizons comes at the cost of high constraint violations ($24 \leq \alpha_{cl} \leq 30$ %). Another price to pay when increasing the prediction horizon is the increasing computation time $Comp$, which can increase up to 19 seconds.
- **Decreasing maximum constraint violation** α_{cl}: If there is only a maximum constraint violation of, e.g., 10 % allowed, then a setting of $N = 80$ would be sufficient, however the price to pay this time is the increase in average temperature deviation ($z_{cl} = 0.67$ K). The computation time $Comp$ does decrease compared to the previous case because of the choice of lower prediction horizon. The computation time at $N = 80$ reads 0.97 seconds for $N_c = 1$ and 2.15 seconds for $N_c = N$. The computation time of the remaining control horizons are in between the mentioned values.

The cases above shows that there is indeed a trade-off between z_{cl}, α_{cl}, and $Comp$ when choosing the right setting of N, N_c. However, the main goal in this project is to reduce the temperature difference z which will reduce the effect of thermal disturbance underneath the reticle. In doing so, one should be willing to accept the cost of constraint violation and computation time. The curve of z_{cl} shows that, in order to guarantee a minimum temperature difference z_{cl}, one has to choose a prediction horizon which contains sufficient time domain information, i.e. $N \geq \tau\ s$ with τ being a time constant. This depends also on the choice of the control horizon N_c, which determines the freedom that the controller has to minimize the objective z_{cl} given constraints. A larger control horizon results in a faster convergence to the minimum. By choosing a control horizon of $N_c = 3$, a much faster convergence of z_{cl} to its minimum is obtained compared to $N_c = 1$. The choice of increasing the control horizon also compensates for the slow dynamics from $u_h \rightarrow z$. The slowest time constants for $u_h \rightarrow z$ is $\tau = 420$ seconds and for $w \rightarrow z$ is $\tau = 200$ seconds. As can be seen, when there is only one degree of freedom $N_c = 1$, the controller needs N to be at least 420 seconds to reach the minimum z_{cl}. However, when increasing the control horizon to i.e., $N_c = 3$, the controller has more freedom for u_h to compensate for the slow dynamics $u_h \rightarrow z$. Now, to compensate for the slowest dynamics $w \rightarrow z$, one has to choose a prediction horizon of at least 200 seconds. To show this, Figure 3.4 shows a situation where N is smaller than 200 ($N = 130$) and when N is bigger than 200 ($N = 250$). As expected, the temperature difference z for $N = 250$ averages to zero in steady state while in the case of $N = 130$ it averages 1 $^\circ C$ above zero (steady state error). Of course, the price to pay is for the non correctable deformation $||\epsilon_x||_1$ to violate its constraints by a maximum of 28 % for $N = 250$. For the reasons above, we consider that the setting ($N = 250, N_c = 3$) to be the best choice in our case and this setting will be used from now on for the remainder of this report. Note however that results in this Section are obtained using full state information. The next Section will consider output feedback in combination with a Kalman Filter to reproduce the full state.

