Moving Horizon Estimation for Nonlinear Networked Control Systems with Unsynchronized Timescales

Peter Philipp * **Boris Lohmann** *

* *Institute of Automatic Control, Technische Universität München, D-85748 Garching, Germany (e-mail: peter.philipp@tum.de, lohmann@tum.de)*

Abstract: This paper deals with the state and parameter estimation problem for nonlinear networked control systems. In contrast to the general assumption of synchronized clocks, the different timescale of each sensor node is explicitly considered in the proposed method. This renders conventional synchronization algorithms redundant and thus frees up the network resources involved in the synchronization. To this end, the sensor measurements are transmitted together with the corresponding time stamps as packets to the estimator via a communication network. These packets are subject to random, variable and unbounded time delays which leads to packet reordering as well as to packet drop. The presented method is capable of dealing with the aforementioned circumstances by formulating the estimation problem as a suitable optimization problem within a moving horizon framework. The resulting nonlinear program is efficiently solved by an adapted sequential quadratic programming approach which exploits the inherent structure in the problem formulation. Simulation results confirm the performance of the proposed method.

Keywords: State estimation; Parameter estimation; Communication network; Time synchronization; Nonlinear systems; Optimization problem; Nonlinear programming.

1. INTRODUCTION

Networked control systems (NCS) have received considerable attention in recent years and are now an active area of research, see e. g. the overview papers of Hespanha et al. (2007); Zampieri (2008); Yang (2006) and the references therein. The growing interest for NCS is motivated by many benefits they offer, such as their reduced installation and maintenance costs coupled with their high flexibility which offers new control oriented possibilities. However, still many problems have to be resolved before all advantages of wired and wireless NCS can be harvested. There is a need for methods that can deal with network induced imperfections and constraints, like e. g. variable sampling/transmission intervals, variable communication delays, packet reorderings, packet drops, quantization effects and unsynchronized timescales. Therefore, it is advisable to incorporate these phenomena in the design process.

Some significant progress has been reported in the area of linear estimation, analyzing the effect of packet loss, see Nahi (1969); Sinopoli et al. (2004); Jin et al. (2006); Smith and Seiler (2003), or the effect of variable random delay, see Alexander (1991); Nilsson et al. (1998); Montestruque and Antsaklis (2004). Both effects simultaneously has been addressed in Yu et al. (2004); Schenato (2008); Shi et al. (2008). However, state estimation for nonlinear NCS has not been widely investigated yet. In Jin et al. (2007), two approaches were discussed in the presence of a packet-dropping network, namely the extended Kalman filter (EKF) and the moving horizon estimation (MHE). A moving horizon observer has been designed in Philipp and Lohmann (2009), capable of dealing with variable time delays and packet drops.

In all these works, the availability of perfectly synchronized timescales is assumed, i. e. the clocks of all elements in the network are exactly synchronized. However, this task is anything but trivial and requires dedicated methods, like e. g. the famous Network Time Protocol (NTP), see Mills (1991). The higher the requirements on the synchronization precision are, the more and the more frequently packets have to be transmitted over the network solely for synchronization. This reduces both the available capacity of the network for the actual control or estimation task and the limited energy supplies of the sensor nodes. The energy cost of transmitting a packet of size 1 kB a distance of 100 m is approximately the same as that for executing 3 million instructions by a 100 million instructions per second/W processor. This justifies the need for NCS methods which do not rely on synchronized clocks and thus circumvent the aforementioned shortcomings.

In this article, an estimator is proposed for state and parameter estimation in nonlinear NCS where each sensor node possesses an unsynchronized timescale. Furthermore, the transmitted packets are subject to random, variable and unbounded time delays leading to packet reordering and packet drop. The contribution of this paper is twofold: first, it is shown how the estimation task which tackles the problem above, can be formulated as a nonlinear program (NLP) by adapting the classical MHE framework; second, it is shown how the resulting NLP can be efficiently solved by extending a previous result on structure exploiting derivative computation for MHE, see Philipp (2011b).

The remainder of this paper is organized as follows: In Section 2, the state and parameter estimation problem for nonlinear NCS is formulated. Based on the MHE framework presented in Section 3, the key goal of this paper, namely the estimator approach for nonlinear NCS is derived in Section 4. Subsequently, in Section 5, a numerical example is given to demonstrate the efficiency of the proposed method in general and especially in contrast to a conventional implemented continuous-discrete extended Kalman filter (CDEKF) before the paper is concluded in Section 6.

2. PROBLEM FORMULATION

Consider the structure of the NCS in Fig. 1. The plant consists of the nonlinear continuous-time system

$$\dot{x}(t) = f(x(t), a, u(t)) + w(t), \tag{1a}$$

where $x \in \mathbb{R}^n$ is the state vector with the initial value x_0, $a \in \mathbb{R}^q$ the parameter vector and $u \in \mathbb{R}^m$ the control vector. The vector $w \in \mathbb{R}^n$ is an additive disturbance affecting the system dynamics. The state vector is observed through the measurement equation

$$y(t) = h(x(t)) + v(t), \tag{1b}$$

where $y \in \mathbb{R}^p$ is the observation vector and $v \in \mathbb{R}^p$ is a measurement noise vector. The functions $f : \mathbb{R}^n \times \mathbb{R}^q \times \mathbb{R}^m \mapsto \mathbb{R}^n$ and $h : \mathbb{R}^n \mapsto \mathbb{R}^p$ are assumed twice differentiable. The statistics of $w(t)$ and $v(t)$ are unknown and are considered as deterministic variables of unknown character.

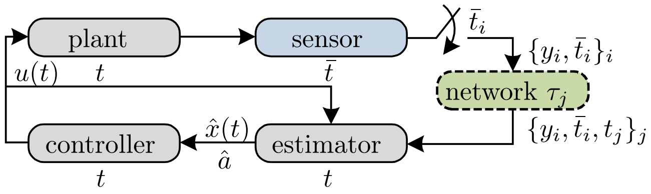

Fig. 1. Structure of the Networked Control System.

The communication network is located between the sensor and the estimator. It is assumed that this network transmits sufficiently large packets so that quantization effects can be ignored. For instance, this is appropriate for ethernet packets where the minimum size is 72 bytes while a typical data point may only consume 2 bytes. Moreover, each packet receives a random, variable and unbounded time delay τ_i, i. e. the arbitrary and possibly time-varying delay distribution is supported on the semi-infinite interval $[0, \infty)$. This can introduce packet reordering, i. e. the order of the packets are inverted in the network due to e. g. multi-path routing or parallelism at the routers and causes possible packet drops if the time delays tend to infinity.

The plant, the controller and the estimator are time synchronized and share the global timescale t, however, the sensor uses its own unsynchronized timescale $\bar{t}$. The sensor samples the output $y(t)$ non-uniformly using an event-based strategy, e. g. when some thresholds are exceeded. Subsequently, the sampled measurement y_i is transmitted through the communication network together with its relative time stamp $\bar{t}_i$ indicating when the sampling occurred as a packet of the form $\{y_i, \bar{t}_i\}$. If the packet arrives at the estimator, the packet is augmented by the arrival time stamp t_j to yield the packet $\{y_i, \bar{t}_i, t_j\}$.

