Dipl.-Ing. Jakob Ludwiger

Discrete Time Sliding Mode Controller Design for Networked Control Systems

Doctoral Thesis

to achieve the university degree of
Doktor der technischen Wissenschaften

submitted to
Graz University of Technology

Supervisor
Univ.-Prof. Dipl-Ing. Dr.techn. Martin Horn

Co-Supervisor
Ass.Prof. Dipl.-Ing. Dr.techn. Martin Steinberger

Institute of Automation and Control

Faculty of Electrical and Information Engineering

Graz, June 2020

Institute of Automation and Control
Graz University of Technology
Inffeldgasse 21/B
8010 Graz, Austria
https://www.tugraz.at/institute/irt

To Vanessa, my parents and my brother.

"Ich weiß wohl, daß man dem das Mögliche nicht dankt,
von dem man das Unmögliche gefordert hat."

Johann Wolfgang von Goethe

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master‘s thesis.

Date	Signature

Abstract

Due to increased flexibility and adaptability, the feedback channel of modern control systems is often closed through a communication network. In general, this communication channel is affected by network imperfections such as variable transmission delays which negatively influence the performance of the closed loop system and can even lead to instability. Hence, these time delays have to be considered in the controller design. In addition, real world control systems are always affected by disturbances which trigger the demand for robust control algorithms such as sliding mode based control algorithms. This dissertation aims to combine sliding mode control and networked control for multi-input linear time-invariant plants, specifically focusing on variable time delays.

For multi-input systems it is possible to either implement the control law in a centralized fashion, i.e. one controller provides the control signals for all input channels, or spatially distributed, i.e. separate controllers provide the control signal for the respective input channel.

Using the centralized topology, a specific structure of the sliding variable is required, which makes classical reaching laws not applicable. Therefore, three sliding mode based reaching laws are proposed for centralized networked control systems. Simulation results demonstrate the effectiveness of the proposed approaches.

The design for the spatially distributed topology is handled by using integral sliding mode control with a specific structure of the sliding variable. This method makes it possible to cast the problem in a form where classical sliding mode based control algorithms are applicable. A nominal controller design based on linear matrix inequalities offers the possibility to achieve the structure required for this part of the control law. Simulation results and experiments using a mechanical real-world system show the effectiveness of the proposed approaches.

Kurzfassung

Aufgrund der höheren Flexibilität und Anpassungsfähigkeit wird in modernen Regelungssystemen der Rückkoppelzweig oft über ein Kommunikationsnetzwerk geschlossen. Im Allgemeinen sind diese Kommunikationskanäle nicht ideal, wodurch es etwa zu variablen Totzeiten bei der Übertragung kommt. Diese Totzeiten verschlechtern das Betriebsverhalten und können bis zur Instabilität des Regelkreises führen, falls die Totzeiten nicht während des Reglerentwurfs berücksichtigt werden. Darüber hinaus treten bei realen Anwendungen immer Störeinflüsse auf, weshalb robuste Regelalgorithmen wie sliding-mode basierte Algorithmen entwickelt wurden. Diese Dissertation verfolgt das Ziel, die sliding-mode Regelung mit der Netzwerkregelung für zeitinvariante Mehrgrößensysteme zu kombinieren. Spezieller Fokus wird dabei auf die Berücksichtigung variabler Totzeiten gelegt.

Für Mehrgrößensysteme ist es möglich, das Regelgesetz entweder zentral, d.h. ein zentraler Regler liefert die Stellsignale für alle Eingangskanäle, oder örtlich verteilt, d.h. separate Regler liefern das Stellsignal zum jeweiligen Eingangskanal, zu implementieren.

Die zentrale Topologie erfordert eine spezielle Struktur der sliding Variable, wodurch klassische reaching laws nicht anwendbar sind. Daher werden in dieser Dissertation drei sliding-mode basierte reaching laws für zentrale Netzwerkregelungen vorgestellt. Die Wirksamkeit der vorgestellten Algorithmen wird auf Basis von Simulationsergebnissen gezeigt.

Für die örtlich verteilten Topologien wird ein Entwurf vorgestellt, der auf integral sliding-mode basiert, wobei eine spezielle Struktur für die sliding Variable notwendig ist. Diese Methode ermöglicht es, das Problem in eine Form zu bringen, in der klassische sliding-mode-Regler anwendbar sind. Da auch für den nominellen Teil des Regelgesetzes eine spezielle Struktur notwendig ist, wird ein Entwurf auf Basis linearer Matrixungleichungen vorgestellt. Die Funktionsweise der vorgestellten Algorithmen wird mittels Simulationsergebnissen und Experimenten mit einem mechanischen Labormodell gezeigt.

Notation

The natural numbers excluding zero are represented by $\mathbb{N}$, the natural numbers including zero by $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$ and the integers by $\mathbb{Z}$. The real numbers are denoted as $\mathbb{R}$ and the positive real numbers including zero as $\mathbb{R}^+$. A complex number $c \in \mathbb{C}$ is either represented by its real part a and imaginary part b, i.e. $c = a + bi$, or by its magnitude $|c|$ and phase $\angle c$, i.e. $c = |c|\, e^{i\angle c}$.

The operator $b = \lceil a \rceil$ yields the smallest integer value $b \in \mathbb{Z}$ larger than or equal to $a \in \mathbb{R}$.