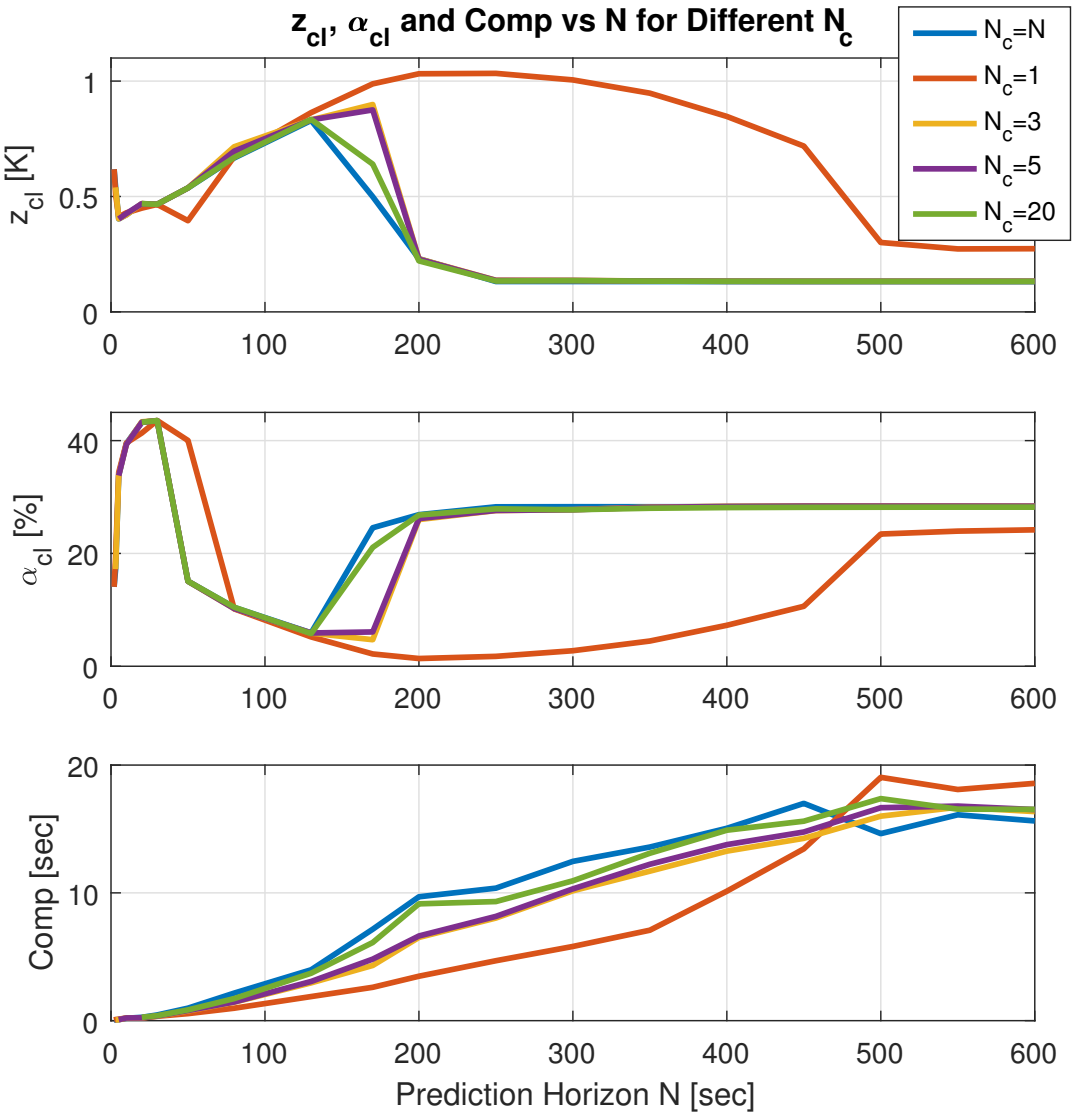


Figure 3.3: Closed-loop performances for for different combinations of N and N_c.

