STATE SPACE MODELS AS A COMMON TOOL FOR CONTROL DESIGN, OPTIMIZATION AND FAULT DETECTION IN BUILDING SYSTEMS

Kai Kruppa¹, Thorsten Müller², Björn Lautenschlager¹, Gerwald Lichtenberg¹, Nicolas Réhault²

¹Hamburg University of Applied Sciences, Germany
²Fraunhofer Institute for Solar Energy Systems, Germany

ABSTRACT

The article shows that parameter identification methods allow a user-friendly modeling process when measurement data and control signals of an HVAC system are available from a monitoring system. With the help of one identified and validated model, several automation tasks as controller design and fault detection can be carried out for performance optimization. The reader is guided through the whole engineering cycle by an example of a heating system.

INTRODUCTION

The next major step in the development of building automation is expected through consistent introduction of model-based engineering methods. Today, these methods often seem abstract and can only be successfully used with large effort and well-trained staff. In building automation, model-based engineering methods are not as widespread as in other disciplines, which allows significant improvements even by application of standard techniques.

As part of the BMWi project Optimizing Building Services (OBServe), model-based methods for the continuous monitoring and optimization of building operations are developed and tested in eight demonstration buildings. Here the interaction of models and methods is investigated with the aim of using the same models for multiple tasks in order to generate the maximum benefit from the modeling effort.

Often these approaches seem to be very complex such that they need much engineering effort. But as soon as measurement data of sensors and actors is available from monitoring, a model can be constructed by parameter identification techniques in a simple way. Therefore, some standard approaches to derive state space models from data of heating systems are summarized here.

State space models are standard representations of models of dynamical systems, Chen, 1999. Thus, many methods in automation are available for state space systems, e.g. in control, Aström et al., 2008 or diagnosis, Isermann, 2006. Even a simple model of the plant can be used for several purposes in building automation. The paper shows how models of heating systems can be constructed without large effort and how these models can be used for fault detection and performance optimization, i.e. controller design.

For fault detection a qualitative model is derived from numerical simulation data and the state space model. Qualitative models have been proposed since decades, but due to their large data structures they never have been applied widely. Here we show a solution how they can be applied to large discrete time systems.

The same simple state space model is used to design a model predictive controller (MPC). MPC is a promising model based control strategy for heating systems and building climate control to reduce and optimize its energy consumption e.g. by considering the weather forecast or occupation, Privara et al. 2011; Henze 2013.

The paper is organized as follows. First some theoretical aspects on state space models and modeling and parameter identification techniques are described. Next a simple model of an example heating system is introduced which is then used for the design of a MPC and fault detection in the next parts.

STATE SPACE MODELS

The dynamical behavior of a physical system should be described mathematically in this section. The described standard approaches can be found in Chen, 1999 or Rugh, 1993. This means that the relation between input signals \( u(t) \) and output signals \( y(t) \) is computed, which is described by the system block in Figure 1.

![Figure 1: System](image)

In many applications the system dynamics are captured by an ordinary differential equation (ODE) of order \( n \). The differential equation describes the relationship between \( u \) and \( y \). It is assumed that the dynamical properties of the system do not change over time such that the coefficients of the ODEs are constant. These coefficients are denoted as parameters of the system. The \( n \)-th order ODE can be written as a system of first order ODEs. In general, the right hand
sides of the ODEs are nonlinear. This leads to the description of the dynamical system behavior as a nonlinear state space model
\[ \dot{x}(t) = f(x(t), u(t)), \]
\[ y(t) = g(x(t), u(t)), \]
where \( f(x, u) \) is the state transition function and \( g(x, u) \) is the output function. The system has \( n \) states, \( m \) inputs and \( r \) outputs.