A matrix

$$\boldsymbol{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,\beta} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,\beta} \\ \vdots & \vdots & \ddots & \vdots \\ a_{\alpha,1} & a_{\alpha,2} & \cdots & a_{\alpha,\beta} \end{bmatrix}$$

with the entries $a_{i,j} \in \mathbb{R}$, $i = 1, 2, \cdots, \alpha$ and $j = 1, 2, \cdots, \beta$ is defined by $\boldsymbol{A} \in \mathbb{R}^{\alpha\times\beta}$. The unity matrix of dimension α is represented by $\boldsymbol{I}_\alpha$. If the absolute operator is applied to a matrix $\boldsymbol{A} \in \mathbb{R}^{\alpha\times\beta}$, the absolute operator is applied to each element of $\boldsymbol{A}$, i.e.

$$|\boldsymbol{A}| = \begin{bmatrix} |a_{1,1}| & |a_{1,2}| & \cdots & |a_{1,\beta}| \\ |a_{2,1}| & |a_{2,2}| & \cdots & |a_{2,\beta}| \\ \vdots & \vdots & \ddots & \vdots \\ |a_{\alpha,1}| & |a_{\alpha,2}| & \cdots & |a_{\alpha,\beta}| \end{bmatrix}.$$

A symmetric square matrix $\boldsymbol{A} \in \mathbb{R}^{\alpha\times\alpha}$ is said to be positive definite, i.e. $\boldsymbol{A} \succ 0$ if, and only if, $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} > 0$ is satisfied $\forall \boldsymbol{x} \in \mathbb{R}^\alpha$ with $\boldsymbol{x} \neq \boldsymbol{0}$.

Contents

Chapter 1

Introduction

Contents

Modern control systems are often characterized by a close interaction between numerous sensors, actuators and controllers. Today's cars, for instance, use more than 100 control devices and several thousand sensors and actuators. One can imagine that enabling these devices to share information with one another is a challenging task. A point-to-point connection would not be feasible even though not every device has to communicate with all other devices.

This increased complexity triggers the need to develop a new control system architecture where communication links are provided by wired or wireless network technologies. These architectures are called networked control systems (NCS) in literature. Figure 1.1 shows a very general block diagram of a networked control system. The p sensors transmit their data through a network to the q controllers. These controllers evaluate their control law and transmit the result to the m actuators.

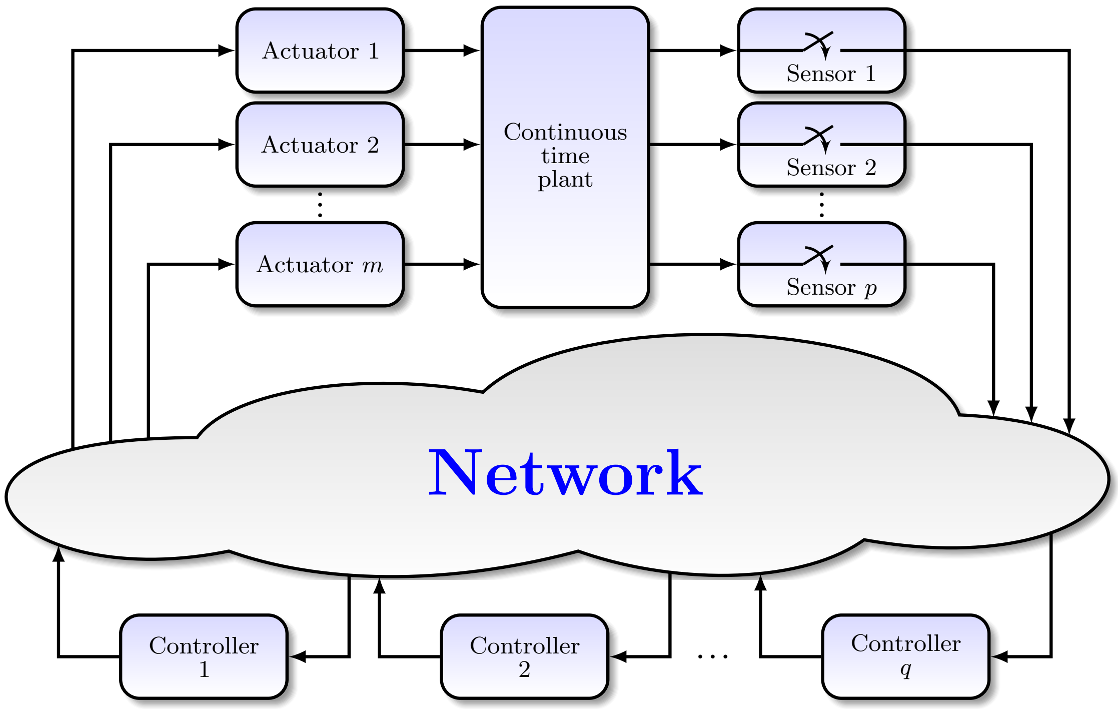


Figure 1.1: Architecture of a networked control system.

This networked architecture has several advantages compared to conventional control systems. The wiring effort is significantly reduced not only for large scale systems with numerous sensors and actuators but also for applications where large spatial distances have to be overcome. In addition, this architecture is characterized by a very high flexibility and adaptability since it offers the possibility to replace components or even add additional ones without changing the architecture.

Nevertheless, this networked architecture poses additional challenges in the design of the control algorithms due to imperfections of the network. One network imperfection is data loss. This means that a transmitted packet might be lost while being transmitted to the receiver. Some network protocols like the Transmission Control Protocol (TCP) [1] ensure a reliable transmission. Using this protocol, data losses are detected and retransmissions are triggered until the data is received. However, protocols like the User Datagram Protocol (UDP) [2] might be chosen for networked control systems due to the reduced overhead. This protocol does not ensure a reliable transmission and therefore packets might be lost. There are several scientific publications regarding this type of imperfection (see e.g. [3, 4, 5]).