The natural arising question is how to design an estimator capable of dealing with the described scenario. More precisely, the objective is to estimate, from the sequence of 3-tuples $\{y_i, \bar{t}_i, t_j\}$ in combination with the input $u(t)$, the current state vector $x(t)$ and the unknown parameter vector a.

Remark 1. For simplicity of presentation, only one sensor is considered. It is straightforward to extend the concept to the general case of several different sensors, each with different timescales $\bar{t}$.

3. MOVING HORIZON ESTIMATION

In this section, state estimation is addressed within a MHE framework for the “classical” case, i. e. without the network, without parameter estimation and with one global timescale t. For further details see e. g. Rao et al. (2003) and the references therein.

The sampled-data representation, obtained by measuring the observation vector at times t_k for $k = 0, 1, \ldots$, is derived by integrating (1) over the interval $[t_k, t_{k+1}]$

$$x_{k+1} = x_k + \int_{t_k}^{t_{k+1}} f(x(t), u(t))\, dt + w_k \tag{2a}$$

$$y_k = h(x_k) + v_k. \tag{2b}$$

The objective in the MHE framework is to derive for any $k = N, N+1, \ldots$ estimates [1] of the stacked state vector $\hat{\boldsymbol{x}}_k = [\hat{x}_{k-N}^T, \ldots, \hat{x}_k^T]^T \in \mathbb{R}^{(N+1)n}$ and the stacked state disturbance vector $\hat{\boldsymbol{w}}_k = [\hat{w}_{k-N}^T, \ldots, \hat{w}_{k-1}^T]^T \in \mathbb{R}^{Nn}$ on the basis of the information vector

$$\begin{aligned}\boldsymbol{\eta}_k &= [\ \underbrace{[y_{k-N}^T, \ldots, y_k^T]}_{\boldsymbol{y}_k^T},\ \ \underbrace{u(t)^T}_{\boldsymbol{u}_k^T}]^T, \quad t \in [t_{k-N}, t_k] \\ &= [\ \boldsymbol{y}_k^T\ ,\ \boldsymbol{u}_k^T\ ]^T \in \mathbb{R}^{(N+1)p+m},\end{aligned} \tag{3}$$

where $N+1$ measurements and the input vector are collected within a “moving horizon” (MH) interval $[k-N, k]$. Note that this general formulation allows several cases for the control vector $u(t)$, like e. g. zero-order hold discretization or even a completely continuous signal, even though the control information in the interval $t \in [t_{k-N}, t_k]$ is denoted with $\boldsymbol{u}_k$. More specifically, the estimator addresses for any $k = N, N+1, \ldots$ the minimization of the following cost function

$$J_k(\boldsymbol{\eta}_k) = \Gamma(\hat{x}_{k-N}) + \sum_{i=k-N}^{k} \Upsilon_i(\hat{v}_i) + \sum_{i=k-N}^{k-1} \Psi_i(\hat{w}_i), \tag{4}$$

where the first term, known as the arrival cost, penalizes the distance from the estimate $\hat{x}_{k-N}$ of the state at the beginning of the moving horizon to some prediction $\bar{x}_{k-N}$. This prediction $\bar{x}_{k-N}$ of the initial state is obtained by incorporating past information $y_0, \ldots, y_{k-N-1}$ and $u(t)$ in the interval $t \in [t_0, t_{k-N}[$, which is not explicitly accounted for in the objective function. The second term is a penalization of the measurement noise, whereas the third term is a penalization of the state disturbance. The

[1] Estimated values are denoted with a “ ˆ ” to distinguish them from the true values.

functions $\Gamma(\cdot)$, $\Upsilon(\cdot)$ and $\Psi(\cdot)$ are often chosen to be squared weighted L_2-norms $\|\cdot\|_M^2$, where M is a symmetric positive definite matrix.

Consequently, the NLP problem of the MHE can be derived by eliminating $\hat{v}_i$ in (4) using the measurement equation (2b)

$$\min_{\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{w}}_k} \Gamma(\hat{x}_{k-N}) + \sum_{i=k-N}^{k} \Upsilon_i(y_i, \hat{x}_i) + \sum_{i=k-N}^{k-1} \Psi_i(\hat{w}_i) \quad (5a)$$

$$\text{s.t. } \hat{x}_{i+1} - \hat{x}_i - \int_{t_i}^{t_{i+1}} f(\hat{x}(t), u(t))\, dt - \hat{w}_i = 0, \quad \forall i \in \mathcal{K} \quad (5b)$$

$$c_i(\hat{x}_i, \hat{w}_i) \geq 0, \quad \forall i \in \mathcal{K} \quad (5c)$$

$$c_k(\hat{x}_k) \geq 0, \quad (5d)$$

where $\mathcal{K} = \{k - N, \dots, k - 1\}$ is the set of indices corresponding to the current moving horizon and c_i are the inequality constraints where $\geq$ denotes componentwise inequality. The functions $c_i : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}^n, \forall i \in \mathcal{K}$ and $c_k : \mathbb{R}^n \mapsto \mathbb{R}^n$ are assumed to be twice differentiable. They allow to model bounded disturbances and physical constraints which are not implicitly enforced by the system equations due to model inaccuracies. From the solution $\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{w}}_k$ of this problem, the current state $\hat{x}_k$ can be extracted.

4. ESTIMATION STRATEGY

Before applying the described MHE framework in a suitable manner to the problem formulated in Section 2, the properties of the sensor clock have to be investigated first.

4.1 Clock Modeling

As sensor nodes are likely to be powered up at random, their clocks will have different values if queried at any given time. This difference is also known as *clock phase error*. At heart of most clocks on computing equipment is a quartz crystal oscillator. While these are more stable than most other forms of oscillators, they do vary with temperature, pressure and other subtle influences. The cheaper the crystal oscillators are, the more questionable is their precision. These facts lead to slightly different crystal oscillation causing the phase error of each individual clock to drift over time. This behavior is known as *clock frequency error*.

Low-cost sensor hardware does not typically contain such dedicated oscillators. Such devices are often only able to make use of internal processor counter to provide a software clock started at power up which has a much lower precision. To incorporate both phenomena, the following clock model is proposed

$$t = s\,\bar{t} + t_o, \quad (6)$$

where t_o and s accommodate for the clock phase error and the clock frequency error, respectively. It is reasonable to assume that s and t_o are constant for sufficiently small time intervals.

4.2 Time Delay Representation

With the assistance of this clock model, the different timescales and the impact of network effects on the transmitted packets can be illustrated in Fig. 2. A cross on the vertical upper left sensor timescale $\bar{t}$ represents the sampled measurements y_i at the time $\bar{t}_i$. This situation can be transformed in the global timescale t by applying (6) and is shown on the horizontal middle timescale t. The parameters s and t_o determine the slope and the axis intercept of the conversion line, respectively. At the time $s\,\bar{t}_i + t_o$, the corresponding packet will be sent through the network and arrives after an unknown time delay τ_j at time t_j at the estimator, provided no packet drop occurs. This τ_j represents an overall time delay caused by e. g. sensor processing and transmission through the network. The resulting situation can be visualized on the horizontal lower timescale t. Note that packet reordering has taken place, whenever two arrows intersected.

