Using Non-Markov models for the Control of Dynamic Systems *

RAINER DEVENTER, JOACHIM DENZLER, HEINRICH NIEMANN

Chair for pattern recognition
Martensstr. 3
D 91058 Erlangen
Telephone: +49 (0)9131 85-27825
{deventer, denzler, niemann}@informatik.uni-erlangen.de

Abstract. Due to shorter life cycles and more complex production processes the automatic generation of models for control purposes is of great importance. Even though Bayesian networks have proven their usefulness in machine learning and pattern recognition and the close relationship between Dynamic Bayesian networks and Kalman Filters respectively difference equations they have not been applied to problems in the area of automatic control. In our work we deduce the structure of a Dynamic Bayesian networks using the state space description and difference equations. Both models are trained by the EM algorithm and used for control purposes. The experiments show that both models performs well, but the training process of the model based on difference equations is much more stable.
Keywords: Dynamic Bayesian Networks, Dynamic System, Controller

1 Introduction

The life cycle of new products is shortening more and more as time to market is a very important point. To guarantee an optimal quality despite complex products and shorter life cycles quality management and intelligent controller, e. g. based on process-models, gain more interest. In most cases the manual development of models is too time consuming, thus models which can be trained by available data are necessary.

Requirements for those training algorithms are the ability to cope with missing or unobserved data, and with discrete and continuous values at the same time. As we aim at the development of a controller the calculation of suitable inputs using a desired output is also one of our requirements.

Even though statistical modeling and particularly Bayesian networks (BN) have ideal prerequisites for modeling purposes being proven in other domains, e.g. data mining and medicine, there are only a few applications in the engineering domain. Due to the close relationship between the description of dynamic systems, e.g. by difference equation and state space description, it is also possible to use a-priori knowledge to simplify the training. In section 3.2 the parallels between the state space description and the description by difference equation on the one hand and Dynamic Bayesian Networks (DBN) on the other hand are exploited to infer an optimal structure for modeling linear dynamic systems by DBNs. The resulting networks are trained by the well-known EM algorithm and step and impulse response as training signals. It is shown that the performance of the trained models is sufficient to act as controller.

The efficiency of the difference equation model as well as the state space model is compared with a controller whose parameters are inferred from a mathematical model. The model based on the difference equation clearly outperforms the state-space model. The training process is more stable and can be done with less iterations.

Both the analytical retrieved state-space model and the trained difference equation model show a good performance, the desired value is reached with low overshoot and the deviation of the actual value from the desired value is below 1%. Given a successful training process the same results are obtained by the state space model.

Even if we focus in the paper on the modeling of stationary, linear dynamic systems of second order, the extension to control nonlinear systems is straight forward using hybrid BNs, i.e. a BN using both discrete and continuous nodes. As shown in [3] hybrid BNs are able to model nonlinearities by approximation of a function by a Taylor series with multiple support points.

An additional advantage of our approach is the possibility to learn not only the parameters of a given structure, but both the structure [4] and probabilities [1] of the Bayesian network from examples.

This article is structured as follows. In section 2 the term control system is defined and the analytical descriptions applied in control theory are introduced. Section 3 deals with Bayesian networks and the structure used for the controller. The usage of BNs to generate control signals is explained in section 4. The results obtained with this new type of controller are presented in section 5. The article finishes with a short summary.

* This work was funded by the ,,Deutsche Forschungsgemeinschaft" (DFG) under grant number SFB 396, project-part C1

2 Dynamic systems and control theory

Humans are regularly encountered with controlling tasks. Typical examples are driving a car. Here the driver has to take care of his speed to avoid being stopped by the police. A lot of controllers are also found in industry. Generally spoken it is the task of the controller to keep the observed output value $\boldsymbol{q}$, e.g. the speed of a car as close as possible to a desired value $\boldsymbol{w}$, e.g. the maximal speed allowed. If the environment is changing, e.g. the speed limit is changed or the car is slowed down due to a hill, the controller should change the input so that the difference between the desired value and the current value is reduced as fast as possible.

This section is structured as follows. First the main elements of control loops are introduced. Afterwards a mathematical description of the dynamic system is given. The used description is taken from control theory, to ensure a broad field of application. Additionally this description is employed to deduce the structure of a Bayesian network used for control purposes.