Nonlinear state space models are the most general class of models since they allow arbitrary functions as right hand sides. More restrictive model classes are linear state space models. The ODEs are described by linear functions. Thus, a linear state space model is given by
\[ \dot{x}(t) = Ax(t) + Bu(t), \]
\[ y(t) = Cx(t) + Du(t), \]
with system matrix \( A \), input matrix \( B \), output matrix \( C \) and feedthrough matrix \( D \). A linear system with two states, two inputs and one output leads to a state space model
\[ \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \]
\[ y = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}. \]

The representation of a model in state space format is not unique. Different state space models can represent the same dynamical system. It depends on the choice of the state variables. Thus it is possible to transform a state space model in another state space representation by a so called state transformation. In the linear case this can be done by a transformation matrix \( T \) that transforms the state variables \( x \) to the new states
\[ \bar{x} = T^{-1}x. \]
To get a state space model that is valid in the transformed states \( \bar{x} \)
\[ \dot{\bar{x}}(t) = \bar{A}\bar{x}(t) + \bar{B}u(t), \]
\[ y(t) = \bar{C}\bar{x}(t) + \bar{D}u(t), \]
the state equations have to be transformed too by
\[ \bar{A} = T^{-1}AT, \quad \bar{B} = T^{-1}B, \]
\[ \bar{C} = CT, \quad \bar{D} = D. \]

With these transformations the state space models (2) and (5) are equivalent, i.e. they show the same input-output behavior.

There are some special forms of state space models that are suitable to analyze the system behavior. These forms are called normal forms. One very common normal form is the canonical normal form of the state equation. The transformation matrix \( T_c \) of the canonical transformation is constructed by the eigenvectors of \( A \) as column vectors of \( T_c \). Transforming a linear state space model with this transformation matrix leads to a model (5) with a system matrix
\[ \bar{A} = T_c^{-1}AT_c = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix}, \]
where the eigenvalues \( \lambda_i \) of the matrix \( A \) are on the diagonal. This gives information of the system, since the eigenvalues describe some dynamical properties of the system.

So far all signals were assumed to be known at every time. This leads to the description of the system in continuous-time. In a discrete-time state space model the states, inputs and outputs are known at fixed sample time steps only, e.g. \( x(0\cdot ts), x(1\cdot ts), \ldots \) for the states with the sample time \( t_s \). For ease of notation the sample time is often neglected such that \( x(k\cdot ts) = x(k) \). In discrete-time the system behavior is not described by differential equations but difference equations since at time \( k \) the next system state \( x(k+1) \) depends on the actual and past states and inputs. Thus a linear state space model in discrete-time is given by
\[ x(k+1) = Ax(k) + Bu(k), \]
\[ y(k) = Cx(k) + Du(k). \]

**MODELLING**

The following section shows how to get from a dynamical system to a model in state space structure. There are different approaches possible with different requirements. Three different methods are highlighted and summarized here from Ljung, 1987.

For the first approach the structure of the physical system has to be known exactly. The dynamical behavior is represented in detail by differential equations, that can be transformed to a system of first order differential equations, i.e. a state space model. This results in a very accurate model when the differential equations describe the system dynamics very detailed. Thus, all internal signals are interpretable since they all have a physical meaning. The parameters of the model are determined by information on the system. This information must be available e.g. from data sheets. Measurement information of the system is not necessary. This makes it possible to derive a model already in the construction phase of a system, even though a prototype is not available. The drawbacks of this approach are that a very detailed knowledge of the system is necessary to describe the dynamical effects mathematically. Additionally, the different components of the system have to be documented very well to get the system parameters. If e.g. data sheets are not available, it is hard to determine or estimate the system parameters. For an accurate model many physical effects have to be considered in this approach. Thus, one also gets a model of high com-
plecty which leads to a large computational effort when working with this model, i.e. simulation, diagnosis or controller design. The modelling effort is very large too. This approach is called white box modelling.

In grey box modelling the model is based on physical equations too. Only the most important dynamical behavior, not all physical effects of the system are captured by differential equations. Minor effects are neglected to reduce the model complexity while the real behavior of the system is described approximately well. The differential equations can be written in state space format. All internal signals have got a physical meaning, which is helpful for plausibility checks during modelling. The parameters of the model can be determined by using measurement data of the system. Therefore, no pre-knowledge of the parameters is necessary but measurement information from the inputs and outputs of the system has to be available. The goal of the parameter identification with measurement data is to estimate the parameters such that the difference between simulated and measured output data is minimized when the same input data is applied. The difference between simulated and measured data is computed by a cost function which is often chosen as the sum of quadratic differences. Figure 2 depicts the method schematically.

![Figure 2: Parameter identification](image)

An optimization algorithm varies the parameter values and simulates the system again with these parameters iteratively until a minimum of the cost function is found. This means that the difference between measured and simulated output data is minimal for this parameter set. Having found a parameter set, the estimation result has to be validated by a second independent data set. The approximation accuracy of the model can be checked by the cost function. The advantage of this approach in contrast to the parameter determination from data sheets is that parameters are directly fit to the specific component and its application and less modelling effort is necessary.