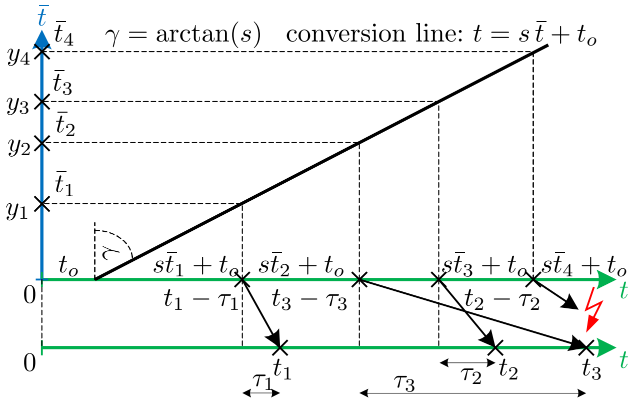

Fig. 2. Illustration of the different timescales and the impact of network effects on the transmitted packets.

Based on these considerations, all time delays can be expressed as

$$\tau_j = t_j - s\,\bar{t}_i - t_o \quad (7)$$

and depend due to the time stamps and the clock model only on the unknown parameters s and t_o.

4.3 Buffer Design

The estimator has the ability to store a fixed number of packets in a buffer which forms the information basis for the estimator. The decision logic that decides which packets to store and which to discard can be seen in Algorithm 1.

Algorithm 1 Estimator Buffer

```
Require: number of measurements in the moving horizon $N + 1$
initialize clock variable $\varphi = 0$
initialize packet counter $l = 0$
while $\varphi > 0$ do
  if $\varphi = t_j$ then
    1) Add packet $j$ to the buffer & set $l \leftarrow l + 1$
    2) Sort the packets in the buffer by the relative time stamps in ascending order
    if $l > N + 1$ then
      Discard first packet from buffer & set $l \leftarrow l - 1$
```

This buffer design has two main advantages. First, it is optimal in the sense that the latest measurements according to the actual sampling order are stored subject to the condition of limited memory buffer capacity. Second, the impact of packet reordering is countered successfully, as long as the time stamp of the latest arrived packet is not the first packet after sorting. In this case, the packet is immediately dropped and the information cannot be used by the estimator. To reduce the occurrence of this effect, the moving horizon length N can be increased.

4.4 Formulation of the MHE Optimization Problem

Based on the previous considerations, the MHE framework presented in Section 3 can now be applied in a suitable manner. The sampled-data representation of the $N+1$ measurements in the buffer can be derived by integrating (1) over the intervals $[t_j - \tau_j, t_{j+1} - \tau_{j+1}]$ or equivalently $[s\,\bar{t}_i + t_o, s\,\bar{t}_{i+1} + t_o]$

$$x_{i+1} = x_i + \int_{s\,\bar{t}_i + t_o}^{s\,\bar{t}_{i+1} + t_o} f(x(t), a, u(t))\, dt + w_i \tag{8a}$$

$$y_i = h(x_i) + v_i \tag{8b}$$

for $i = k-N, \ldots, k$, where k denotes the last packet in the sorted buffer. Regarding the NLP (5), the equality constraint (5b) describing the system dynamics is replaced by (8a). Consequently, the optimization variables, the arrival cost function and the inequality constraints are extended by $\hat{s}$, $\hat{t}_o$ and $\hat{a}$. These modifications yield

$$\min_{\substack{\hat{s}, \hat{t}_o, \hat{\boldsymbol{x}}_k \\ \hat{a}, \hat{\boldsymbol{w}}_k}} \Gamma(\hat{s}, \hat{t}_o, \hat{x}_{k-N}, \hat{a}) + \sum_{i=k-N}^{k} \Upsilon_i(y_i, \hat{x}_i) + \sum_{i=k-N}^{k-1} \Psi_i(\hat{w}_i) \tag{9a}$$

subject to

$$\hat{x}_{i+1} - \hat{x}_i - \int_{\hat{s}\,\bar{t}_i + \hat{t}_o}^{\hat{s}\,\bar{t}_{i+1} + \hat{t}_o} f(\hat{x}(t), \hat{a}, u(t))\, dt - \hat{w}_i = 0,\ \forall i \in \mathcal{K} \tag{9b}$$

$$c_i(\hat{x}_i, \hat{w}_i) \geq 0,\ \forall i \in \mathcal{K} \tag{9c}$$

$$c_k(\hat{x}_k) \geq 0 \tag{9d}$$

$$d(\hat{s}, \hat{t}_o, \hat{a}) \geq 0, \tag{9e}$$

where the inequality $d : \mathbb{R} \times \mathbb{R} \times \mathbb{R}^q \mapsto \mathbb{R}^{q+2}$ is assumed to be twice differentiable and allows to incorporate additional information about $\hat{s}$, $\hat{t}_o$ and $\hat{a}$ in the form of constraints.

4.5 Newton Type Optimization

In order to shorten the notation involved for solving the NLP (9), an overall optimization variable $\boldsymbol{p} \in \mathbb{R}^{(N+1)n+q+2}$

$$\boldsymbol{p} = [\hat{s}, \hat{t}_o, \hat{x}_{k-N}^T, \hat{a}^T, \hat{w}_{k-N}^T, \ldots, \hat{w}_{k-1}^T]^T \tag{10}$$

and an overall inequality function $\boldsymbol{c} : \mathbb{R}^{(N+1)n+q+2} \mapsto \mathbb{R}^{(N+1)n+q+2}$

$$\boldsymbol{c} = [c_{k-N}(\hat{x}_{k-N}, \hat{w}_{k-N})^T, \ldots, c_{k-1}(\hat{x}_{k-1}, \hat{w}_{k-1})^T, c_k(\hat{x}_k)^T, d(\hat{s}, \hat{t}_o, \hat{a})^T]^T \tag{11}$$

are defined. The equality constraint (9b) uniquely determines all the states $\hat{x}_i$ in the current moving horizon if the vectors $\boldsymbol{p}$ and $\boldsymbol{u}_k$ are fixed. Thus, an implicit function $\tilde{x}_i(\boldsymbol{p}, \boldsymbol{u}_k)$ that satisfies (9b) for all $\boldsymbol{p}$ and $\boldsymbol{u}_k$ by a system simulation can be defined. Consequently, the constraint (9b) can be replaced in the optimization problem by substituting the function $\tilde{x}_i(\boldsymbol{p}, \boldsymbol{u}_k)$ with $\hat{x}_i$. Hence, the NLP can be reduced to

$$\min_{\boldsymbol{p}} \Gamma(\hat{s}, \hat{t}_o, \hat{x}_{k-N}, \hat{a}) + \sum_{i=k-N}^{k} \Upsilon_i(y_i, \tilde{x}_i(\boldsymbol{p}, \boldsymbol{u}_k)) + \sum_{i=k-N}^{k-1} \Psi_i(\hat{w}_i) \tag{12a}$$

$$\text{subject to:} \quad c_i(\tilde{x}_i(\boldsymbol{p}, \boldsymbol{u}_k), \hat{w}_i) \geq 0, \quad \forall i \in \mathcal{K} \tag{12b}$$

$$c_k(\tilde{x}_k(\boldsymbol{p}, \boldsymbol{u}_k)) \geq 0 \tag{12c}$$

$$d(\hat{s}, \hat{t}_o, \hat{a}) \geq 0. \tag{12d}$$

The advantage of this approach is the strongly reduced variable space compared to the original problem. However, the computation of the involved derivatives is more costly, but there is a certain structure in the problem which will be exploited in Section 4.7.