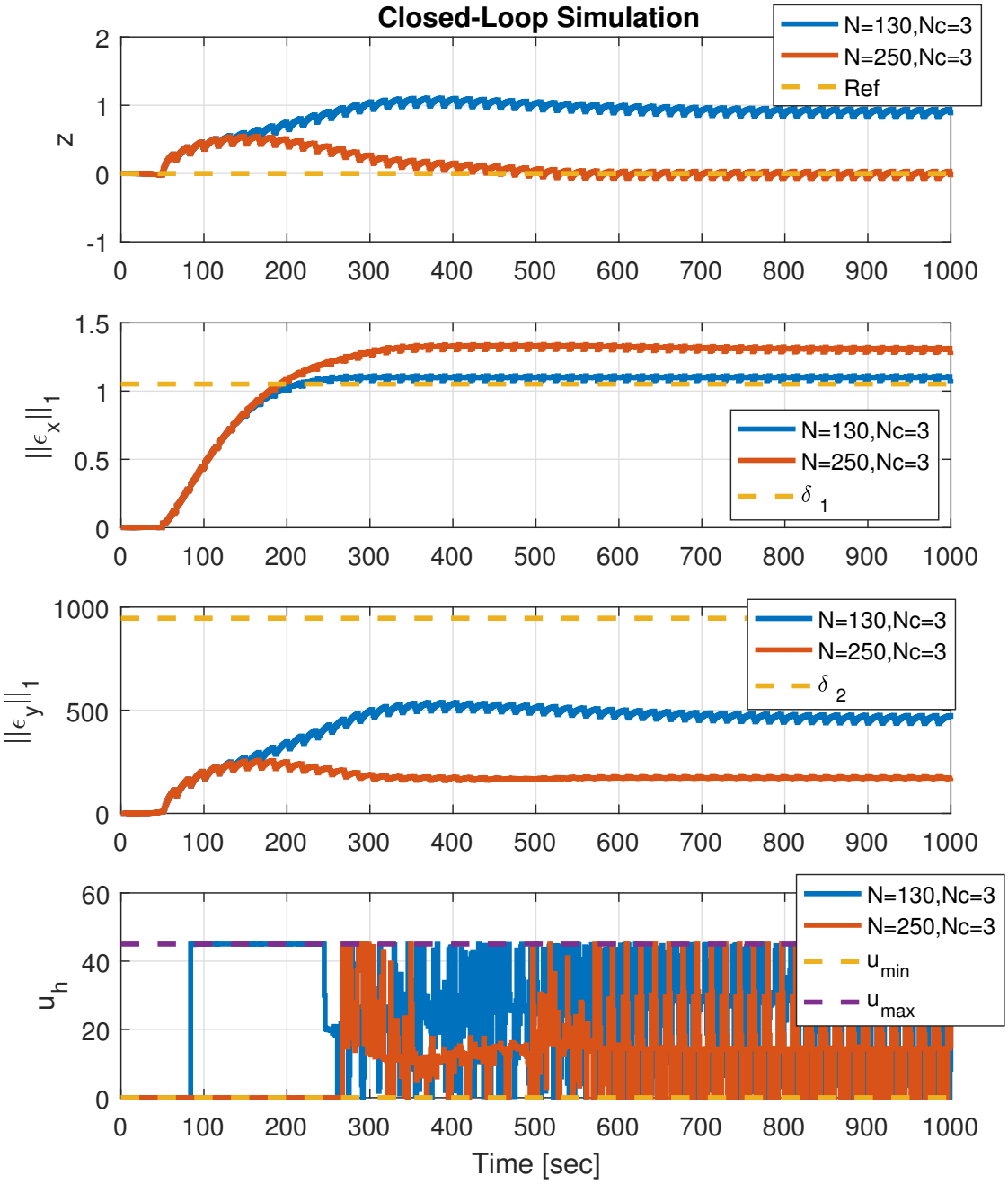


Figure 3.4: Time domain simulation results for the MPC settings $(N = 130, N_c = 3)$ and $(N = 250, N_c = 3)$.

4

Model Predictive Control with Output Feedback

As discussed previously, the results obtained in Section 3.3 are derived using full state information. However, in reality, there is only access to the sensor output $y(t)$. For this reason, Section 4.1 introduces a Kalman Filter design to estimate the entire states of the system in a real-time operation. Section 4.2 discusses the performance loss of a situation in which the initial conditions of the real system differ from the ones used in the Kalman Filter and Section 4.3 does the same with the offset in disturbance w. Lastly, Section 4.4 compares the performance of the designed MPC controller and Kalman Filter with the current state of practice.

4.1. Kalman Filter Design

For the design of the Kalman Filter, the plant P given by (3.1) is used under the assumption that only the measurement $y(k)$ is available for control. However, the full state has to be reconstructed in order to be able to use it in the MPC framework. With the information of the measurement (and measurement noise), the state can be reconstructed according to the following Kalman Filter design

$$\hat{x}(k+1|k) = \Phi\hat{x}(k|k-1) + \Gamma_u u_h(k) + \Gamma_d \begin{bmatrix} T_{pelt} \\ w(k) \end{bmatrix} + Ke(k) \tag{4.1}$$

where $K \in \mathbb{R}^{N_A+N_R+N_S}$ represents the Kalman gain, which is a solution to the Ricatti equation (DARE in Matlab). The estimated state $\hat{x}$ is used as an initialization to the MPC controller in order to solve for the optimization. The error signal $e(k) \in \mathbb{R}^1$ is the difference between the Kalman Filter output $\hat{y}$ and the output of the actual system y such that

$$\begin{aligned} e(k) &= y(k) - \hat{y}(k|k-1) \\ &= C_y\,(x(k) - \hat{x}(k|k-1)) + v(k) \end{aligned} \tag{4.2}$$

where $v \in \mathbb{R}^1$ is a zero mean white noise. The error signal will be used in Section 4.2 as a performance measure for the Kalman Filter.

4.2. Performance Loss due to Offset in Initial Condition

The performance of the MPC as described in Section 3 is based on a certain initial condition x_0 of the plant P. However, in reality, the initial condition used in the observer $\hat{x}_0$ might not always coincide with the real initial condition x_0. Even though the Kalman Filter makes sure that the states will eventually converge, the offset in initial conditions may still affect the performance of the system. Figure 4.1 shows the simulation results when there is an initial condition offset between the real system and the observer. The initial condition of