The last approach shown here is the black box identification. Input-output measurements are used to find a state space model. The system is taken as a black box without any knowledge of the underlying physical relationships. First of all, a model class has to be fixed. Here we focus on the black box identification of linear systems in state space format. From experiments, the measurements of the inputs and outputs have to be available. An optimizer determines the parameters of the model, i.e. \( A, B, C \) and \( D \) matrices in the linear case, such that the quadratic difference between the measured and the simulated outputs is minimized. Thus, the model is directly fitted to the application. The information on the system is taken from the measurement data. There are very efficient algorithms available for black box identification of linear state space models, like the N4SID algorithm (Numerical algorithm for subspace state space system identification), van Overschee et al., 1996.

With the N4SID algorithm a linear state space model can be constructed very fast when measurement data is available. Since no information on the physical relationships of the system is used here, it is not guaranteed that specific properties of the system are captured by the model. If the information of some properties is not contained in the measured data, the model cannot capture these effects. The internal signals are not interpretable here so that the model describes the input-output behavior of the component only.

**EXAMPLE**

In this section the model of a heating system will be introduced. This model is the basis for the fault detection part and the design of a model predictive controller (MPC).

This model represents the heating system of a school building and includes two parts, a supply and a consumer part. The basic model structure was introduced in Pangalos et al., 2013. The heating system is modelled as a grey box model like introduced in the Modelling section, where only the most important dynamical behavior of the system is captured by the model. The supply part is represented by a boiler with a burner, which satisfies the heat demand of the consumer. The consumer, which includes the radiators and the building, is modelled as a one zone model. This means that the whole building is summarized into one room and all radiators are centralized into one radiator, which leads into a model with a high abstraction level of the heating system (see Figure 3).

![Figure 3: Scheme of the heating system](image)

The boiler and the consumer are modelled by thermal heat balances, which lead into three differential equations. The consumer is supplied by the boiler with the supply temperature \( T_s \). The differential equation of the supply temperature is given by

\[
\dot{T}_s(t) = \frac{\dot{V}(t) (T_i(t) - T_s(t))}{V_{\text{boiler}}} + \frac{P_{\text{in}(t)} - h_{\text{boiler}} (T_s(t) - T_u)}{c\rho V_{\text{boiler}}} \tag{9}
\]

where \( V_{\text{boiler}} \) is the volume of the boiler, \( c \) and \( \rho \) are the specific heat capacity and density of water, \( \dot{V} \) is
the volume flow and $T_r$ is the return temperature, $k_{\text{boiler}}$ is the heat transfer coefficient from the boiler to the surrounding, $T_o$ is the surrounding temperature and $P_{\text{in}}$ is the thermal power of the boiler. The thermal power can be controlled by the modulation signal $\alpha$ and is given by $P_{\text{in}} = \alpha P_{\text{max}}$ where $P_{\text{max}}$ is the maximal thermal power of the boiler and $\alpha \in [0,1]$.

The building includes two heat transfers. On the one hand the heat transfer from the radiators to the building, which satisfies the heat demand of the building. It will be assumed that the heat transfer is proportional to the difference between the building temperature $T_b$ and the return temperature $T_r$. On the other hand, the building has thermal losses to the environment, proportional to the difference between the building temperature $T_b$ and the outside temperature $T_{\text{out}}$. This leads into the differential equation of the building temperature

$$\dot{T}_b(t) = \frac{k_{b,c}(T_b(t) - T_s(t)) - k_{b,o}(T_b(t) - T_{\text{out}}(t))}{C_b},$$

with the heat capacity of the building $C_b$, the heat transfer coefficient from the radiators to the building $k_{b,c}$ and the heat transfer coefficient from the building to the outside $k_{b,o}$.

Heat is transferred to the building by the radiators, supplied by the volume flow $V$ with the supply temperature $T_s$. Cold water with the return temperature $T_r$ leaves the radiators and returns to the boiler. The return temperature $T_r$ is given by the differential equation

$$\dot{T}_r(t) = \frac{\dot{V}(t)(T_s(t) - T_r(t))}{V_c} = \frac{k_{r,c}(T_s(t) - T_r(t))}{c_p V_c},$$

with the overall volume of the radiators $V_c$.