Another network imperfection is the time delay induced by the limited transmission speed of the network. Depending on the network structure and used protocol, this time delays may be time-varying. There are some approaches allowing the design of control laws that guarantee stability of networked control systems affected by time-varying delay using Lyapunov-Krasovskii analysis (see e.g. [6, 7]). Other approaches are based on the prediction of future state variables and achieve the compensation of communication delays and data losses (see e.g. [8, 9]). Some approaches make use of event-triggered control to reduce the network load, which should indirectly lead to smaller time delays and fewer packet losses (see e.g. [10, 11]). Overviews of existing techniques for networked control are given in [12, 13].

In addition to the network imperfections, every control system implemented in the real world is affected by perturbations. Therefore, it is necessary to consider known properties of these perturbations (e.g. the highest absolute value or the highest change rate) in the controller design to achieve robust control systems. This robustness can be achieved by applying so-called sliding mode techniques. Depending on the algorithm used, they are characterized by the advantages of finite time convergence, theoretically exact compensation for a wide class of perturbations and simple parameter tuning.

Although research in networked control became very popular in the last couple of years, there are few publications targeting the application of sliding mode control for networked control systems. In the past, the idea of using event-triggered control in combination with sliding mode control was already studied and can be found e.g. in [14, 15]. The main drawback of these methods is that the network imperfections are not explicitly modeled. They only focus on the control design to reduce the communication load, which should indirectly lead to smaller delays and fewer packet losses. Since the network effects are not considered directly, no guarantees on stability and performance can be given.

This dissertation aims to develop a powerful networked control system by bringing together the fields of networked control and sliding mode control. This is done by deriving a mathematical model of the networked control system which is suitable for the design of discrete time sliding mode controllers. As the applicability of the different sliding mode control algorithms highly depends on the specific problem statement, different sliding mode control algorithms are presented in order to cover a large variety of applications. The effectiveness of the presented approaches is validated by means of laboratory experiments and numerical simulations.

In this dissertation the influence of the network-induced time delays are investigated. The influence of data loss is neglected. It is assumed that a protocol is used which ensures reliable transmissions. However, the question arises why it even is necessary to consider time delays in the controller design. This is motivated in the following section.

1.1 Why consider Time Delays in the Controller Design?

Consider a classical unity feedback loop as depicted in figure 1.2 where $R(s)$ denotes the transfer function of the controller and $P(s)$ the transfer function of the plant.

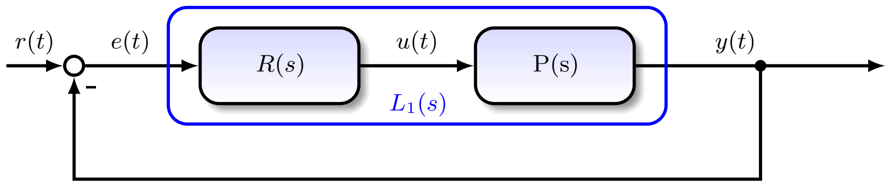


Figure 1.2: Classical unity feedback loop.

Assume that the open loop transfer function $L_1(s) = R(s)P(s)$ fulfills the following properties:

- the DC gain is positive
- all poles of $L_1(s)$ have negative real parts except for one possible pole on the imaginary axis of the complex s-plane
- the magnitude of the frequency response intersects the 0 dB line exactly one time and remains below it for increasing frequencies.

If all these properties are fulfilled, the simplified Nyquist criterion [16] can be applied. This means, bounded input bounded output (BIBO) stability of the closed loop system is ensured if, and only if, the phase margin

$$\Phi_r = \angle L_1(j\omega_c) + 180^\circ \qquad \text{with} \qquad |L_1(j\omega_c)| = 1 \tag{1.1}$$

is positive, i.e.

$$\Phi_r > 0. \tag{1.2}$$

Assume that the controller transfer function $R(s)$ is designed in such a way that $L_1(s)$ satisfies (1.2). Hence, BIBO stability of the closed loop system is ensured.

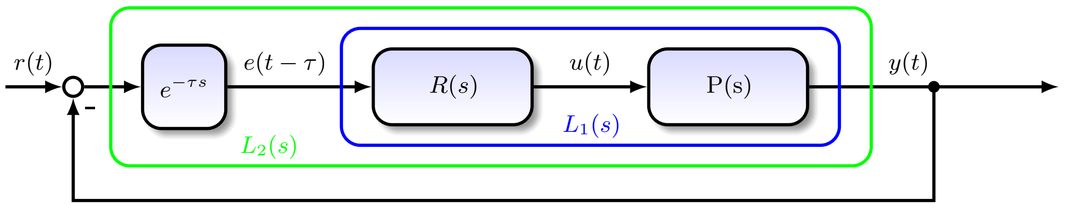


Figure 1.3: Unity feedback loop with time delay.

Now consider the same unity feedback loop but with a constant time delay τ at the input of the controller as depicted in figure 1.3. This results in the open loop transfer function $L_2(s) = e^{-\tau s}L_1(s)$. The magnitude and phase is then given by

$$|L_2(j\omega)| = \left|e^{-\tau j\omega}\right| |L_1(j\omega)| = |L_1(j\omega)|\,, \qquad \angle L_2(j\omega) = \angle L_1(j\omega) - \tau. \tag{1.3}$$

The magnitude of the frequency response is unaffected by the time delay. Thus, the frequency ω_c is the same as for $L_1(s)$. As the argument of the frequency response is negatively influenced, it is clear from (1.2) and (1.3) that an upper limit $\bar{\tau}$ exists at which BIBO stability of the closed loop system is lost if τ exceeds $\bar{\tau}$.