the real system is chosen such that the reticle has an offset of 500 mK (blue line). As for the first 400 seconds, the temperature difference z for the case of an offset in x_0 shows a worse performance compared to the perfect situation with the same initial condition with the highest values being 1 and 0.5 K, respectively. However, despite the initial condition offset, the temperature difference z for the two cases reach their steady state (or reference) in about the same time span (400 seconds). For the constraint variable $||\epsilon_x||_1$, there is not much difference between the two cases where the maximum constraint violation for both cases ends up to be ± 31 %. The constraint variable $||\epsilon_y||_1$ for both cases satisfies the constraint δ_2 and also reaches the steady-state value after 400 seconds. Not much can be said about the control input u_h other than that u_h for the case of an initial condition offset remains at zero for a longer time period. The reason for this is that the reticle needs to be cooled for a longer time period when it starts at a higher temperature before it reaches its reference (maximum cooling means $u_h = 0$). The error signal e for the observer in the case of initial condition offset shows fast convergence (within 100 seconds), which means that the Kalman Filter does not have difficulties in converging the states after the initial condition offset between the observer and the real system. Note that the observer error shows faster dynamics than the control performance z, which is a recommendation when designing an observer in combination with a controller. The observer error for the case of same initial conditions is around zero for every time step, since in this case there is no offset in the states such that according to (4.2) $e(k) = v(k)\ \forall k$. In conclusion, the observer designed in Section 4.1 is able to get rid of the effect of offset in initial conditions and even though there is a small loss in performance, the performance still reaches its reference in steady state.