To simplify the analysis, the arguments of all functions are in the following suppressed from the notation when the meaning is otherwise clear. The NLP (12) can be iteratively solved by applying the sequential quadratic programming (SQP) method. The basic idea of the SQP approach is to linearize in every iteration step the Karush-Kuhn-Tucker (KKT) conditions. It turns out that the resulting linear complementary system can be interpreted as the KKT conditions of the following quadratic program (QP)

$$\min_{\Delta \boldsymbol{p}} \frac{\partial \mathcal{L}}{\partial \boldsymbol{p}}^T \Delta \boldsymbol{p} + \frac{1}{2} \Delta \boldsymbol{p}^T \frac{\partial^2 \mathcal{L}}{\partial \boldsymbol{p}^2} \Delta \boldsymbol{p} \tag{13a}$$

$$\text{subject to: } \boldsymbol{c} + \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{p}} \Delta \boldsymbol{p} \geq 0. \tag{13b}$$

Thereby, $\mathcal{L}$ is the Lagrange function

$$\mathcal{L}(\boldsymbol{p}, \boldsymbol{\lambda}) = \Gamma + \sum_{i=k-N}^{k} \Upsilon_i + \sum_{i=k-N}^{k-1} \Psi_i - \boldsymbol{\lambda}^T \boldsymbol{c}, \tag{13c}$$

where $\boldsymbol{\lambda} = [\lambda_{k-N}^T, \ldots, \lambda_k^T, \mu^T]^T \in \mathbb{R}^{(N+1)n+q+2}$ is a vector of Lagrange multipliers. The accuracy as well as the solution time of the SQP method mainly depend on two circumstances: a proper initial condition of $\boldsymbol{p}$ and $\boldsymbol{\lambda}$, and an accurate and fast computation of the first and second-order derivative of the Lagrange function (13c).

4.6 Choice of the Initial Conditions

In general, the initial conditions for $\boldsymbol{p}$ and $\boldsymbol{\lambda}$ are taken from the solution of the previous NLP for a *warm start* of the optimization. For the first optimization, however, some proper initial guesses have to be provided. This is especially difficult for the clock parameters $\hat{s}$ and $\hat{t}_o$. Fortunately, these two values are, unlike all other parameters in $\boldsymbol{p}$, not required for the prediction of the current state right from the start. This property is used in Algorithm 2 to form a proper guess of s and t_o.

As soon as the $N+1$-th packet arrives at the estimator, the main part of the algorithm is executed. The time stamps in each packet of the buffer are used in association with (7) and some provided guesses of the lower and upper bound of the time delay, to form for each packet the inequality $\tau_{\min} \leq t_j - \hat{s}\,\bar{t}_i - \hat{t}_o \leq \tau_{\max}$. Note that the bounds $\tau_{\min}$ and $\tau_{\max}$ do not have to be tight or even true in order to yield some proper initial guesses for the optimization. Combing all inequalities yields the polytope $\mathcal{P}$, see Fig. 3 (left).

A reasonable choice for the initial conditions of $\hat{s}$ and $\hat{t}_o$ would be the center of $\mathcal{P}$. However, its determination involves costly numerical calculations, like e. g. finding the inequalities in $\mathcal{P}$ which are actually necessary for describing the domain of $\mathcal{P}$ and the subsequent computation of its vertexes. An alternative choice is the center Ω of an easy

Algorithm 2 Initial Conditions for $\hat{s}$ and $\hat{t}_o$

```
Require: guesses of lower and upper bound $\tau_{\min}$, $\tau_{\max}$
Require: directions of over-approximation $m_1, \dots, m_z$
1: initialize clock variable $\varphi = 0$
2: while $\varphi > 0$ do
3:   if $\varphi = t_{N+1}$ then
4:     1) Form polytope $\mathcal{P}$
         $\mathcal{P} = \left\{ \begin{bmatrix} \hat{s} \\ \hat{t}_o \end{bmatrix} \in \mathbb{R}^2 \middle| \begin{bmatrix} \bar{t}_i & 1 \\ -\bar{t}_i & -1 \end{bmatrix} \begin{bmatrix} \hat{s} \\ \hat{t}_o \end{bmatrix} \leq \begin{bmatrix} -\tau_{\min} + t_j \\ \tau_{\max} - t_j \end{bmatrix}, \; j = 1 \dots, N+1 \right\}$
5:     2) Determine bounds $b_i$ in the directions of $m_i$
         $b_i = \max_{[\hat{s}\ \hat{t}_o]^T \in \mathcal{P}} m_i^T \begin{bmatrix} \hat{s} \\ \hat{t}_o \end{bmatrix}, \quad i = 1, \dots, z$
6:     3) Calculate center $\Omega$ of over-approximation $\mathcal{A}$
         $\mathcal{A} = \left\{ \begin{bmatrix} \hat{s} \\ \hat{t}_o \end{bmatrix} \in \mathbb{R}^2 \middle| \begin{bmatrix} m_1^T \\ \vdots \\ m_z^T \end{bmatrix} \begin{bmatrix} \hat{s} \\ \hat{t}_o \end{bmatrix} \leq \begin{bmatrix} b_1 \\ \vdots \\ b_z \end{bmatrix} \right\}$
Output: Center $\Omega$ as initial conditions
```

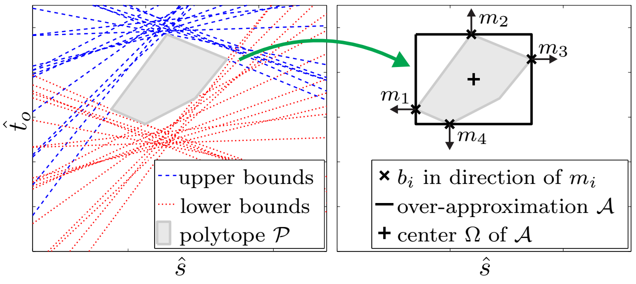
$\hat{t}_o$
upper bounds
lower bounds
▨ polytope $\mathcal{P}$
$\hat{s}$
m_2
m_3
m_1
m_4
× b_i in direction of m_i
— over-approximation $\mathcal{A}$
+ center Ω of $\mathcal{A}$
$\hat{s}$

Fig. 3. Illustration of the intersections of the half-spaces defining the polytope $\mathcal{P}$ (**left**) and the over-approximation $\mathcal{A}$ with its center Ω (**right**).

to compute over-approximation of $\mathcal{P}$, see Fig. 3 (right). To this end, some directions of the over-approximations m_i have to be provided, e. g. the unit vectors $\pm e_1$ and $\pm e_2$. The over-approximation can be formulated as the polytope $\mathcal{A}$ (see line 6 of Alg. 2) with the elements b_i resulting from the solution of the linear programs defined in line 5.