2.1 Performance measures

A typical controller is triggered by the error e , i.e. first the desired value w is compared with the output q of the system. When driving a car, e.g. the desired speed is compared with the current speed. The difference between desired and current value e is used as input for the controller which calculates the input for the actuator. The changed input value of the dynamic system results in a new output, where the manipulation reaction describes how the system responds to new inputs.

Additionally the controller has to deal with influences from the environment. As we do not need a model for the system's reaction to disturbances, it is assumed that the disturbances take effect at the output of the system. Thus the observed output q

$$q(t) = y(t) + z(t) \tag{1}$$

results from adding the disturbing value z to the undisturbed output y of the system. As our experiments are done with a single-input, single output system, the output q is a scalar, in theory usually systems with vector valued outputs $\boldsymbol{q}$ are discussed.

The most frequently used controllers use the error as input. In contrary our Bayesian controller use former in- and output and the desired value instead. The internal model of the controller is able to predict the system's reaction and calculate the new input accordingly.

To judge the quality of a controller a measure is needed. In control theory a lot of measures are known and the selection depends on the application. A metric that is frequently used integrates the squared error

$$Q = \int_0^\infty (e(t) - e_\infty)^2 dt \tag{2}$$

with $e(t)$ as the difference between the desired value $w(t)$ and the actual value $q(t)$. The term e_∞ is the steady state error and is necessary to guarantee that the integral remains limited even if there remains a small error. This measure has to be adapted as we work in a time discrete environment. Therefore the integral is replaced by

$$Q_d = \sum_{t=0}^{t_{max}} \Delta T (e_t - e_\infty)^2 \tag{3}$$

with t_{max} as time until convergence took place. Please note that e_t denotes the error in a time-discrete system, whereas e(t) is used for systems with time as a continuous variable. For the evaluation of our experiments convergence is assumed when the deviation of the last 25 output signals is below 0.0001.

To get an impression of the convergence ratio the time $te_{max}(c)$ is measured until the relative error is below $c\%$. For example assume that the measured steady state error is 4.5%, thus $te_{max}(3\%) = \infty$, as the remaining error is greater than 3%. But if fast convergence is observed t_{max} may be very small.

Additionally the so called steady state error e_∞, i.e. the remaining error when the output signal is converged, is used as quality measure.

The next section 2.2 deals with a mathematical description of the manipulation reaction to deduce the structure of a suitable model.

2.2 Controlled systems

A dynamic system may be regarded as a black box with several input and output signals $\boldsymbol{u}$ and $\boldsymbol{y}$ respectively, where the output does not depend solely on the input signal, but additionally on an internal state $\boldsymbol{x}$. Linear, time-invariant systems are regularly described by differential equations

$$\sum_{i=0}^{n} a_i \frac{d^i \boldsymbol{y}(t)}{dt^i} = \sum_{j=0}^{m} b_j \frac{d^j \boldsymbol{u}(t)}{dt^j} \tag{4}$$

whereas only systems with $m <= n$ are physical realizable. It is possible to rewrite the differential equation (4) of order n to n differential equations of first order

$$\frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \tag{5}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{E}\boldsymbol{u}(t) \tag{6}$$

called the state-space description. The transition matrix $\boldsymbol{A}$ describes the transition from one state $\boldsymbol{x}$ to the next, $\boldsymbol{B}$ the influence of the input $\boldsymbol{u}$ on the state. The output $\boldsymbol{y}$ depends on the state, as described by $\boldsymbol{C}$ and on the input $\boldsymbol{u}$, depicted by $\boldsymbol{E}$. In the models used later the state is only regarded at discrete time steps. Therefore equations (5) and (6) has to be rewritten to

$$\boldsymbol{x}_{t+1} = \boldsymbol{A}_{BN}\boldsymbol{x}_t + \boldsymbol{B}_{BN}\boldsymbol{u}_t \tag{7}$$

$$\boldsymbol{y}_t = \boldsymbol{C}\boldsymbol{x}_t + \boldsymbol{E}\boldsymbol{u}_t \tag{8}$$