The volume flow changes with the time. The pump determines the volume flow $\dot{V}$ in dependency of the difference between the actual building temperature $T_b$ and the desired reference of the building temperature $T_{b,r}$. For large differences between the building temperature and the reference building temperature the volume flow is in saturation. Because of that it is assumed that the difference between $T_b$ and $T_{b,r}$ is not too large, which leads into the linear relation

$$\dot{V}(t) = \dot{V}_{\text{mean}} + b(T_{b,r}(t) - T_b(t)), \quad (12)$$

with the mean volume flow $\dot{V}_{\text{mean}}$ and the slope $b$.

The three differential equations combined with the equation of the volume flow describe the dynamic behavior of the heating system. The result is a continuous-time nonlinear model in state space form (1). The state space model has three states, the supply temperature $T_s$, the building temperature $T_b$ and the return temperature $T_r$ and three inputs, the modulations signal $\alpha$, the outside temperature $T_{\text{out}}$ and the reference of the building temperature $T_{b,r}$. That results in the input and state vector

$$\mathbf{u} = [\alpha \quad T_{\text{out}} \quad T_{b,r}]^T, \quad \mathbf{x} = [T_s \quad T_b \quad T_r]^T. \quad (13)$$

The three outputs of the model are equal to the states.

The unknown parameters of the model are identified by using measurement data of the system. This grey-box process is described in the Modelling section. Considering the boiler as an example the differential equation (9) includes three unknown parameters, the maximum power of the boiler $P_{\text{max}}$, the heat transfer coefficient $k_{\text{boiler}}$ and the volume of the boiler $V_{\text{boiler}}$. That means that the identification algorithm estimates the values of these three parameters. The same procedure of the parameter identification will be applied to the other unknown parameters of the model. Figure 4 shows the simulation result of the building temperature, after the parameter estimation, compared to the measurement data of the building temperature.

![Figure 4: Comparison measurement - simulation](image)

The result shows that the model only represents the main dynamics of the heating system.

For many applications a linear model is beneficial. Therefore, the nonlinear state space model has to be linearized around an operating point. Linearization means that the nonlinear state space model of the heating system is approximated by a linear one such that the dynamical behavior of the plant is approximated well by the linear model in the neighborhood of the operating point. The operating point is a steady state point in the region of the nominal plant behavior. The accuracy of the linear state space model is only sufficient in an area around the operating point. Thus, linearizing the model of the heating system leads to the linear state space model with the common form, see (2), with 3 states, 3 input, and 3 outputs. The states are equal to the outputs, i.e. the supply temperature, the building temperature and the return temperature.

For the fault detection and the controller design part a discrete-time state space model is necessary. Because of that the linear continuous-time state space model of the heating system will be discretized with the sample time $t_s = 60$ s. The result is a discrete-time linear state space model, see (8). Figure 5 shows the comparison between the simulation results of the nonlinear continuous-time model (blue line) and the linearized discrete-time model (red line) of the supply temperature $T_c$. One can see that the linear model captures the dynamical behavior of the system around the operating point well.
CONTROLLER DESIGN

In the following the linear discrete-time state space model will be used to design a controller for the heating system as introduced in Maciejowski, 2002.

Model predictive control

Inside the controller the model of the heating system should be used to predict the reaction of the plant on future control inputs. This is the idea of model predictive control (MPC) with the structure shown in Figure 6. Using the model, the controller computes the optimal future control inputs at each sampling instant with respect to a cost function describing the control goal, e.g. tracking a reference trajectory.

A linear MPC is designed here. This means that the linear state space model (8) is used inside the controller. With this model the controller acts as follows. Using the actual state of the plant $x(k)$ as initial value, the controller tries to optimize the control sequence by minimizing the quadratic cost function

$$ J(k) = \sum_{i=1}^{H_p} \left[ (y(k+i) - r(k+i))^T Q(k) (y(k+i) - r(k+i)) + \Delta u(k+i)^T R(k+i)) \right] $$

(14)

with input changes $\Delta u(k+i) = u(k+i) - u(k)$. A reference for the outputs is given by $r(k)$. Thus, minimizing the cost function means that the outputs of the plant should follow a reference trajectory with a certain control effort. The prediction horizon $H_p$ gives the number of time steps the controller predicts the future plant behavior. The inputs are changed during the prediction for the control horizon $H_u$. The weighting matrices $Q$ and $R$ are used to weight the difference of the outputs from the reference and the input changes respectively. An increase in $Q$ puts more weight on the output reference tracking which in general results in a larger control effort. Increasing $R$ leads to less change in the control signal which yields in general in a slower system. Thus, tuning the weights of the controller is a tradeoff between control effort and tracking performance.