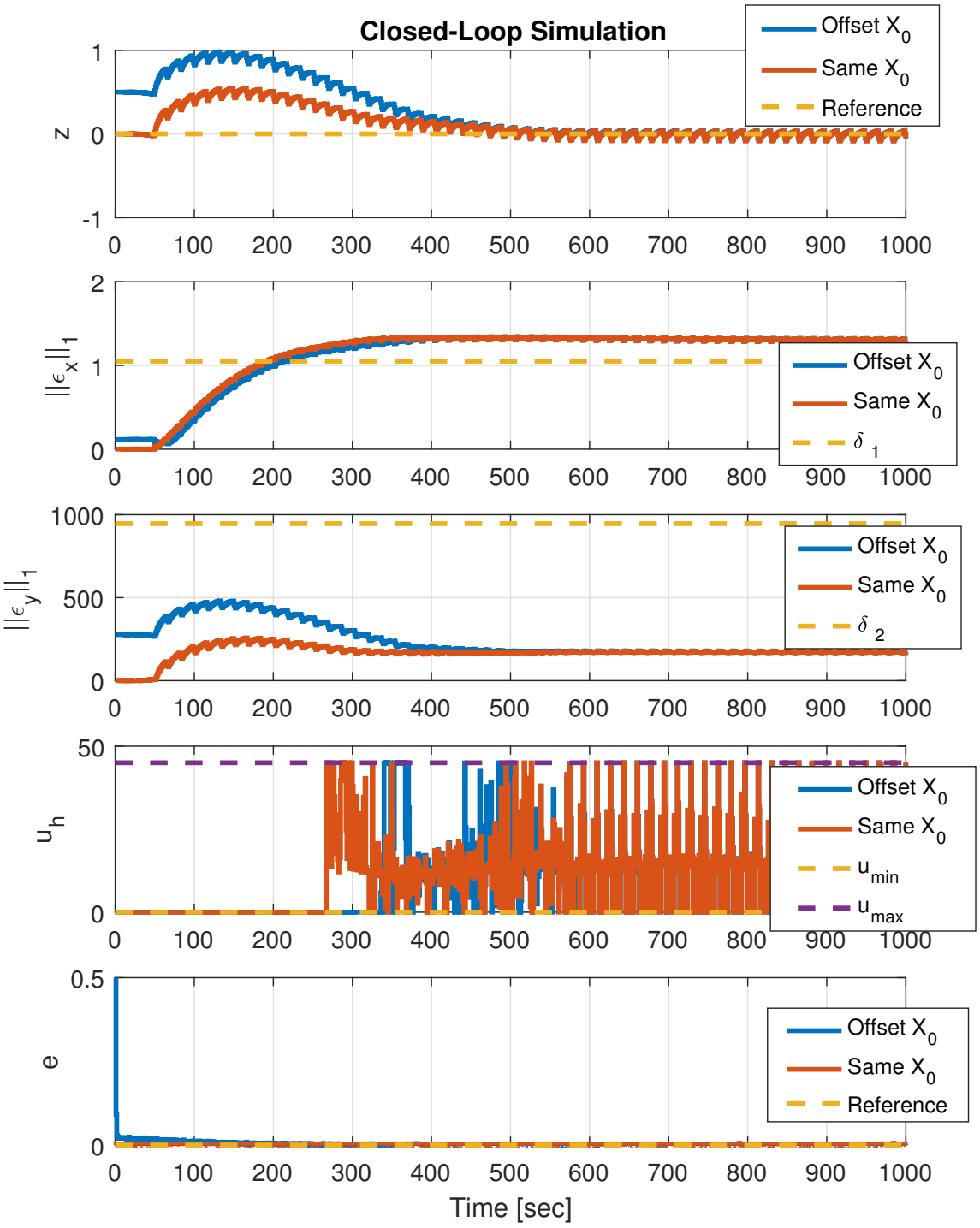


Figure 4.1: Time domain simulation results for the case of an offset in initial condition between observer and real system.

4.3. Performance Loss due to Offset in Disturbance

Another possible situation is when there is a small offset between the disturbance w used in the observer and the real system. This Section discusses the effect of a small offset in disturbance amplitude on the performance compared to the case when disturbance ampli-

tude is known. A realistic situation in practice is that the disturbance on the real system might have a 5 % constant offset in amplitude compared to the real system. Figure 4.2 shows the result obtained in this situation. The temperature difference z has worsened throughout the whole simulation time in the case of an offset in disturbance w compared to the ideal situation. The observer is also not able to reduce this effect, which results in a steady-state error for z (blue line). The observer error e is now biased due to the constant disturbance offset, which implies $E[e(k)] \neq E[v(k)]$ and thus the offset steady state error of z will remain present. However, this can be solved by adding an output integrator and use the Kalman Filter to estimate the constant offset, which is outside of the scope of this project but which is clearly explained in the Matlab's Model Predictive Control Guide (Morari et al [10]).