where the different values of $\boldsymbol{A}_{BN}$ and $\boldsymbol{B}_{BN}$ in comparison to $\boldsymbol{A}$ and $\boldsymbol{B}$ results from the adaptation to time discrete systems. Also the differential equation is rewritten for discrete time, so that the new output signal y_t

$$\boldsymbol{y}_t = -\sum_{i=1}^{n} \alpha_i \boldsymbol{y}_{t-i} + \sum_{i=1}^{n} \beta_i \boldsymbol{u}_{t-i} \tag{9}$$

is a weighted sum of former input and output signals. Thus a dynamic system can not only be described by the more common state space description [5], but also by a difference equation which does not meet the Markov assumption. It will be shown, that this disadvantage is more than compensated by a higher modeling accuracy, and a more stable signal generation. This results from the removal of the unobservable state nodes which leads to a stable training procedure. The state space description from equations (7) and (8) as well as the difference equation (9) leads to a special structure of the BN.

3 Bayesian networks

Modeling with BNs is equivalent with learning a probability distribution $p(X_1, X_2, \cdots, X_n)$ which represents the data to be modeled, e.g. the input- output behavior of a dynamic system, as well as possible. Assuming independencies between multiple random variables X_i, the joint distribution simplifies to

$$p(x_1, x_2, \cdots, x_n) = p(x_1) \cdot \prod_{i=2}^{n} p(x_i | pa(i)). \tag{10}$$

where the instantiation x_i of a random variable X_i depends only on its parents $Pa(X_i)$. The instantiation of $Pa(X_i)$ is denoted by $pa(i)$. Usually BNs are represented as graphs with the random variables as nodes. A link from X_i to X_j, i.e. $X_i \in Pa(X_j)$ means that X_i has a direct influence on X_j.

There are several types of BNs, which can be distinguished by the type of nodes used. We restrict ourselves to normally distributed, continuous nodes. The distribution p for a node X with parents Y is

$$p(\boldsymbol{x}|\boldsymbol{y}) = \mathcal{N}(\boldsymbol{\mu}_{X_0} + \boldsymbol{W}_X \boldsymbol{y}, \boldsymbol{\sigma}_X), \tag{11}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are the instantiations of X and Y, $\boldsymbol{\mu}_{\boldsymbol{X}_0}$ is the mean when no parent exists or all parent have zero values. The weight matrix $\boldsymbol{W}_X$ is used to characterize the influence of Y on X. $\boldsymbol{\sigma}_X$ denotes the covariance matrix of the normal distribution. The restriction to normally distributed nodes enables us to use the inference algorithms described in [8], avoiding time consuming sampling procedures. Additionally there is no need to bother about convergence problems. This is important as a controller has to react in real-time. One of the most important operations on BNs is the calculation of marginal distributions. Given a full distribution $p(X)$ with $X = \{X_1, \cdots, X_n\}$ an arbitrary distribution $p(X \backslash C)$ with $C \subset X$ can be calculated by integration over all variables in C:

$$p(X \backslash C) = \int_C p(X) dC. \tag{12}$$

A more detailed description of the algorithms used for BNs are given in [8, 9] or [6].

3.1 Dynamic Bayesian networks

For many purposes a static description is sufficient, see e.g. [3]. But there are a lot of applications when time is an important factor, i.e. the distribution of a variable $X(t)$ depends not only on other variables, but also on its own value at a previous time step. One example are systems described by equations (5) and (6). For such cases dynamic Bayesian networks (DBN) are developed, which are able to monitor a set of variables at arbitrary, but fixed points of time.

For each modeled point in time a static BN is used. These time slices are linked to represent the state of a variable at different points in time. In a DBN the random variables are regarded as normally distributed, i.e. when the state space description is used the distribution of the next state

$$p(\boldsymbol{x}_{t+1} \mid \boldsymbol{x}_t, \boldsymbol{u}_t) = \mathcal{N}(\boldsymbol{A}_{BN} \boldsymbol{x}_t + \boldsymbol{B}_{BN} \boldsymbol{u}_t, \boldsymbol{\sigma}) \tag{13}$$

depends on the input $\boldsymbol{u}_t$ and the former state $\boldsymbol{x}_t$. For the evaluation a DBN can be interpreted as a static BN with equal parameter for all time slices respectively between the time slices. A deeper introduction is found in [7]. Well known DBNs are Hidden Markov Models and Kalman filter which are mostly used in control theory for tracking and prediction of linear dynamic systems.