Every sampling step the function (14) is minimized. The controller solves the optimization problem

$$ \min_{u(k+i)} J(k), \quad \text{s.t.} \quad u_{\text{min}} \leq u(k+i) \leq u_{\text{max}} $$

(15)

where the resulting optimal control inputs $u_{\text{opt}}(k+i)$, with $i=1,...,H_u$, are constrained by $[u_{\text{min}}, u_{\text{max}}]$. Using the quadratic cost function in combination with the linear state space model shows the advantage of linear MPC. With a linear state space model the optimization problem (15) is convex and can be solved very efficiently by standard quadratic programming solvers. Using a nonlinear model would lead to more computational effort during optimization. This is an important factor for MPC since it has to be guaranteed that the result of the optimization has to be available at the next sample step. The first element of the resulting control sequence $u_{\text{opt}}(k+1)$ is given to the plant. The minimization is repeated at every time instance. This is called the moving horizon principle.

Closed loop simulation

Applying this design method to the heating system leads to a closed loop as shown in Figure 6. Since the controller works on a given sample rate the discrete-time linear state space model is used. For controller design the inputs of the plant are differentiated to control and disturbance inputs. Control inputs are the signals that are available for the controller to influence the actuators of the system. Signals from the disturbance inputs are outside influences that disturb the system behavior and cannot be influenced by the controller. Applying this differentiation of the inputs to (8) results in the linear state space model

$$ x(k+1) = A x(k) + B u(k) + B_d u_d(k) $$

$$ y(k) = C x(k) + D u(k) $$

(16)

with 3 states, 1 control input denoted by $u$, 2 disturbance inputs denoted by $u_d$ and 3 outputs. The states are equal to the outputs, i.e. the supply temperature, the building temperature and the return temperature. The input to the model is the modulation signal of the boiler. The disturbances are the outside temperature and the reference for the room temperature.

By minimizing the cost function

$$ J(k) = \sum_{i=1}^{H_p} Q \cdot (T_{s,r}(k+i) - T_{s,r}(k+i))^2 $$

(17)

the controller sets the modulation signal of the boiler such that the supply temperature follows the supply temperature reference $T_{s,r}$. The controller gets information on the supply, return and room temperature.
from the plant to get its actual state. Additionally, estimated predictions of the disturbances, i.e. outside temperature and room temperature reference are given to the MPC. The controller uses the disturbance predictions to simulate the plant behavior. To achieve a good closed loop performance, the controller parameters $H_p$, $H_u$, $Q$ and $R$ have to be tuned. The control and input horizon were set to 6 and 4 hours respectively. The weightings were chosen such that a good tradeoff between tracking of the supply temperature reference and variation in boiler modulation signal is achieved. The sampling time of the MPC is one minute. The designed controller was tested in closed loop simulation with the nonlinear state space model. The result of the simulation is shown in Figure 7, where the supply temperature reference and the simulated supply temperature are compared.

Figure 7: MPC simulation

The figure shows that the controller sets the modulation signal such that the boiler follows the supply temperature reference well.

QUALITATIVE MODELLING

In contrast to quantitative models, qualitative models describe the behavior of a system only roughly. Instead of numerical values, qualitative models deal with a symbolic representation of the system signals.

The process shown in Figure 8 is a dynamic discrete-time, continuous-variable system whose qualitative behavior is described by a quantized system which in turn can be represented by a qualitative model.

Figure 8: Quantized system, Schröder, 2003

The numerical input $u(k)$, output $y(k)$ and state $x(k)$ vectors consist of a finite number of components, where each component belongs to a system signal. Each of these input $u(k)$, output $y(k)$ and state $x(k)$ signals is quantized into a number of qualitative intervals (see Figure 9).