Example 1.1. Consider the open loop transfer function

$$L_1(s) = \frac{s+1}{s^2+2s+2} \tag{1.4}$$

which fulfills the properties listed above. Therefore, the simplified Nyquist criterion can be applied. The blue lines in figure 1.4 verify that the transfer function $L_1(s)$ satisfies the simplified Nyquist criterion and thus ensures BIBO stability of the closed loop system. Figure 1.4 additionally depicts the arguments of the frequency response $L_2(j\omega)$ with the time delays $\tau = 1$ in red, $\tau = 7$ in brown and $\tau = 10$ in black. This figure illustrates that the simplified Nyquist criterion is violated for $\tau \geq 7$. Hence, BIBO-stability of the closed loop system is lost for $\tau \geq 7$.

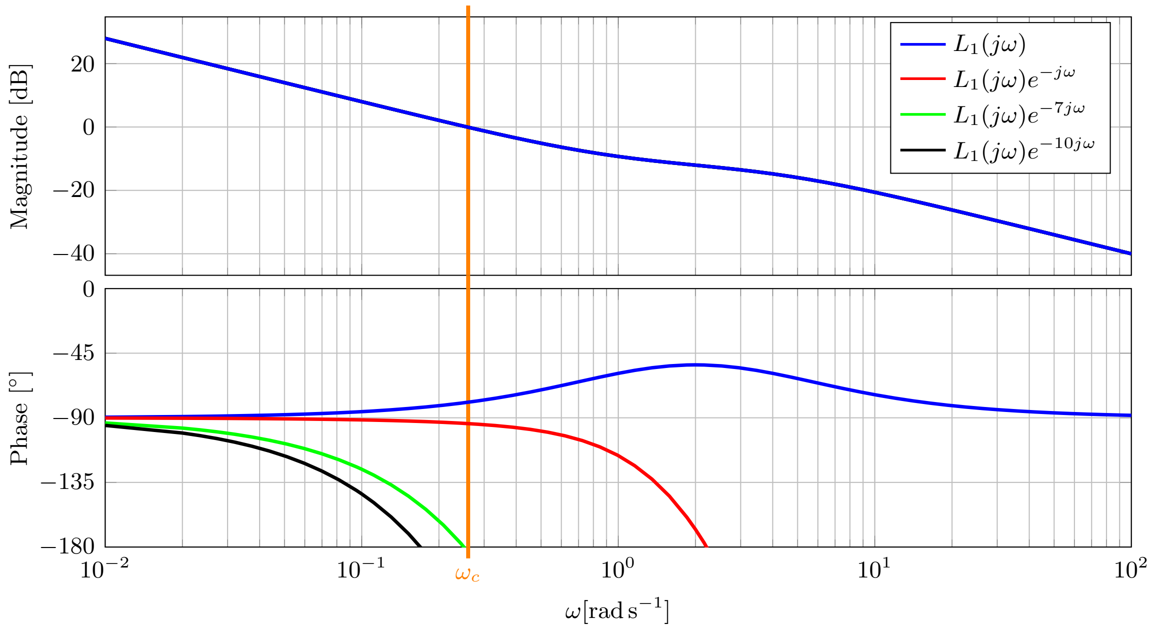


Figure 1.4: Bode plots of $L_1(j\omega)$ and $L_2(j\omega)$ with increasing time delay τ.

Even these simple considerations clearly show that it is necessary to consider time delays in the controller design. Otherwise, even stability of the closed loop system can not be guaranteed for larger time delays.

1.2 Considered Architecture

The architecture considered in this dissertation is shown in the form of a block diagram in figure 1.5. Consider the linear time-invariant plant

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{A}_c\boldsymbol{x}(t) + \boldsymbol{B}_c(\boldsymbol{u}^*(t) + \boldsymbol{f}(t)) \tag{1.5}$$

with state vector $\boldsymbol{x} \in \mathbb{R}^n$, inputs $\boldsymbol{u}^* = \begin{bmatrix} u_1^* & u_2^* & \cdots & u_m^* \end{bmatrix}^\mathrm{T}$ and matched perturbations $\boldsymbol{f} = \begin{bmatrix} f_1 & f_2 & \cdots & f_m \end{bmatrix}^\mathrm{T}$. The dynamic matrix $\boldsymbol{A}_c$ and input matrix $\boldsymbol{B}_c$ have appropriate dimensions.