3.2 Structure of the Bayesian Network

This section introduces the structure of the used DBN. The first network is inferred from the difference equation, the second from the state-space description. Both are suitable to model dynamic systems, but as we will see the former uses less hidden nodes, i.e. nodes to model unobservable values. This leads to better training results.

Using equation (9) results in links from the input to the output $u_{t-2} \rightarrow y_t$, $u_{t-1} \rightarrow y_t$ and also from former undisturbed output to the current output $y_{t-2} \rightarrow y_t, y_{t-1} \rightarrow y_t$. Equation (1) results in links $y_t \rightarrow q_t, z_t \rightarrow q_t$ with a fixed weight of one. Additionally it is assumed that the statistical characterization of the disturbing value does not change from one time slice to the next, thus z gets a low covariance matrix, and a link with weight one $z_t \rightarrow z_{t+1}$ is added. These considerations lead to the structure depicted in figure 1.

As it can be seen in figure 1 the Markov assumption is not used in this model, y_t depends e.g. on u_{t-2} and y_{t-2}. To be able to use standard tools we mapped the structure

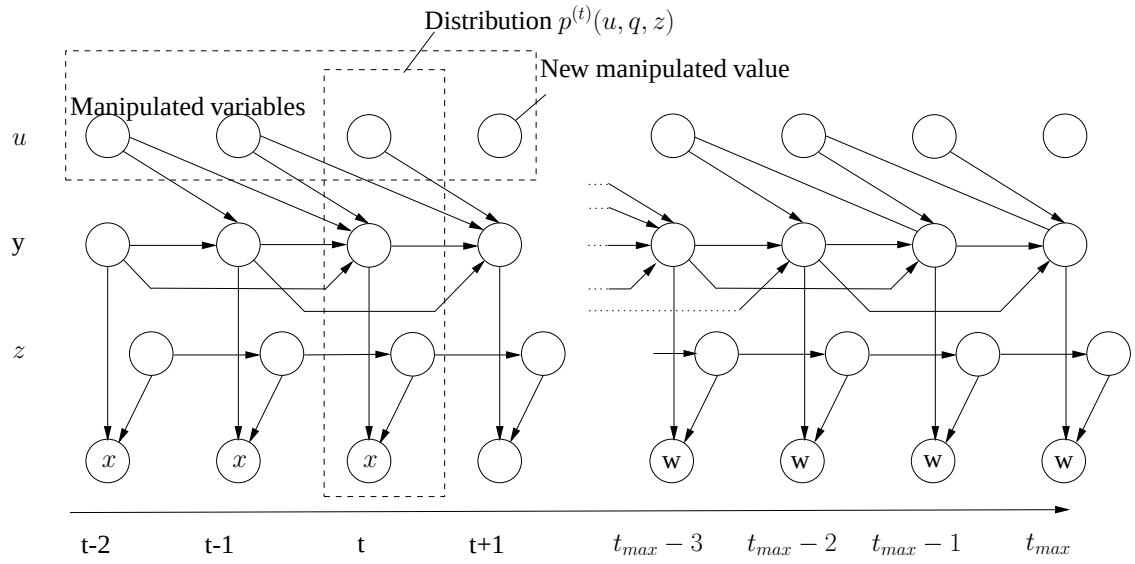

Fig. 1. Principal structure of a BN used for control purposes

of figure 1 to a DBN which meets the Markov Assumption. The trick is to use redundant nodes, i.e. two nodes in different time slices get the same value. This assumption is guaranteed either by assignment of the same evidence to two nodes (Assignment of former input to u) or by a link with weight 1 (y_t is represented in two different time slices. These two nodes are connected by a link with weight one.) between two nodes. The modeling of

the disturbance variable is not changed. The resulting model, called Difference equation model, is depicted in figure 2. The advantage of this model is that there are only two unobserved nodes. For training purposes z we assume that there is no disturbance, thus no hidden nodes are left. This results in a stable training process.