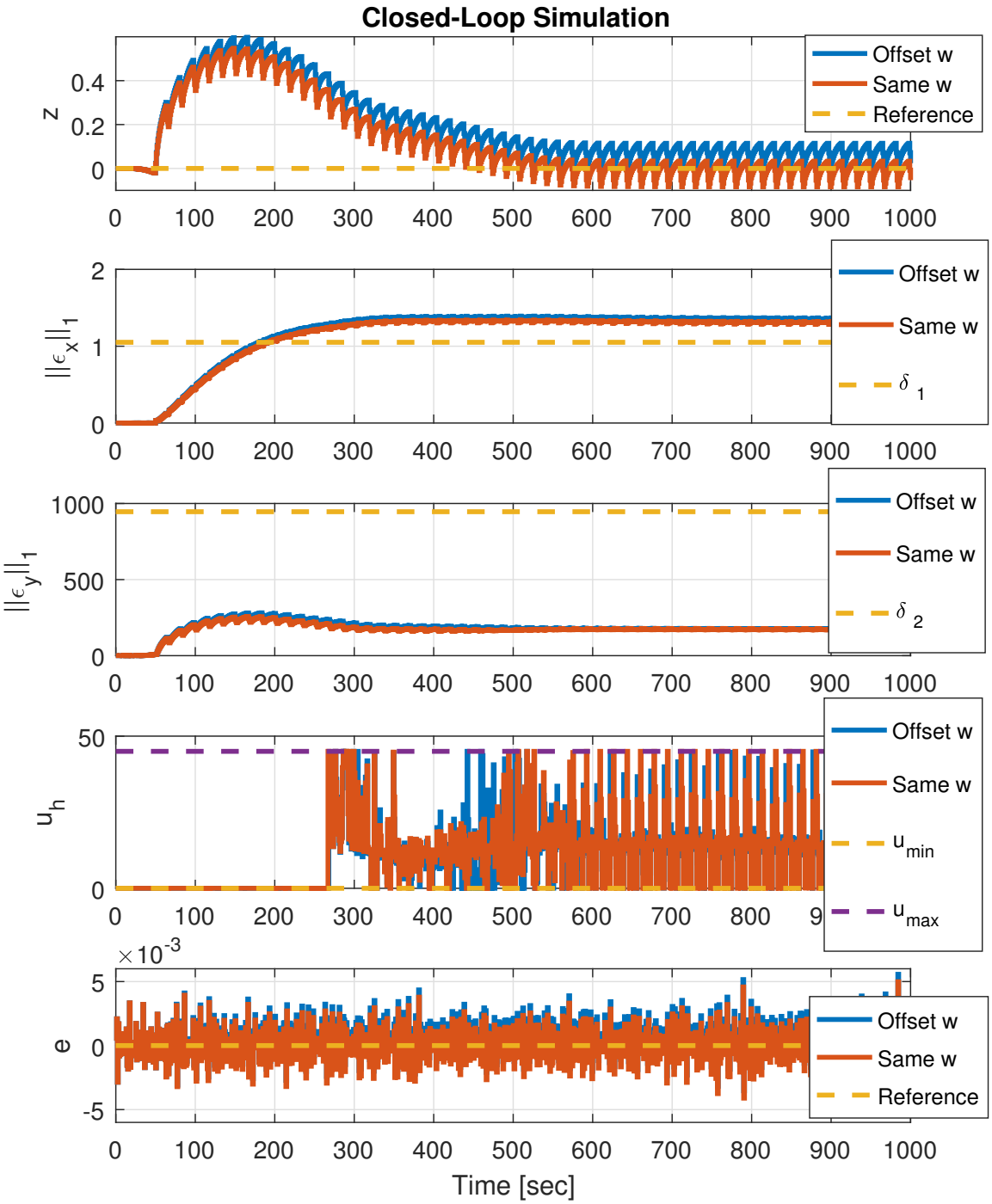


Figure 4.2: Time domain simulation results for the case of an offset in disturbance between observer and real system.

4.4. Results of MPC Versus Current State of Practice

This Section compares the result of the designed MPC and Kalman Filter with the current state of practice. As discussed in Section 1, the current state of practice consists of the Open-Loop Control and the Reticle Deformation Correction method. The biggest difference between

feedback controlled cooling with MPC and Open-Loop cooling is that with MPC adaptation occurs to the change in temperature difference z, while the aim of Open-Loop cooling is to blow cool air at a constant temperature over the reticle. As can be seen in Figure 4.3, the temperature difference z for the Open-Loop case settles around 1.8 K in steady-state, while MPC settles around 0 K as discussed previously. This shows the strength of feedback controlled cooling. However, feedback controlled cooling seemingly introduces more shapes that are not correctable by Reticle Deformation Correction, hence $||\epsilon_x||_1$ for the Open-Loop case is lower than that of MPC. However, due to the implementation of soft constraints, MPC ensures that the non-correctable deformations do not exceed the Open-Loop case too much. $||\epsilon_y||_1$ seems to have been improved with the implementation of the MPC algorithm. The control input u_h between Open-Loop and MPC differ from each other in the sense that the Open-Loop control input reacts only to the nozzle temperature T_{air} until it reaches its constant reference of 21.8 $^{\circ}C$ after 200 seconds where u_h remains constant. The MPC control input on the other hand is actively reacting to the change in z while making sure the constraint violations are kept as small as possible.