4.7 Derivative Computation

The second part that significantly contributes to the efficient solution of the NLP (9) is the derivative computation of the Lagrange function (13c). A common method to compute the gradient of $\mathcal{L}$ is by finite differences. For instance, the elements of $\partial\mathcal{L}/\partial\boldsymbol{p}$ can be approximated by the central-difference formula $\partial\mathcal{L}/\partial p_i \approx (\mathcal{L}(\boldsymbol{p} + \epsilon\, e_i) - \mathcal{L}(\boldsymbol{p} - \epsilon\, e_i))/(2\epsilon)$ for $i = 1, \dots, (N+1)n + q + 2$, where ϵ is a small positive scalar and e_i is the i-th unit vector. However, it is strongly advised not to use this method here due to the high numerical complexity and the poor achieved accuracy. The evaluation is as costly as solving $2n[(N+1)n + q + 2]$ ODEs over the complete moving horizon interval $[\hat{s}\,\bar{t}_{k-N} + \hat{t}_o, \hat{s}\,\bar{t}_k + \hat{t}_o]$.

Alternatively, the exact derivatives of (13c) can be calculated by applying the chain rule whereby the dependence of $\tilde{x}_i$ on $\boldsymbol{p}$ has to be considered. This leads to the exact gradient

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{p}} = \frac{\partial\Gamma}{\partial\boldsymbol{p}} + \sum_{i=k-N}^{k} \frac{\partial\tilde{x}_i}{\partial\boldsymbol{p}}^T \frac{\partial\Upsilon_i}{\partial\tilde{x}_i} + \sum_{i=k-N}^{k-1} \frac{\partial\Psi_i}{\partial\boldsymbol{p}} - \sum_{i=k-N}^{k} \frac{\partial\tilde{x}_i}{\partial\boldsymbol{p}}^T \frac{\partial c_i}{\partial\tilde{x}_i}^T \lambda_i - \frac{\partial d}{\partial\boldsymbol{p}}^T \mu \tag{14}$$

where

$$\frac{\partial\tilde{x}_i}{\partial\boldsymbol{p}} = \left[\frac{\partial\tilde{x}_i}{\partial\hat{s}}, \frac{\partial\tilde{x}_i}{\partial\hat{t}_o}, \frac{\partial\tilde{x}_i}{\partial\hat{x}_{k-N}}, \frac{\partial\tilde{x}_i}{\partial\hat{a}}, \frac{\partial\tilde{x}_i}{\partial\hat{w}_{k-N}}, \dots, \frac{\partial\tilde{x}_i}{\partial\hat{w}_{k-1}} \right] \tag{15}$$

are for $i = k-N, \dots, k$ the first-order sensitivities of $\tilde{x}_i$ with respect to $\boldsymbol{p}$ evaluated at the corresponding sampling times on the moving horizon interval.

In the following, the approach developed in Philipp (2011b) for computing these sensitivities is extended. It is assumed that $w(t)$ is discretized using zero-order hold leading to a piecewise constant estimation $\hat{w}(t) = \hat{w}_j/(\hat{s}\,T_j)$, $\hat{s}\,\bar{t}_j + \hat{t}_o < t \leq \hat{s}\,\bar{t}_{j+1} + \hat{t}_o$, where $T_j = \bar{t}_{j+1} - \bar{t}_j$. Note that this assumption is not necessary but it does simplify the description to follow.

Notation 1. The abbreviation X_β^α, Y_β^α and ${}^jZ_\beta^\alpha$ denotes the solution of the corresponding first-order sensitivity ODE $\dot{X} = \frac{\partial f}{\partial\tilde{x}} X$, $\dot{Y} = \frac{\partial f}{\partial\tilde{x}} Y + \frac{\partial f}{\partial\hat{a}}$ and ${}^j\dot{Z} = \frac{\partial f}{\partial\tilde{x}} {}^jZ + \frac{1}{\hat{s}\,T_j} I$ at the time $\hat{s}\,t_\beta + \hat{t}_o$ with the initial value I, 0 and 0 at the initial time $\hat{s}\,t_\alpha + \hat{t}_o$, respectively.

Theorem 1. The first-order sensitivities defined in (15) are for $i = k-N+1, \dots, k$

$$\begin{aligned} \frac{\partial\tilde{x}_i}{\partial\hat{s}} &= \bar{t}_i f_i - \bar{t}_{k-N} X_i^{i-1} X_{i-1}^{i-2} \dots X_{k-N+1}^{k-N} f_{k-N} \\ &\quad + \sum_{j=k-N}^{i-1} X_i^{i-1} X_{i-1}^{i-2} \dots X_{j+2}^{j+1} \left(\bar{t}_{j+1} I - \bar{t}_j X_{j+1}^j \right) \frac{\hat{w}_j}{\hat{s}\,T_j} \\ &\quad - \frac{1}{\hat{s}} \sum_{j=k-N}^{i-1} X_i^{i-1} X_{i-1}^{i-2} \dots X_{j+2}^{j+1}\, {}^jZ_{j+1}^j\, \hat{w}_j \end{aligned} \tag{16a}$$

$$\begin{aligned} \frac{\partial\tilde{x}_i}{\partial\hat{t}_o} &= f_i - X_i^{i-1} X_{i-1}^{i-2} \dots X_{k-N+1}^{k-N} f_{k-N} \\ &\quad + \sum_{j=k-N}^{i-1} X_i^{i-1} X_{i-1}^{i-2} \dots X_{j+2}^{j+1} \left(I - X_{j+1}^j \right) \frac{\hat{w}_j}{\hat{s}\,T_j} \end{aligned} \tag{16b}$$

$$\frac{\partial\tilde{x}_i}{\partial\hat{x}_{k-N}} = X_i^{i-1} X_{i-1}^{i-2} \dots X_{k-N+2}^{k-N+1} X_{k-N+1}^{k-N} \tag{16c}$$

$$\frac{\partial\tilde{x}_i}{\partial\hat{a}} = Y_i^{k-N} \tag{16d}$$

$$\frac{\partial\tilde{x}_i}{\partial\hat{w}_l} = \begin{cases} 0, & i < l+1 \\ X_i^{i-1} X_{i-1}^{i-2} \dots X_{l+2}^{l+1}\, {}^lZ_{j+1}^l, & i \geq l+1 \end{cases}, \quad l = k-N, \dots, k-1 \tag{16e}$$

and for $i = k-N$ all identical to 0, except $\partial\tilde{x}_{k-N}/\partial\hat{x}_{k-N}$ which is identity I.