This model is compared with the state-space description of equations (7) and (8) where the new state is calculated using the former state and the input. That results in the links $x_t \rightarrow x_{t+1}$ and $u_t \rightarrow x_{t+1}$. The output depends on the state and the input, thus the links $x_t \rightarrow q_t$ and $u_t \rightarrow q_t$ are needed. The latter can be omitted if changing the input leads to no immediate change of the output. The disturbing value is modeled by separate random values with a link of weight one between z_t and z_{t+1} and z_t and q_t. These considerations result in the structure of figure 3.

4 Calculation of control signals

In section 3.2 the structure of the BN is inferred from a mathematical description, weight matrices, covariances and means are trained using the EM algorithms with the exception of the parameters of node z and the covariance matrix of u which is set to a great value, to ensure that u is changed to come to a instantiation with a high-probability.

The next question to be answered is the sampling rate. According to control theory the natural angular frequency of a second order system $K\boldsymbol{u}(t) = \boldsymbol{y}(t) + T_1\frac{d\boldsymbol{y}}{dt} + T_2^2\frac{d^2\boldsymbol{y}}{dt^2}$ is $\omega_0 = \frac{1}{T_2}$ which is reduced by damping the system. The minimal sampling rate has to be twice the frequency of the system. Of course this provides only a hint because usually damping and T_2 are unknown. This concludes our discussion about the structure and the parameters of the DBN, we proceed to the calculation of control signals.

For the generation of the input variable u a DBN with a fixed number of time-slices as depicted in figure 2 respectively in figure 3 is used. To generate the manipulated value u_{t+1} the first part of the input nodes is used to enter the history. In figure 2 these are the nodes u_{t-2} up to u_t. For our experiments we used 10 nodes for the representation of the past and 15 for the future, thus t_{max} is 25 [1]. Moreover the observed output values are stored and entered as evidence using the nodes q_{t_0}, the oldest stored output value, till q_t. The undisturbed output and the disturbance variable cannot be observed, so no evidence is given for the random variable y and z. Now it is the task of the DBN to calculate a signal that can be used to change the system's output to the desired signal and to keep that output constant. To tell the system to do so the desired value w is also entered as evidence. This means the desired future values for the output nodes are treated as they were already observed and entered as evidence for all the nodes q_{t+2} till $q_{t_{max}}$. No evidence is given for q_{t+1} as this value is determined by an already calculated input. To control the plant it is necessary to calculate the value u_{t+1} which leads to the desired

[1] The experiments were done with Matlab, Simulink to simulate the dynamic systems, and the BN-Toolbox, an expansion of Matlab which is freely available at http://www.cs.berkeley.edu/∼murphyk/Bayes/bnt.html