The qualitative modelling approach showed here uses a stochastic automaton (SA) as qualitative model, Lunze, 1994. The qualitative signal values $[u(k)]$, $[z(k)]$ and $[y(k)]$ are assigned to equivalent automaton inputs $w=(v_1(k),...,v_m(k))$, states $z=(z_1(k),...,z_n(k))$ and outputs $w=(w_1(k),...,w_r(k))$, where the variables $m$, $n$ and $r$ define the number of the numerical input, state and output signals. A stochastic automaton can be defined as 5-tuple

$$S = (N_z, N_v, N_w, L, p_z(0)).$$ (18)

The vector $p_z(0)$ is introduced in the Section Fault Detection. Figure 10 shows an automaton graph of a small system with $N = 4$ qualitative states, $M = 2$ qualitative inputs and $R = 2$ qualitative outputs.

Figure 10: Automaton graph

In the case at hand the graph shows the possible state transitions from $z$ to $z'$ for the input $v = 1$. Thereby the round circles denote the states of the SA. The green edges represent the transition probabilities for the output $w = 1$ and the blue ones for the output $w = 2$. For a fixed input $v$ and state $z$, the sum of the transition probabilities for a state change is 1

$$\forall z, v : \sum_{z' \in N_z} \sum_{w \in N_w} L(z', w, z, v) = 1$$ (20)

The SA with the behavior relation

$$L(z', w, z, v) =$$

$$p_{z}(0) \left( \begin{array}{c} [x(k+1)] = z' \\
[y(k)] = w \\
[u(k)] = v 
\end{array} \right)$$ (21)

is called the qualitative model, Lichtenberg and Lunze, 1997. The behavior relation $L$ of the SA is
represented as an n-way array, resp. a tensor of dimension
\[ L \subset [0,1]^{N_1 \times \cdots \times N_n} \times R_1 \times \cdots \times R_e \times N_1 \times \cdots \times N_m \times M_1 \times \cdots \times M_m \] (22)

Here it becomes clear, that a large number of inputs, outputs and states lead to an exponential growth of the behavior relation (see Table 1).

For making fault detection with qualitative models applicable to large discrete time systems, the amount of values to be stored and the calculation effort have to be reduced. As shown in Müller et al., 2015, this can be done by tensor decomposition methods.

Table 1: Transitions depending on the number of signals and qualitative intervals

<table>
<thead>
<tr>
<th>#</th>
<th>n</th>
<th>( N_i )</th>
<th>( M_i )</th>
<th>( r )</th>
<th>( R_i )</th>
<th>Values in ( L )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>( 5^2 \cdot 5^1 \cdot 5^1 = 625 )</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>4 ( \cdot 5^2 \cdot 4^1 = 1 , 638 , 400 )</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>6</td>
<td>( 5^0 \cdot 4^1 \cdot 6^1 = 5.4 \cdot 10^{11} )</td>
</tr>
</tbody>
</table>

We will use a non-negative CP decomposition, because the behavior relation contains probabilities and negativity of the tensor elements has to be avoided. The term CP is referred to different names in literature, but here it relates to the CANDECOMP (canonical decomposition) and PARAFAC (parallel factors) models. For further reading and notations see Kolda and Bader, 2009 and Cichocki et al., 2009.

For our case the CP decomposition of the behavior relation can be denoted as
\[ L = [Z_1, \ldots, Z_n, W_1, \ldots, W_r, Z_1', \ldots, Z_n', V_1, \ldots, V_m] \cdot \lambda \] (23)

The parameter \( \lambda \) is a weighting vector and the terms inside the parentheses are called the factor matrices of the states \( Z_i \), the outputs \( W_r \), the successor states \( Z'_i \) and the inputs \( V_i \). The number of rows of the factor matrices is fixed and given by the number of qualitative intervals of the respective signal. The number of columns is given by the so-called rank of the decomposition, which is adjustable. In general, choosing the rank of the decomposition is a trade-off between storage amount and approximation accuracy. For the example in the 2nd row of Table 1 the number of values to be stored can be reduced down to 2150 by choosing a rank of 50 for the decomposition.