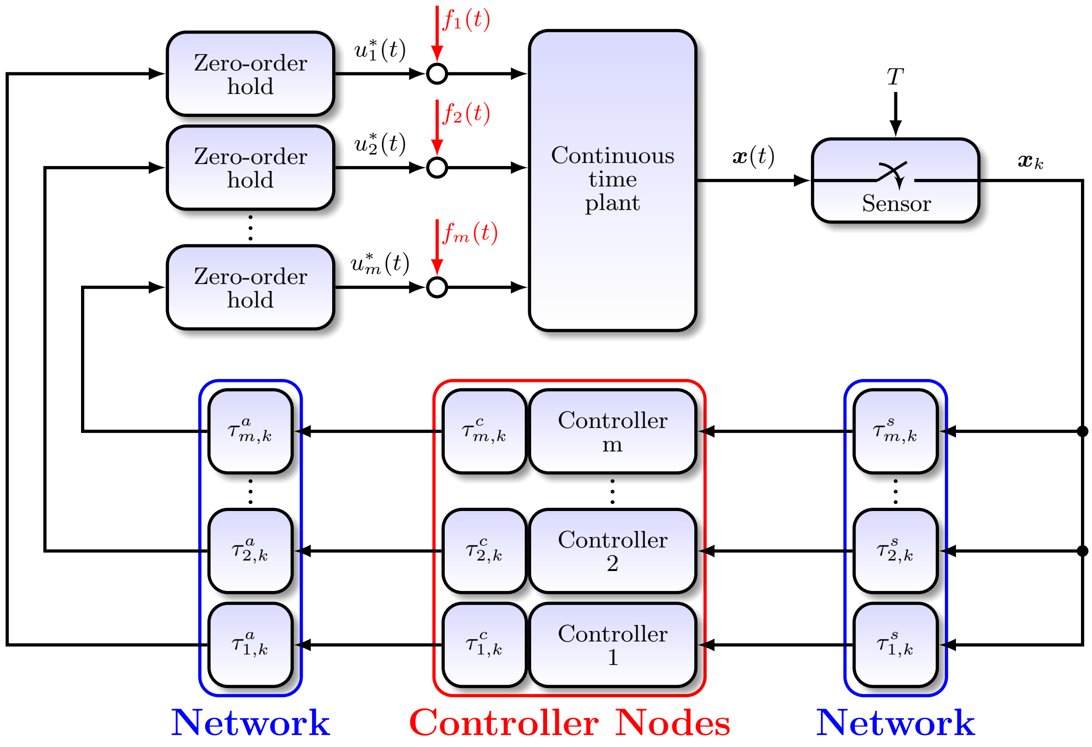


Figure 1.5: Architecture of the considered networked control systems.

The sensor block represents the measurement devices which are attached to the continuous time plant. These measurement devices measure the continuous states $\boldsymbol{x}(t)$ periodically with the constant sampling time T. This results in the sampled state vector $\boldsymbol{x}_k = \boldsymbol{x}(kT)$, $k \in \mathbb{N}_0$. For each of the m input channels, a controller node is implemented which receives this measurement data $\boldsymbol{x}_k$ through a communication network. Due to network imperfections, the i^{th} controller node receives the sensor data delayed by the variable time delay $\tau^s_{i,k} \in \mathbb{R}^+$. As the controller nodes may be implemented on different devices, these delays are in general different for each channel. Each controller node evaluates the control law based on the received data, which introduces an additional variable time delay $\tau^c_{i,k} \in \mathbb{R}^+$ that accounts for limited computational power of the controller nodes computing platform. The controller node transmits the computed output to its corresponding actuator which receives the data delayed by $\tau^a_{i,k} \in \mathbb{R}^+$. Throughout this dissertation, the following assumptions hold.

Assumption 1.2. *System* (1.5) *is controllable, the input matrix* $\boldsymbol{B}_c$ *has full column rank and the constant sampling time* T *is non-pathological.*

Remark 1.3. Controllability and/or observability can be lost by sample and hold discretization of a controllable and observable continuous time system. The specific sampling times, where controllability and/or observability is lost, are called pathological sampling times. One can show (see [17]) that the discrete time system is controllable and observable if distinct eigenvalues of the continuous time dynamic matrix are mapped to distinct eigenvalues of the continuous time dynamic matrix, i.e.

$$e^{s_i T} \neq e^{s_k T} \quad \text{for} \quad s_i \neq s_k \tag{1.6}$$

where

$$s_i = \sigma_i + j\omega_i \qquad \text{and} \qquad s_k = \sigma_k + j\omega_k \tag{1.7}$$

denote eigenvalues of the continuous time dynamic matrix. Applying (1.7) to (1.6) results in

$$e^{T\sigma_i} e^{jT\omega_i} \neq e^{T\sigma_k} e^{jT\omega_k}. \tag{1.8}$$

It is clearly visible that (1.8) can only be violated if

$$\sigma_i = \sigma_k \qquad \text{and} \qquad T\omega_k = T\omega_i \pm 2\nu\pi \quad \text{with} \quad \nu \in \mathbb{N}. \tag{1.9}$$

Therefore, the condition

$$T \neq \frac{\pm 2\nu\pi}{\omega_k - \omega_i} \quad \text{with} \quad \nu \in \mathbb{N} \tag{1.10}$$

has to be fulfilled for all complex eigenvalues of the continuous time dynamic matrix with equal real parts. Otherwise, the sampling time is pathological. More details are shown in [17].

Assumption 1.4. *The communication networks ensure loss-free communication, i.e. no packet dropouts occur, and the sum of all time delays for each input channel is bounded by an integer multiple of the sampling time* T *i.e.,*

$$\tau_{i,k} = \tau^s_{i,k} + \tau^c_{i,k} + \tau^a_{i,k} \leq \delta_i T, \quad \forall k, \quad i = 1, 2, \dots, m \tag{1.11}$$

with $\delta_i \in \mathbb{N}$.

The aims and contributions of this dissertation in the field of robust networked control are summarized in the following section.

1.3 Aims and Contributions of this Work

The main aim of this dissertation is to develop control algorithms for networked control systems which are robust with respect to a certain class of perturbation. Network induced variable time delays are explicitly considered in the controller design. Robustness with respect to the perturbation is achieved by developing control algorithms based on sliding mode techniques.

To satisfy this aim, a discrete time mathematical model of the networked control system is developed which contains the networked induced variable time delays. Due to the variable time delays, this mathematical model is time-variant. Introducing a buffering mechanism at the receiving end of the feedback channels makes it possible to reduce the time-variant mathematical model to a time-invariant one. Using this model, novel control strategies for the centralized and spatially distributed network topologies are proposed in this work.