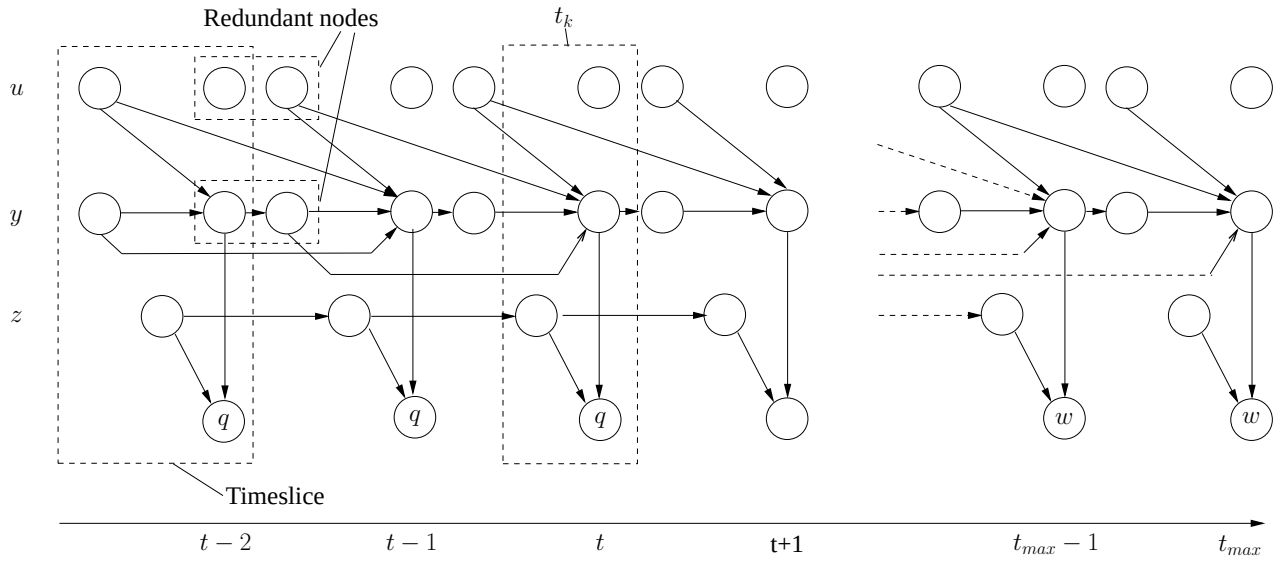

Fig. 2. Non-Markov-Model with redundant nodes

value w. This can be done by marginalization. The new input is passed to the simulation of the dynamic system and the resulting output is calculated. Then a complete cycle is finished. The used input and the resulting output are added to the history and the next input is calculated. To ensure that the calculation of the input signal is not limited to a certain amount of time the evidence is shifted to the left after each time step, i.e. the oldest input and output values are deleted. Then the current signal is entered at time t. The future values may remain unchanged if the desired value is not changed. This works well for slow systems, for systems with the ability to oscillate this would result in oscillating input signals. Thus it is necessary to damp the input. To do so a weighted sum of u_t to u_{t+k} is used. For our experiments, described in section 5.2, a weighted sum of 4 nodes is used.

5 Experiments

5.1 Test systems

Our experiments were done with three different systems of second order which are simulated by Simulink. These systems are described by differential equations

$$K\boldsymbol{u}(t) = \boldsymbol{y}(t) + T_1\frac{d\boldsymbol{y}}{dt} + T_2^2\frac{d^2\boldsymbol{y}}{dt^2} \tag{14}$$

and their behavior depends mainly on the two parameters T_1 and T_2. If the damping $D = \frac{T_1}{2T_2}$ is greater than one the system has no tendency to overshoot which means these systems are easy to control. Systems with $0 < D < 1$ have the ability to oscillate and overshooting can be regarded when the input signal is changed. Our test systems are described in table 1.

Table 1. Description of test systems

Nr	K	T_1	T_2	Description
1	2	1	0.1	Damped system with gain two which has no tendency to overshoot
2	2	0.1	0.1	System with $D < 1$ which means that there is a tendency to overshoot
3	10	0.05	0.1	System with high gain and a large tendency to overshoot

5.2 Comparison state-space and Non-Markov model

First experiments were done with state-space models whose parameters were retrieved analytically. The reason for these experiments is to get comparative values. The model calculation is described in [2]. The main idea is to get the weights and means of a Bayesian network by comparison with a Kalman Filter [5]. This analytical model is compared with a state-space model [13, 12] whose parameters are trained by the EM-algorithm [11] and a model based on the difference equation described in section 2.2. The training of the state space model was stopped after 20 iterations done with 40 examples each. During the training of the difference equation model it is assumed that there is no disturbance, thus z = 0 and q = y. Using these assumptions there are no hidden nodes left. It is therefore possible to reduce the number of iterations to five. Due to the reduced number of iterations the training of the difference equation model is much faster. All three models are used to control the test systems described in table 1 in the following scenario. First w and z was set to zero. After convergence w was set to 10, to test the reference reaction of the control loop. When the output settled down to its new value z was set to 1, so that