Proof. For the proof of (16a)-(16b) see Philipp (2011a). The proof of (16c)-(16e) results from Philipp (2011b) and is omitted here due to space limitations. ■

The advantage of this theorem over the finite difference method is two-fold. First, the number of ODEs that have to be solved over the complete interval $[\hat{s}\,\bar{t}_{k-N} + \hat{t}_o, \hat{s}\,\bar{t}_k + \hat{t}_o]$ is independent of the horizon length N and thus independent of the number of unknowns, namely $n(2n + 1 + q)$. Second, these ODEs are independent of each other and can thus be solved in parallel.

The finite differences method can once again be applied to approximate the Hessian of $\mathcal{L}$. However, the computational complexity is even higher than in the former case. An alternative and more commonly used method is to estimate the Hessian by applying a quasi-Newton approximation which measures only the changes in the gradients, see Nocedal and Wright (2000). The drawback of this approximation method is that it does not explicitly consider the true structure of the Hessian which may result in significant deviations from the real one. As a consequence, the quality of the iterations can be poor, and many costly steps are required to converge to a local optimum. For this reason, the following approach is proposed to improve the convergence speed by exploiting the structure of the Hessian.

The exact Hessian of $\mathcal{L}$ is derived from (14) by applying the chain rule and partitioned into two parts $\partial^2\mathcal{L}/\partial\boldsymbol{p}^2 = H_1 + H_2$. All terms that do not involve the second-order sensitivities are collected in H_1:

$$H_1 = \frac{\partial^2\Gamma}{\partial\boldsymbol{p}^2} + \sum_{i=k-N}^{k} \frac{\partial\tilde{x}_i}{\partial\boldsymbol{p}}^T \frac{\partial^2\Upsilon_i}{\partial\tilde{x}_i^2} \frac{\partial\tilde{x}_i}{\partial\boldsymbol{p}} + \sum_{i=k-N}^{k-1} \frac{\partial^2\Psi_i}{\partial\boldsymbol{p}^2}$$
$$- \sum_{i=k-N}^{k} \sum_{j=1}^{n} \lambda_{i,j} \frac{\partial\tilde{x}_i}{\partial\boldsymbol{p}}^T \frac{\partial^2 c_{i,j}}{\partial\tilde{x}_i^2} \frac{\partial\tilde{x}_i}{\partial\boldsymbol{p}} - \sum_{j=1}^{q+2} \mu_j \frac{\partial^2 d_j}{\partial\boldsymbol{p}^2}, \quad (17a)$$

where μ_j, d_j, $c_{i,j}$ and $\lambda_{i,j}$ denote the j-th element of μ, d, c_i and λ_i, respectively. Consequently, H_2 contains only the terms involving the second-order sensitivities $\partial^2\tilde{x}_{i,j}/\partial\boldsymbol{p}^2$ of the j-th element of $\tilde{x}_i$:

$$H_2 = \sum_{i=k-N}^{k} \sum_{j=1}^{n} \left(\frac{\partial\Upsilon_i}{\partial\tilde{x}_{i,j}} - \left(\lambda_i^T \frac{\partial c_i}{\partial\tilde{x}_{i,j}} \right) \right) \frac{\partial^2\tilde{x}_{i,j}}{\partial\boldsymbol{p}^2}. \quad (17b)$$

H_1 can be easily calculated due to the already available first-order sensitivities that were necessary for the computation of $\partial\mathcal{L}/\partial\boldsymbol{p}$. Depending on the nonlinearities of the system, H_2 can be either neglected or approximated by a BFGS update strategy. For further details see Philipp (2011b).

4.8 NCS Moving Horizon Estimator Algorithm

The main parts of the overall NCS moving horizon estimator (NCS-MHE) are summarized in Algorithm 3 (details are omitted). After the SQP algorithm which is initialized in line 14 is terminated, the estimated states are updated in line 19 leading to a possible jump in the states. The prediction in line 20 is performed by solving the system equations forward in time.

5. NUMERICAL CASE STUDY

In this section, the NCS-MHE and a conventional CDEKF, see appendix A, are applied to a nonlinear continuously stirred tank reactor (CSTR), see Hicks and Ray (1971); Uppal et al. (1974), described by

$$\dot{x}_1(t) = p_1(x_a - x_1(t)) - p_2 x_1(t) \exp\left(-\frac{E_A}{R\,x_2(t)}\right) + w_1(t)$$
$$\dot{x}_2(t) = p_1(x_b - x_2(t)) + p_2 p_3 x_1(t) \exp\left(-\frac{E_A}{R\,x_2(t)}\right)$$
$$+ p_4(u(t) - x_2(t)) + w_2(t)$$

Algorithm 3 NCS Moving Horizon Estimator

```
Require: number of measurements in the moving horizon
    $N + 1$
Require: guesses of lower and upper bound $\tau_{\min}$, $\tau_{\max}$
Require: directions of over-approximation $m_1, \dots, m_z$
 1: initialize clock variable $\varphi = 0$
 2: initialize packet counter $l = 0$
 3: while $\varphi > 0$ do
 4:   if $\varphi = t_j$ then
 5:     1) Add packet $j$ to the buffer & set $l \leftarrow l + 1$
 6:     2) Sort the packets in the buffer by the relative
        time stamps in ascending order
 7:     if $l > N + 1$ then
 8:       Discard first packet from buffer & set $l \leftarrow l - 1$
 9:     if $\varphi = t_{N+1}$ then
10:       1) Form polytope $\mathcal{P}$
11:       2) Determine bounds $b_i$ in the directions of $m_i$
12:       3) Calculate center $\Omega$ of over-approximation $\mathcal{A}$
13:     if $l > N$ & packet $j \in$ buffer then
14:       while termination conditions = false do
15:         1) Calculate $\hat{x}_i$ for the last $N + 1$ packets in
            the buffer
16:         2) Calculate $\mathcal{L}$, $\partial\mathcal{L}/\partial\boldsymbol{p}$, $\partial^2\mathcal{L}/\partial\boldsymbol{p}^2$
17:         3) Solve quadratic program
18:         4) Update $\boldsymbol{p}$, $\boldsymbol{\lambda}$
19:       Update $\hat{x}(\varphi)$, $\hat{a}$, $\hat{s}$ and $\hat{t}_o$
20:   Predict $\hat{x}(\varphi)$
```

The system involves two states $x = [x_1, x_2]^T$, corresponding to the concentration and the temperature, respectively, one control u corresponding to the cooling water temperature and two process noise sequences $w = [w_1, w_2]^T$. The initial condition is $x_0 = [0.5, 19.6]^T$ and the model parameters are $E_A = 10$, $R = 1$, $x_a = 2$, $x_b = 1$, $p_1 = 1$, $p_2 = 5$, $p_3 = 1$ and $p_4 = 1$. The sensor node which measures the temperature shall be able to move freely in the surrounding reaction mixture. This node possesses an unsynchronized clock in compliance with the clock model (6) where the parameters are set to $s = 1.1$ and $t_o = 15s$. A packet is transmitted, consisting of a measurement and the corresponding relative time stamp, if one of the following conditions are true based on the information in latest transmitted packet: the measured temperature changes about 0.5 or $0.9s$ are elapsed according to the clock of the sensor node. The measurements are generated from a simulated closed-loop feedback control scenario with Gaussian process noise sequences with standard deviations $\sigma_{w_1} = 0.2828$ and $\sigma_{w_2} = 2$. Afterwards, the resulting temperatures are corrupted with different levels of Gaussian noise with standard deviation $\sigma_y = 1$ to simulate measurement noise. The objective is to estimate the current states of the CSTR.