Networked control systems in which one single controller node evaluates the control law for all plant input channels are called centralized networked control systems. Due to this specific topology, each feedback channel is affected by the same time delay since there is only one feedback path in the network. In addition, the centralized controller is capable of using global information, i.e. the history of control samples of all input channels. In this work it will be shown that sliding variables with a specific structure are necessary for these centralized networked control systems. As a result, classical reaching law based sliding mode control algorithms are not directly applicable in this setting. Therefore, three reaching laws for centralized networked control systems are developed. The parameter choices and achieved accuracies are discussed and compared. The results for single-input systems are published in

> J. Ludwiger, M. Steinberger, M. Rotulo, M. Horn, A. Luppi, G. Kubin, and A. Ferrara, "Towards networked sliding mode control," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, Dec. 2017, pp. 6021–6026
>
> J. Ludwiger, M. Steinberger, M. Horn, G. Kubin, and A. Ferrara, "Discrete time sliding mode control strategies for buffered networked systems," in *2018 IEEE Conference on Decision and Control (CDC)*, Dec. 2018, pp. 6735–6740.

Networked control systems in which the control law for each input channel is implemented in separate controller nodes are called spatially distributed networked control systems. This increased flexibility comes with the additional challenge that the time delays for each feedback channel can be different since these feedback channels are implemented by different network paths. Moreover, only local information, i.e. the history of only the own control samples, is available. These challenges are tackled by developing an integral sliding mode based control algorithm with a specific choice of the sliding variables and by designing the nominal control laws by solving linear matrix inequalities. The effectiveness of the integral sliding mode based approach is verified for single-input systems by means of a laboratory experiment and for multi-input systems by means of numerical simulations. This approach is published for single-input systems in

> J. Ludwiger, M. Reichhartinger, M. Steinberger, and M. Horn, "Discrete-time super twisting controller for networked control systems," *IFAC-PapersOnLine*, vol. 52, no. 16, pp. 789–794, Sep. 2019, 11th IFAC Symposium on Nonlinear Control Systems NOLCOS 2019

and the extension to multi-input spatially distributed systems in

> J. Ludwiger, M. Steinberger, and M. Horn, "Spatially distributed networked sliding mode control," *IEEE Control Systems Letters*, vol. 3, no. 4, pp. 972–977, Oct. 2019.

A simulation toolbox is developed to conveniently perform simulations of centralized and spatially distributed networked control systems. This toolbox is designed with a special focus on the ability to integrate custom sliding mode control algorithms with very little effort. The interface to include custom sliding mode control algorithms adds a layer of abstraction to standardize the implementation process for newly developed algorithms.

The contributions of this work are summarized as follows:

- Development of a discrete time mathematical model for networked control systems with different variable time delays for each feedback channel.
- Introduction of a buffering mechanism at the receiving end of the feedback channels to ensure constant round trip times. This results in a time-invariant mathematical model.
- Sliding mode based controller design for centralized networked control systems.
  - Three different algorithms for centralized networked control systems.
  - Robustness of the resulting algorithms with respect to first order actuator dynamics is shown for a mechanical system.
- Sliding mode based controller design for spatially distributed control systems.
  - Design of an integral sliding mode based control algorithm which makes it possible to cast the problem in a form where recently developed sliding mode controllers are applicable.
  - The nominal and the sliding mode part of the control law are designed in such a way that different time delays in the feedback channels and the availability of only local information are considered.
  - Development of a design procedure for the nominal control law which is based on solving linear matrix inequalities.
- Development of a highly flexible simulation toolbox to conveniently design and simulate centralized and spatially distributed networked control systems.

Chapter 2

Modeling Networked Control Systems

Contents

In this chapter, the modeling of the considered networked control system will be presented. To improve readability, modeling is carried out for the single-input case first and the results are then extended to the multi-input case.

2.1 Modeling of Single-Input Networked Control Systems

Consider the system

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{A}_c \boldsymbol{x}(t) + \boldsymbol{b}_c(u^*(t) + f(t)) \tag{2.1}$$

which results from (1.5) with $m = 1$. The architecture in figure 1.5 is also adapted to the single-input case as shown in figure 2.1.

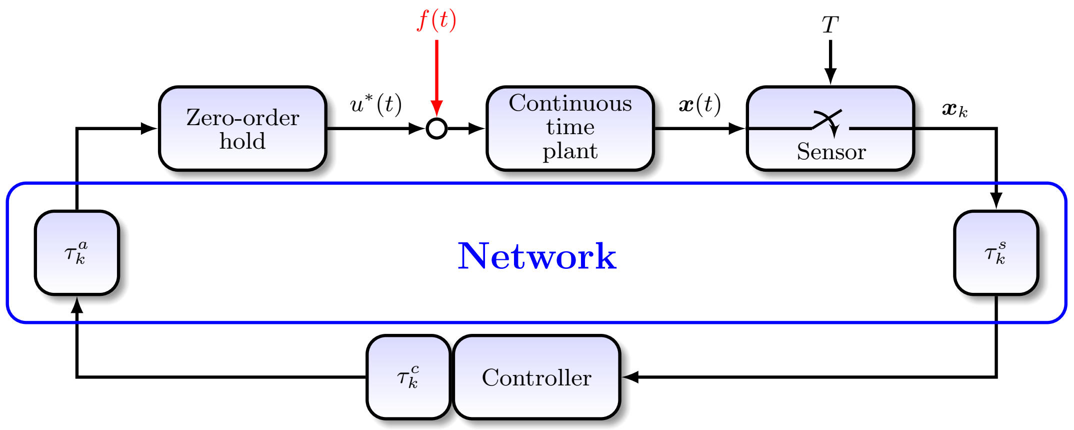


Figure 2.1: Architecture of the considered single-input networked control systems.