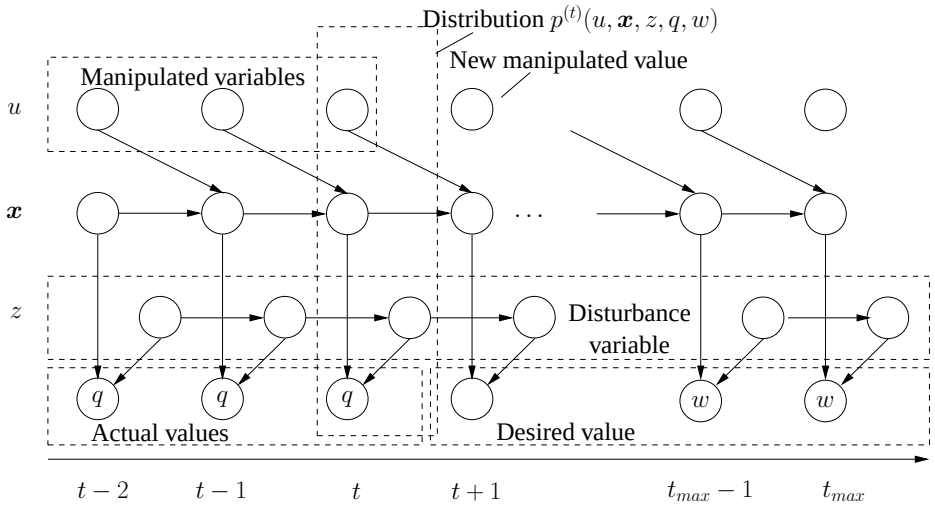

Fig. 3. State space model

Table 2. Used quality measures

$Q_d(z = d)$	Squared error sum as defined in equation (3). Tests done with a disturbance variable $z = d$.
Overshoot	The difference between the maximal output value q_{max} and w.
e_∞	The remaining error $e = q - w$ after convergence takes place.
$te_{max}(z = 0, c\%)$	Raise time until the output has changed from q = 0 to $q = w$ when no disturbance occurs.
$te_{max}(z = 1, c\%)$	Settling time until the error is smaller than p% starting with the occurrence of the disturbance input.

the disturbance reaction of the system could be examined (see figure 4 for the signals of the non-Markov controller). The test criteria, discussed in section 2.1 are summarized in table 2. The results, being the mean of 10 experiments, are given in table 3. Regarding the reference action for the test system 2 all three models perform well, with slightly better performance for the trained state space model. The reason for the smaller Q_d of the state space model are greater input signals which leads to a greater overshoot. This is in compliance with control theory. Regarding dynamic systems two and three the trained non-Markov-model clearly outperforms the trained state space model. For dynamic system number 2 the state space model fails to converge one time, the mean is therefore calculated only from 9 cases. The table shows therefore a slightly distorted view, because the convergence problems are not visible. The state space controller of system 3 fails two times to reach the desired value w within an accuracy of 1%. Thus the mean of the steady state error is much greater in that case. If only the successful training cases were regarded the mean of e_∞ = 0.01, the mean of the overshoot is 1.07.

Table 3. Results

	Anal. state space			Tr. state space			Difference eq.		
System	1	2	3	1	2	3	1	2	3
$Q_d(z = 0)$	8.4	9.62	10.28	7.53	9.47	11.62	8.51	9.82	9.92
$e_\infty(z = 0)$	0.02	0.01	0	0.01	0.0	0.26	0.07	0.04	0.01
Overshoot	0.02	0.56	0.92	0.48	0.88	1.73	0.07	0.45	1.15
$Q_d(z = 1)$	0.15	0.19	0.23	0.16	0.27	0.43	0.11	0.16	0.19
$e_\infty(z = 1)$	0	0	0.01	0.03	0.01	0.35	0.03	0.05	0.03
$te_{max}(z = 0, 1\%)$	0.45	0.45	0.7	0.68	0.72	2.95	0.82	0.45	0.77
$te_{max}(z = 0, 3\%)$	0.35	0.4	0.45	0.48	0.53	2.67	0.42	0.4	0.51
$te_{max}(z = 1, 1\%)$	0.45	0.55	0.6	0.53	0.78	3.18	0.36	0.35	0.6
$te_{max}(z = 1, 3\%)$	0.3	0.35	0.45	0.33	0.62	2.56	0.25	0.25	0.26

The results for the disturbance reaction are similar. In that case the relative steady state error for system 3 is greater than one in three cases.

The result of the comparison of the state space model with the non-Markov-model is that both models are suitable for control purposes, but the training of the non-Markov-model is much faster and more stable.

6 Summary and future works

Starting from an analytical description two different structures of a dynamic Bayesian network are developed. These networks are used as controller by using the desired value as evidence and using the marginal distribution of the input nodes as input signal for the controlled system.