To this end, the NCS-MHE uses the following cost function

$$J_k = \|\boldsymbol{\varrho} - \bar{\boldsymbol{\varrho}}\|_P^2 + \sum_{i=k-N}^{k} \|y_i - \hat{y}_i\|_R^2 + \sum_{i=k-N}^{k-1} \|\hat{w}_i\|_Q^2$$

where $\boldsymbol{\varrho} = [\hat{s}, \hat{t}_o, \hat{x}_{k-N}^T]^T$ collects some optimization variables and $\bar{\boldsymbol{\varrho}} = [\bar{s}, \bar{t}_o, \bar{x}_{k-N}^T]^T$ are the corresponding predictions resulting from the optimal solution of the preceding horizon. The weighting matrices are $P = \mathrm{diag}(10, 10, 1)$, $R = 1/\sigma_y^2$ and $Q = \mathrm{diag}(1/\sigma_{w_1}^2, 1/\sigma_{w_2}^2)$ and the number of measurements in the horizon is $N + 1 = 8$. The optimization variable $\boldsymbol{p}$ is initialized with $[*, *, 1.5, 5, 0, \dots, 0]^T$. Thereby, the two stars indicate the application of Algorithm 2 leading to the automatically generated initial conditions $\hat{s} = 1.104$ and $\hat{t}_o = 13.695$ where $\tau_{\min} = 0$ and $\tau_{\max} = 5$ where used. Note that precisely this scenario can be seen in Fig. 3.

The weighting matrices and the initial condition of the CDEKF are identical to the NCS-MHE. The CDEKF checks whenever a new packet arrives if its relative time stamp is newer than the one used for the last update step. If this is true, an update step (A.3) is performed where the arrival time is used as the time where the corresponding sampling occurred, i. e. the time delays are neglected. Otherwise the packet will be dropped and the prediction according to (A.2) proceeds.

In Fig. 4, the performance of the NCS-MHE and the CDEKF is presented. Both methods perform identical until the first packet arrives. In contrast to the NCS-MHE which has to wait until the 8th packet arrives, the CDEKF already performs at this point its first update step. The corresponding updates result for both methods in a jump in the estimated states. The estimation quality of the NCS-MHE is quite satisfying in general and especially good compared to the CDEKF. It only performs reasonable in regions where $y(t_i) = y(t_i - \tau_i)$, i. e. when the system dynamics are slow compared to the time delays, like e. g. in the interval $t \in [7s, 10s]$. Otherwise, the time shift of the estimated states caused by the time delays can be seen whenever an update step occurs. The numerical complexity for the NCS-MHE and the CDEKF consists in solving ten (per iteration) and six ODEs, respectively. The NCS-MHE requires three iterations in the first and only one in the later moving horizons.

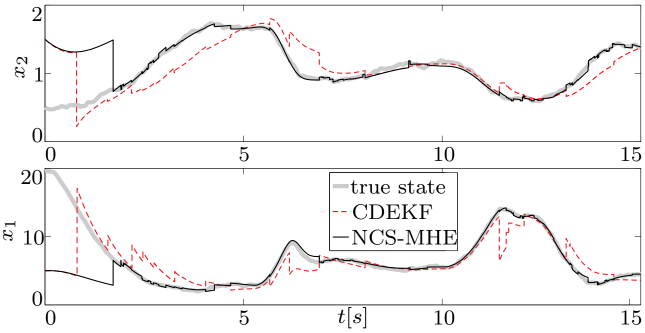

Fig. 4. Comparison between the true and estimated states of the CDEKF and NCS-MHE.

In the upper plot of Fig. 5, the above detailed sampling strategy for the measured temperature can be seen in the interval $[0.5s, 4s]$. A cross or a circle represents a sampling instance whereas the former indicates a successful transmission of the corresponding packet and the latter implies a packet drop. The sampling times of these measurements can also be seen in the lower plot of Fig. 5 which are equivalent to the sending time points. They are located on the vertical axis with value 1 and are connected by a solid line to the arrival time points at the estimator which are on the vertical axis with value 0. Packet reordering has taken place whenever two lines intersected. Note that the aforementioned jumping behavior of the estimated states can be observed more clearly in the upper plot of Fig. 5.

In the upper plot of Fig. 6, the time delay distribution is specified. The packet drop rate is 14.14%. The performance of the NCS-MHE regarding the estimation of the time delays is presented in the lower plot of Fig. 6. The relative errors of the estimated parameters $\hat{s}$ and $\hat{t}_o$ are below 2% leading to a good estimation quality of the time delays.

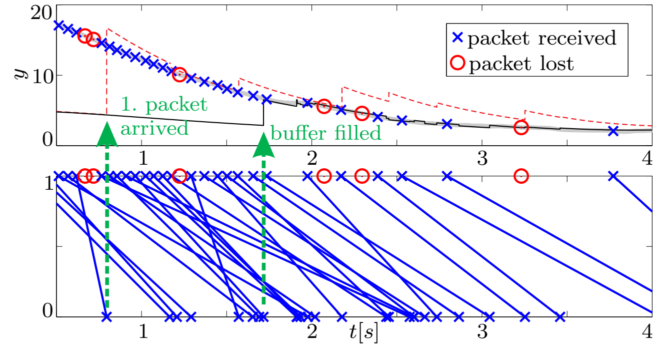

Fig. 5. **Top**: Sampling instances of the measured temperature. **Bottom**: Run of the packets through the network.

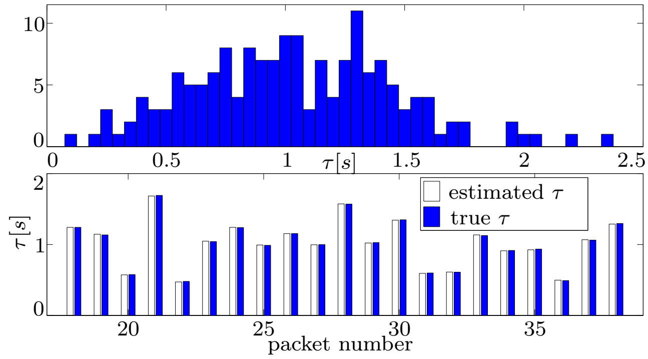

Fig. 6. **Top**: Illustration of the time delay distribution. **Bottom**: Comparison of the true and estimated time delays.