In order to simplify the modeling of the networked control system, the three individual time delays should be considered as one single time delay

$$\tau_k = \tau_k^s + \tau_k^c + \tau_k^a, \tag{2.2}$$

the so-called round trip time. The next section focuses on the circumstances under which this simplification is valid.

2.1.1 Combining the Time Delays in the Networked Control System to the Round Trip Time

In order to study the conditions under which the three individual delays τ_k^s, τ_k^c and τ_k^a can be combined to the round trip time τ_k, a notation is introduced that links each discrete time sample to the time instant when it is received. Consider the single delay from the sensor to the controller as depicted in figure 2.2. The notation $\boldsymbol{x}_k^{\tau_k^s}$ signifies that the sample $\boldsymbol{x}_k$ leaves the delay block at time instant $t = kT + \tau_k^s$. This notation can now be used to

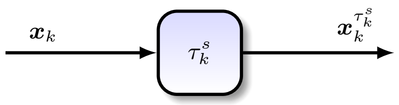
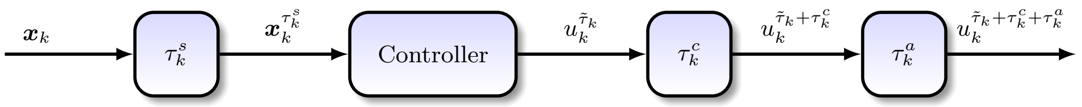


Figure 2.2: Illustration of the delayed discrete time signal.

study the feedback path of the networked control system depicted in figure 2.1. The discrete time control law is assumed as the general function

$$u_k = g(\boldsymbol{x}_k, \boldsymbol{x}_{k-1}, \boldsymbol{x}_{k-2}, \dots). \tag{2.3}$$

Since this function uses previous samples of $\boldsymbol{x}_k$, the controller is dynamic. The feedback channel of the networked control system with the applied notation is depicted in figure 2.3. As already shown in figure 2.2, the controller receives the sample $\boldsymbol{x}_k$ at time instant $kT + \tau_k^s$.



$\boldsymbol{x}_k$
τ_k^s
$\boldsymbol{x}_k^{\tau_k^s}$
Controller
$u_k^{\tilde{\tau}_k}$
τ_k^c
$u_k^{\tilde{\tau}_k+\tau_k^c}$
τ_k^a
$u_k^{\tilde{\tau}_k+\tau_k^c+\tau_k^a}$

Figure 2.3: Feedback channel of the networked control system with the applied notation.

It is evident that the control law (2.3) can only be evaluated after all arguments are available. In general, the controller must therefore wait until the necessary sensor measurements are received. Evaluating the control law results in

$$u_k^{\tilde{\tau}_k} = g(\boldsymbol{x}_k^{\tau_k^s}, \boldsymbol{x}_{k-1}^{\tau_{k-1}^s}, \boldsymbol{x}_{k-2}^{\tau_{k-2}^s}, \dots) \tag{2.4}$$

where the delay induced by waiting for the required sensor measurements is considered in $\tilde{\tau}_k$. The additional restriction

$$\tau_{k-1}^s - \tau_k^s \leq T \quad \forall k \tag{2.5}$$

on the sensor to the controller time delay is necessary to ensure that all previous samples are already available whenever $\boldsymbol{x}_k$ is received. This ensures that the control law can be evaluated whenever a packet is received since no additional time delay is introduced by waiting for necessary samples, i.e.

$$u_k^{\tilde{\tau}_k+\tau_k^c+\tau_k^a} = u_k^{\tau_k^s+\tau_k^c+\tau_k^a} = u_k^{\tau_k}. \tag{2.6}$$

If (2.5) is satisfied, the output of the feedback system depicted in figure 2.3, which represents the input to the zero-order hold at the plant, is given by $u_k^{\tau_k^s+\tau_k^c+\tau_k^a}$.

The two possibilities to consider the three delays as one single round trip time are depicted in figures 2.4 and 2.5. In figure 2.4 the round trip time is considered at the sensor side of the feedback channel. This means that the controller receives the samples delayed by the round trip time τ_k. In order to be able to evaluate the control law whenever a sample is received, all necessary previous samples must have previously been received in this case also. As a consequence, the additional condition

$$\tau_{k-1} - \tau_k \leq T \quad \forall k \tag{2.7}$$

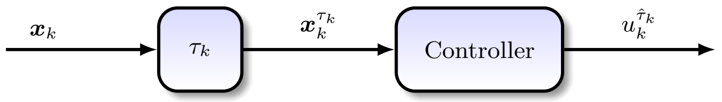


Figure 2.4: Single time delay considered at the sensor side.

has to be satisfied. Otherwise, the controller must wait for previous samples to arrive. This induces an additional time delay which is considered in $\hat{\tau}_k$. Conclusively, the configurations depicted in figures 2.3 and 2.4 are only equivalent for dynamic controllers if the conditions (2.5) and (2.7) are satisfied.

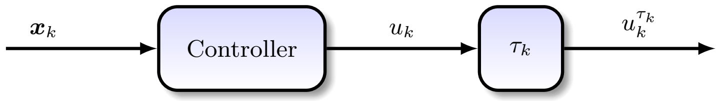


Figure 2.5: Single time delay considered at the actuator side.

The round trip time can also be considered at the actuator side of the feedback channel, which is illustrated in figure 2.5. This representation has the appealing property that whenever (2.5) is satisfied, the architectures in figures 2.3 and 2.5 are equivalent. Therefore, this configuration is used in the remainder of this dissertation.