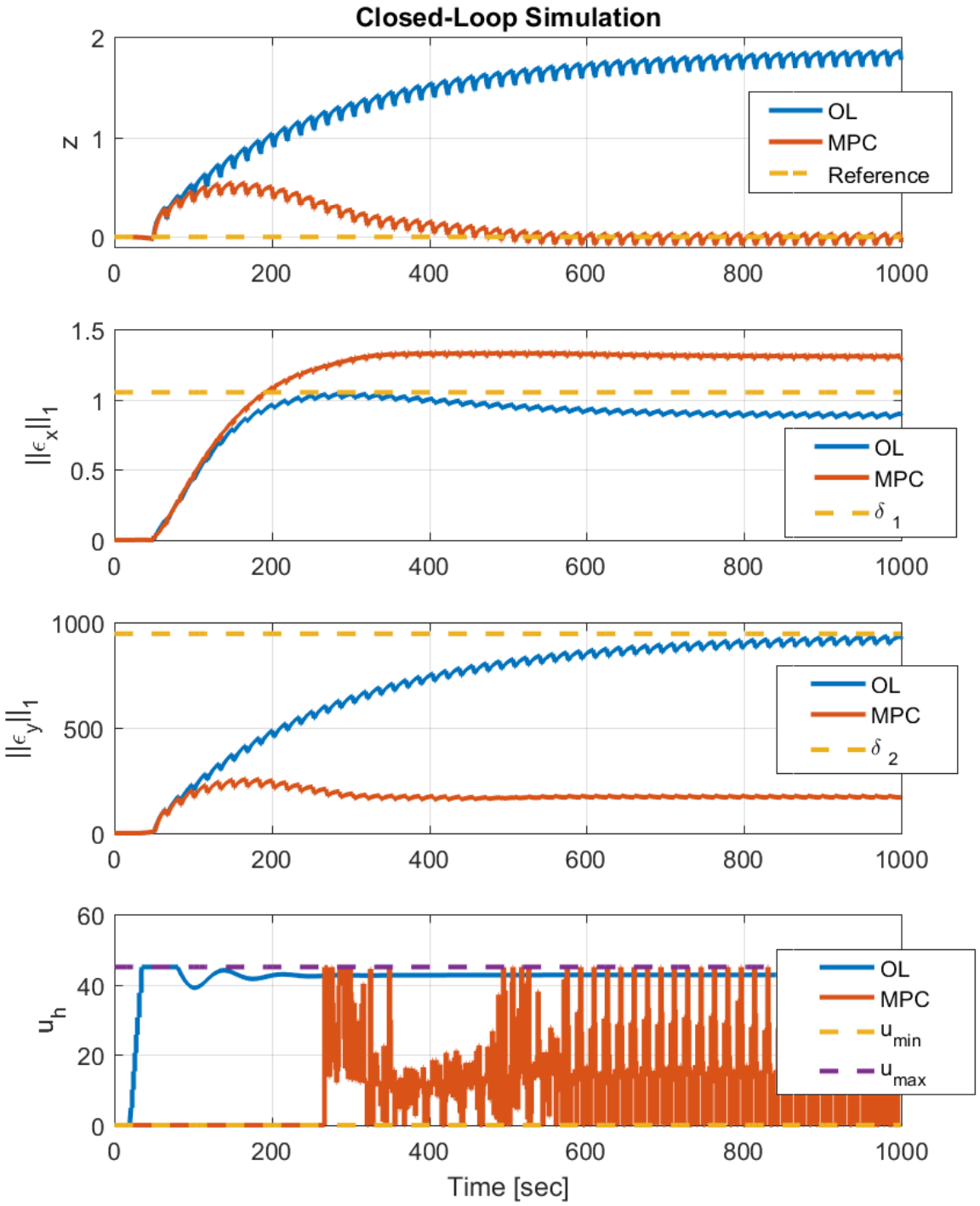


Figure 4.3: Time domain simulation results for MPC and Open Loop Cooling method.

5

Conclusions and Recommendations

In conclusion, in this project, a successful derivation of the process model has been achieved and used for the first time in an MPC framework. Also, a control problem is derived in which the outputs of the process model served as performance and constraint variables in the MPC algorithm. MPC is a suitable approach due to its ability to cope with time domain constraints and due to its predictive nature. Subsequently, tuning guidelines for choosing the optimal control and prediction horizon for the MPC-controller in the context of the reticle cooling application was given. It was shown that by choosing a sufficient control horizon N_c and a prediction horizon N of at least the slowest time constant of the transfer function w to z, it is possible to eventually minimize the effect of the disturbance w in the performance variable z. Also, by using the Kalman Filter in combination with the sensor output to reconstruct the states, it was still possible for the MPC to minimize the performance variable z despite the absence of a direct measurement of z. Lastly, results showed improvements in performance with respect to the current state of practice at the cost of computation time and constraint violation.

The biggest drawback of using MPC in this framework is the high computation time $Comp$. The computation time for the chosen optimal prediction and control horizon of $N = 250$ and $N_c = 3$ is on average 8 s for every closed-loop iteration. Since the controller and model are sampled at $T_s = 1\ s$, this implies that the controller could not yet be implemented in reality. A future recommendation in order to decrease the computation time is to use a lower order thermal model, if possible. A lower model order will reduce the dimension of the optimization problem, which reduces the complexity and thus the computation time. State constraints also increase the complexity of the optimization problem and thus the computation time. However, state constraints are an important part of the problem and cannot be removed. It is thus recommended to replace state constraints with solely an objective function, which means that the problem definition has to be either changed or relaxed in the future. This could be done by replacing everything with one objective function, which minimizes the over-all temperature difference between the reticle and the environment. This is not guaranteed to have the same results (or better) as what is shown in this report, but it still focuses on reducing thermal disturbances as well as thermal deformations.

A

Matrices of the Actuator Model

$$A_A = \begin{bmatrix} -0.10 & -0.09 & -0.02 \\ 0.06 & 0 & 0 \\ 0 & 0.03 & 0 \end{bmatrix},$$

$$B_A = \begin{bmatrix} 0 & 0.03 \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

$$C_A = \begin{bmatrix} 0 & 0 & 0.05 \end{bmatrix},$$

$$D_A = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

B

Matrices of the Sensor Model

$$A_S = \begin{bmatrix} -0.06 & -0.02 \\ 0.03 & 0 \end{bmatrix},$$

$$B_S = \begin{bmatrix} 0.25 \\ 0 \end{bmatrix},$$

$$C_S = \begin{bmatrix} 0.20 & 0.09 \end{bmatrix},$$

$$D_S = 0.$$

C

Matrices of the Total Process Model

$$A = \left[\begin{array}{c|c|c} A_A & 0 & 0 \\ \hline B_R(2)C_A & A_R & 0 \\ \hline B_S D^R_{T_{air_{out}}}(2)C_A & B_S C^R_{T_{air_{out}}} & A_S \end{array}\right],$$