6. CONCLUSION AND FUTURE WORK

In this paper, the state and parameter estimation problem for nonlinear networked control systems is considered. Based on a suitable buffer design, a moving horizon estimator is presented which is capable of dealing with the network induced imperfections such as unsynchronized clocks and unbounded variable time delays of the transmitted packets. The resulting nonlinear program is efficiently solved by a sequential quadratic programming approach which exploits the structure in the problem formulation as wells as in the derivatives.

Future work is directed on the one hand towards deriving stability and observability properties of the proposed estimator and on the other hand to conduct experiments using a test-rig to demonstrate the results of this paper.

ACKNOWLEDGEMENTS

The authors thank the German Research Foundation (DFG), Priority Program 1305 for supporting this work.

REFERENCES

Alexander, H. (1991). State estimation for distributed systems with sensing delay. In *Proc. SPIE: Data Structures and Target Classification*, volume 1470, 103–111.

Gelb, A. (1974). *Applied Optimal Estimation.* The MIT Press.

Hespanha, J.P., Naghshtabrizi, P., and Xu, Y. (2007). A survey of recent results in networked control systems. In *Proc. IEEE*, volume 95, 138–162.

Hicks, G. and Ray, W. (1971). Approximation methods for optimal control synthesis. *Can. J. Chem. Eng.*, 49, 552–529.

Jin, Z., Gupta, V., and Murray, R. (2006). State estimation over packet dropping networks using multiple description coding. *Automatica*, 42(9), 1441–1452.

Jin, Z., Ko, C.K., and Murray, R. (2007). Estimation for nonlinear dynamical systems over packet-dropping networks. In *Proc. of the American Control Conference.*

Mills, D.L. (1991). Internet time synchronisation: the network time protocol. *IEEE Trans. Commun.*, 39(10), 1482–1493.

Montestruque, L. and Antsaklis, P. (2004). Stability of model-based networked control systems with time-varying transmission times. *IEEE Trans. Autom. Control*, 49(9), 1562–1572.

Nahi, N. (1969). Optimal recursive estimation with uncertain observation. *IEEE Trans. Inf. Theory*, 15(4), 457–462.

Nilsson, J., Bernhardsson, B., and Wittenmark, B. (1998). Stochastic analysis and control of real-time systems with random delay. *Automatica*, 34, 57–64.

Nocedal, J. and Wright, S. (2000). *Numerical Optimization.* Springer-Verlag.

Philipp, P. (2011a). Exact state sensitivity calculation and its role in moving horizon estimation for nonlinear networked control systems. In G. Roppenecker and B. Lohmann (eds.), *Methoden und Anwendung der Regelungstechnik (to appear).* Shaker-Verlag, Aachen.

Philipp, P. (2011b). Structure exploiting derivative computation for moving horizon estimation. In *Accepted and to be presented at the American Control Conference.*

Philipp, P. and Lohmann, B. (2009). Gauss-newton moving horizon observer for state and parameter estimation of nonlinear networked control systems. In *Proc. of the European Control Conference*, 1740–1745.

Rao, C., Rawlings, J., and Mayne, D. (2003). Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations. *IEEE Trans. Autom. Control*, 48(2), 246–258.

Schenato, L. (2008). Optimal estimation in networked control systems subject to random delay and packet drop. *IEEE Trans. Autom. Control*, 53(5), 1311–1317.

Shi, L., Johansson, K.H., and Murray, R.M. (2008). Estimation over wireless sensor networks: Tradeoff between communication, computation and estimation qualities. In *Proc. of the 17th IFAC World Congress*, 605–611.

Sinopoli, B., Schenato, L., Franceschetti, M., Poolla, K., Jordan, M., and Sastry, S. (2004). Kalman filtering with intermittent observations. *IEEE Trans. Autom. Control*, 49(9), 1453–1468.

Smith, S.C. and Seiler, P. (2003). Estimation with lossy measurements: Jump estimatiors for jump systems. *IEEE Trans. Autom. Control*, 48(12), 2163–2171.

Uppal, A., Ray, W., and Poore, A. (1974). On the dynamic behavior of continuous stirred tank reactors. *Chem. Eng. Sci.*, 29, 967–985.

Yang, T.C. (2006). Networked control systems: a brief survey. In *IEE Proc. - Control Theory Appl.*, volume 153, 403–412.

Yu, N., Wang, L., Chu, T., and Hao, F. (2004). An lmi approach to networked control systems with data packet dropout and transmission delay. *J. Hybrid Syst.*, 3(11).

Zampieri, S. (2008). Trends in networked control systems. In *Proc. of the 17th IFAC World Congress*, 2886–2894.

Appendix A. CONTINUOUS-DISCRETE EXTENDED KALMAN FILTER

Consider the continuous nonlinear system

$$\dot{x}(t) = f(x(t), u(t)) + w(t) \tag{A.1a}$$

where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^m$ is the control vector and $w \in \mathbb{R}^n$ is the normally distributed process noise vector, i. e. $w(t) \sim \mathcal{N}(0, Q(t))$. The state vector is observed through the discrete measurement equation

$$y_k = h(x(t_k)) + v_k, \quad k = 1, 2, \dots \tag{A.1b}$$

where $y_k \in \mathbb{R}^p$ is the observation vector and $v_k \in \mathbb{R}^p$ is the normally distributed measurement noise vector, i. e. $v_k \sim \mathcal{N}(0, R_k)$.

The mean and the covariance of the initial state are assumed to be $\hat{x}_0$ and P_0, respectively, i. e. $x(t_0) \sim \mathcal{N}(\hat{x}_0, P_0)$. Then the continuous-discrete extended Kalman filter (CDEKF) for prediction and updating may be stated as follows, see Gelb (1974), where the upper indices $^-$ and $^+$ denotes the times immediately *before* and immediately *after* a discrete measurement.

Prediction:

The one-step ahead predictions for the state $\hat{x}_k^- = \hat{x}(t_k)$ and its covariance $\hat{P}_k^- = \hat{P}(t_k)$ are computed as the solution to the system of differential equations

$$\dot{\hat{x}}(t) = f(\hat{x}(t), u(t)) \tag{A.2a}$$

$$\dot{P}(t) = F(t)P(t) + P(t)F(t)^T + Q(t), \tag{A.2b}$$

in which

$$F(t) = \left.\frac{\partial f}{\partial x}\right|_{x(t)=\hat{x}(t)} \tag{A.2c}$$

with the initial conditions $\hat{x}(t_{k-1}) = \hat{x}_{k-1}^-$ and $\hat{P}(t_{k-1}) = \hat{P}_{k-1}^-$.

Updating:

Utilizing the gain matrix

$$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k\right)^{-1}, \tag{A.3a}$$

the filtered state $\hat{x}_k^+$ and its covariance P_k^+ are calculated by

$$\hat{x}_k^+ = \hat{x}_k^- + K_k \left(y_k - h(\hat{x}_k^-)\right) \tag{A.3b}$$

$$P_k^+ = (I - K_k H_k) P_k^- \tag{A.3c}$$

where

$$H_k = \left.\frac{\partial h}{\partial x}\right|_{x(t_k)=\hat{x}_k^-}. \tag{A.3d}$$