The performance of a Bayesian controller based on the state space description is compared with the structure resulting from the difference equation regarding both the reference and the disturbance reaction of the controlled system. It is shown that both systems are suited for control purposes, but better training results are observed for the model inferred from the difference equation.

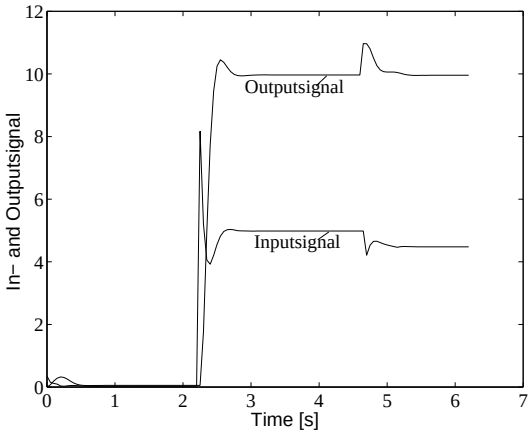

Fig. 4. Signals for system 2 controlled by a trained controller

There are several means to deal with higher order systems. Single input, single output systems can be modeled by a serial connection of systems of second order. The other possibility is to increase dimension of the state nodes (state space model) or the number of former nodes used for the prediction of the next output. Of course this would result in an increase of time complexity, particularly for the state space model the training would get more complicated.

For the future there will be several points of interest. To get a better impression of the practical applicability the performance of a Bayesian controller has to be compared with PID controller which are widely used in industry and it has to be examined how the Bayesian controller can be optimized with respect to special constraints, e.g. no overshoot or limited input. Other important points are the ability to react in real time and the modeling of non-linearities. An approach to cope with non-linearities are hybrid Bayesian networks. This approach leads to additional discrete hidden nodes which might result in run time problems. A second approach are particle filters used e.g. under real time conditions in robotics and for fault detection [10]

Using Bayesian networks for control purposes is a relative new approach, so there is nearly no literature about controllers based on BNs. Welch [14] proves that BNs might be used in real time for control purposes, but he is restricted to static BNs and the choice between two possible actions.

References

1. X. Boyen and D. Koller. Approximate learning of dynamic models. In *Proc. of the 12th Annual Conference on Neural Information Processing Systems, Denver, Colorado*, 1998.
2. R. Deventer, J. Denzler, and H. Niemann. Control of Dynamic Systems Using Bayesian Networks. In Leliane Nunes de Barros et. al, editor, *Proceedings of the IBERAMIA/SBIA 2000 Workshops (Atibaia, São Paulo, Brazil)*, pages 33–39, São Paulo, November 2000. Tec Art Editora.
3. R. Deventer, J. Denzler, and H. Niemann. Non-linear modeling of a production process by hybrid Bayesian Networks. In Werner Horn, editor, *ECAI 2000 (Berlin)*, pages 576–580, Amsterdam, Berlin, Oxford, Tokyo, Washington DC, August 2000. IOS Press. ISBN: 1-58603-013-2.
4. N. Friedman, K. Murphy, and S. Russel. Learning the structure of dynamic probabilistic networks. In *12th UAI*, 1998.
5. A. Gelb, editor. *Applied optimal estimation*. MIT Press, Cambridge, Massacusetts, USA, 1994.
6. F. V. Jensen. *An introduction to Bayesian networks*. UCL Press, 1996.
7. U. Kjærulff. A computational schema for reasoning in dynamic probabilistic networks. In *8th UAI*, 1992.
8. S. L. Lauritzen. Propagation of probabilities, means, and variances in mixed graphical association models. *J. of the American Statistical Association*, 1992.
9. S. L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.
10. U. Lerner and R. Parr. Inference in hybrid networks: Theoretical limits and practical algorithms. In *17th UAI*, 2001.
11. Kevin P. Murphy. Inference and Learning in Hybrid Bayesian Networks. Technical report, University of California, Computer Science Division (EECS), January 1998.
12. Heinz Unbehauen. *Regelungstechnik II*. Vieweg, 1993.
13. Heinz Unbehauen. *Regelungstechnik I*. Vieweg, 1997.
14. R. L. Welch and C. Smith. Bayesian control for concentrating mixed nuclear waste. In *15th UAI*, 1999.