$$B_u = \left[\begin{array}{c} B_A(2) \\ \hline B_R(2)D_A(2) \\ \hline B_S D^R_{T_{air_{out}}}(2)D_A(2) \end{array}\right],$$

$$B_d = \left[\begin{array}{c|c} B_A(1) & 0 \\ \hline B_R(2)D_A(1) & B_R(1) \\ \hline B_S D^R_{T_{air_{out}}}(2)D_A(1) & B_S D^R_{T_{air_{out}}}(1) \end{array}\right],$$

$$\begin{bmatrix} C_{\epsilon_x} \\ C_{\epsilon_y} \\ C_z \\ C_y \end{bmatrix} = \left[\begin{array}{c|c|c} D^R_{\epsilon_x}(2)C_A & C^R_{\epsilon_x} & 0 \\ \hline D^R_{\epsilon_y}(2)C_A & C^R_{\epsilon_y} & 0 \\ \hline D^R_z(2)C_A & C^R_z & 0 \\ \hline D_S D_{T_{air_{out}}}(2)C_A & D_S C_{T_{air_{out}}} & C_S \end{array}\right],$$

$$\begin{bmatrix} D^u_{\epsilon_x} \\ D^u_{\epsilon_y} \\ D^u_z \\ D^u_y \end{bmatrix} = \left[\begin{array}{c} D^R_{\epsilon_x}(2)D_A(2) \\ \hline D^R_{\epsilon_y}(2)D_A(2) \\ \hline D^R_z(2)D_A(2) \\ \hline D_S D^R_{T air_{out}}(2)D_A(2) \end{array}\right],$$

$$\begin{bmatrix} D^d_{\epsilon_x} \\ D^d_{\epsilon_y} \\ D^d_z \\ D^d_y \end{bmatrix} = \left[\begin{array}{c|c} D^R_{\epsilon_x}(2)D_A(1) & D_{\epsilon_x}(1) \\ \hline D^R_{\epsilon_y}(2)D_A(1) & D_{\epsilon_y}(1) \\ \hline D^R_z(2)D_A(1) & D_z(1) \\ \hline D_S D^R_{T air_{out}}(1) & D_S D_{T_{air_{out}}}(2)D_A(1) \end{array}\right].$$

Bibliography

[1] Rawlings, James B. ”Tutorial overview of model predictive control.” *IEEE Control Systems* 20.3 (2000): 38-52.

[2] Qin, S. Joe, and Thomas A. Badgwell. ”A survey of industrial model predictive control technology.” *Control engineering practice* 11.7 (2003): 733-764.

[3] van den Boom, Ton JJ, and T. C. P. M. Backx. ”Model predictive control.” *DISC Course, Lecture Notes* 16 (2010).

[4] Zafiriou, Evanghelos, and André L. Marchal. ”Stability of SISO quadratic dynamic matrix control with hard output constraints.” *AIChE Journal* 37.10 (1991): 1550-1560.

[5] de Oliveira, Nuno MC, and Lorenz T. Biegler. ”Constraint handing and stability properties of model□predictive control.” *AIChE journal* 40.7 (1994): 1138-1155.

[6] Zafiriou, Evanghelos. ”Robust model predictive control of processes with hard constraints.” *Computers & Chemical Engineering* 14.4-5 (1990): 359-371.

[7] Hovd, Morten, and Richard D. Braatz. ”On the use of soft constraints in MPC controllers for plants with inverse response.” *IFAC Proceedings Volumes* 34.25 (2001): 251-256.

[8] Hovd, Morten, and Richard D. Braatz. ”Handling state and output constraints in MPC using time-dependent weights.” *American Control Conference, 2001. Proceedings of the 2001.* Vol. 3. IEEE, 2001.

[9] Rawlings, J. B., and D. Q. Mayne. ”Postface to “Model Predictive Control: Theory and Design”.” *Nob Hill Pub (2012):* 155-158.

[10] Morari, Manfred, and N. Lawrence Ricker. Model predictive control toolbox: for use with MATLAB. *MathWorks Incorporated, 1998*.

[11] de Carvalho Monteiro, Patrik. ”Literature report” *Adaptive Reticle Cooling at the Area of Interest, 2018*.