FAULT DETECTION

In the field of Fault Detection, Lichtenberg and Steele, 1996 proposed an approach based on qualitative observation. The qualitative observation algorithm is initialized with a probability vector \( p_{\alpha}(0) = (1/N_i, \ldots, 1/N_i) \) and gives a prediction for the successor state probability distribution \( p_{\alpha}(k+1) \) based on the current measured input output combination \( (w(k), v(k)) \). Thereby, the probability vector \( p_{\alpha}(k+1) \) yield non-vanishing probabilities as long as the measured input output tuple is consistent with the qualitative model. If the measured input output combination is inconsistent with the qualitative model, the probability vector contains only zeros and a fault can be detected. However, the algorithms developed in the 1990s are only applicable to matrix representations of the behavior relation. An algorithm, that is applicable to the CP decomposed tensor structure in (23), has been developed by Müller and Lichtenberg, 2016
\[
p_{\alpha}(k+1) = Z_i' \cdot \left( \lambda \otimes \left( \bigotimes_{y=1}^{n} Z_{y} \right) \cdot p_{\alpha}(k) \right) 
\otimes \left( \bigotimes_{q=1}^{r} W_{q} \cdot p_{\alpha}(k) \right) \otimes \left( \bigotimes_{s=1}^{m} V_{s} \cdot p_{\alpha}(k) \right)
\] (24)

For demonstration, the system shown in Figure 3 is used and a qualitative nominal model was generated by the use of the stochastic qualitative identification method (Lichtenberg, 1998). For the qualitative model \( m = 3 \) inputs and \( r = 2 \) outputs were taken into account. The inputs are related to the reference building temperature \( T_{\text{ref}} \), the modulation signal \( \alpha \) and the outside temperature \( T_{\text{out}} \). Each input signal is partitioned into \( m = 9 \) qualitative intervals. The signals of the building temperature \( T_{\text{in}} \) and the return temperature \( T_{\text{r}} \) are used as outputs. Since return and supply temperature show similar qualitative dynamics, the return temperature is used here only for reduction of dimensionality. The state signals were set equal to the output signals \( (x(k) = y(k)) \) and then transformed into the canonical normal form with the transformation matrix \( T \) and the use of equation (4), what lead to the new state variables \( \hat{x}(k) \). Thus, the number of state signals is \( n = 2 \). The number of qualitative intervals is \( R_1 = 9 \) for the output signal \( T_{\text{in}} \) and \( R_2 = 12 \) for \( T_{\text{r}} \). The same hold for the state signals \( (N_1 = 9, N_2 = 12) \). The nominal system behavior was trained over a time period of 5 days, by the use of the simulation data from the modelling section.

Figure 11 shows the qualitative state trajectory of \( \hat{x}_2 \) for a selected nominal time range. The state trajectory is a result of the qualitative observation algorithm and it shows the probability distribution of the state vector for each discrete time step \( k \). The different grey shades of the bars denote the probabilities. Note that due to the state transformation the state variables no longer have a physical meaning.

For demonstrating the fault detection ability, a fault has been embedded in the numerical simulation mod-
el. The fault applies to the supply temperature sensor which is not mounted deep enough into the thermowell, what is an often occurring fault in real systems. In the case at hand the sensor measures a 13 °C too low temperature what lead to a too high real supply temperature as well as to an increasing room temperature.

Figure 12 shows the fault indication signal, where 1 means that the system is in nominal condition and 0 means that a fault has appeared. As can be seen, the fault was detected around timestep $k = 3000$, what coincides with the beginning of the faulty simulation period.

In the simulation model, the faulty operation occurs continuously from its first appearance until the end of the simulation. As the figure shows, there are some false negative detected areas what depend on the fact that the qualitative behavior in these areas is equal to that of the nominal condition. In our use-case, the behavior relation of the SA has $918 \ 330 \ 048$ possible state transitions what makes it impossible to install such a qualitative model directly on a Building Management System. The behavior relation from the example above was decomposed with rank 100 what lead to an amount of values to be stored of 9100 and corresponds to a reduction-factor of more than 100 000.

CONCLUSION

This paper shows the benefits of model based methods for building automation. A simple model of a heating system of a school building could be derived. A validation with measurement data from a real building showed that the dynamics of the real system are captured well. The system was approximated by a linear discrete-time state space model. With this model a qualitative detection and a model predictive controller were designed. Simulations showed promising results for the detection part as well as for the controller part. Both approaches are based on the same model which shows that once the modeling is done it could be used for different purposes in building automation to optimize the building performance.

REFERENCES

Müller, T., Lichtenberg, G., 2016. Fault Detection with CP-Decomposed Qualitative Models. Accepted. To be presented at the 4th IFAC ICONS (2016), Reims, France.