Remark 2.1. Note that the three architectures shown in figures 2.3–2.5 are always equivalent (no constraint on the time delays) if the control law is static, i.e.

$$u_k = g(\boldsymbol{x}_k). \tag{2.8}$$

2.1.2 The Network Timing

The results from section 2.1.1 offer the possibility to simplify the networked control system by introducing the round trip time τ_k. The simplified architecture shown in figure 2.6 is equivalent to the architecture in figure 2.1 under assumption 2.2.

Assumption 2.2. *One of the following conditions holds.*

1. *A static control law is used, i.e. $u_k = g(\boldsymbol{x}_k)$.*
2. *A dynamic control law is used, i.e. $u_k = g(\boldsymbol{x}_k, \boldsymbol{x}_{k-1}, \dots)$, and the time delay from the sensor to the controller satisfies $\tau_{k-1}^s - \tau_k^s \leq T$.*

In order to derive a mathematical model of the networked control system, the timing within the network has to be investigated in more detail. As figure 2.6 shows, the control law is immediately evaluated in each sampling step resulting in the control signal u_k. This control signal is then delayed by the round trip time τ_k. Depending on the limit of the time delay, one can distinguish two cases, the small delay case where $\tau_k < T$, $\forall k$ and the large delay case where $\exists k: \ \tau_k \geq T$.

A typical timing diagram for the small delay case is shown in figure 2.7. The control samples are generated at the time instants $t \in \{kT, (k+1)T, (k+2)T, \dots\}$. The vertical arrows

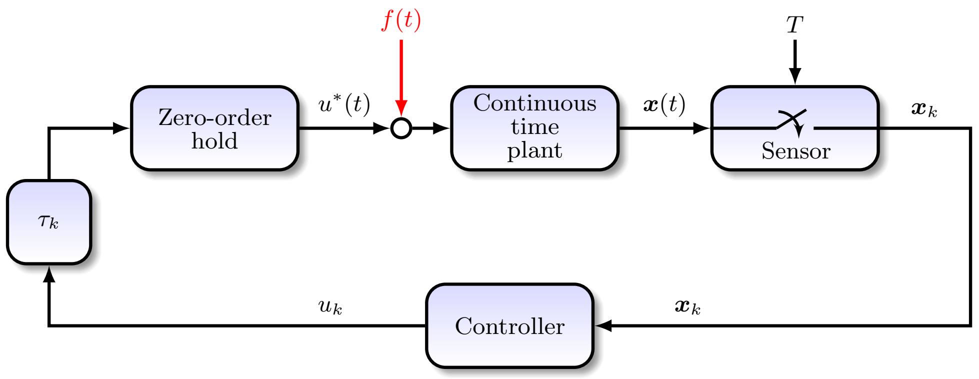


Figure 2.6: Architecture considering the round trip time as a single delay at the actuator side of the plant.

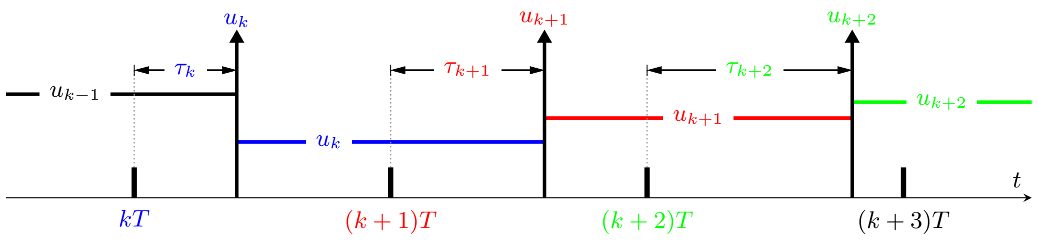


Figure 2.7: Timing diagram for the small delay case (i.e. $\tau_k < T,\ \forall k$).

represent the time instants at which the corresponding sample is available at the zero-order hold. The output of the zero-order hold $u^*(t)$ is represented by the colored trace. The corresponding samples are also labeled on this trace. Note that all elements with the same color correspond to the same sample. Figure 2.7 additionally shows the property of the small delay case that exactly one new control sample is received between two sampling instances.

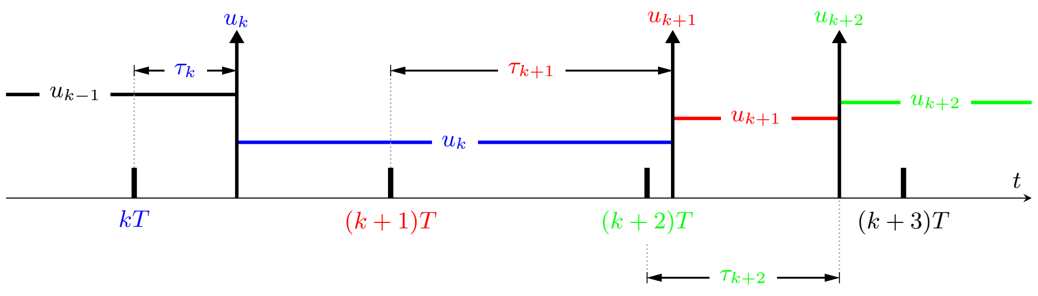


Figure 2.8: Timing diagram for the large delay case (i.e. $\exists k,\ \tau_k \geq T$).

In the large delay case, it is possible that zero, one or up to δ samples are received within one sampling period for the delay being bounded by $\tau_k < \delta T$ where $\delta \in \mathbb{N}^+$. In figure 2.8, the delay is bounded by $\tau_k < 2T,\ \forall k$. Additionally, data can be received out-of-order in the

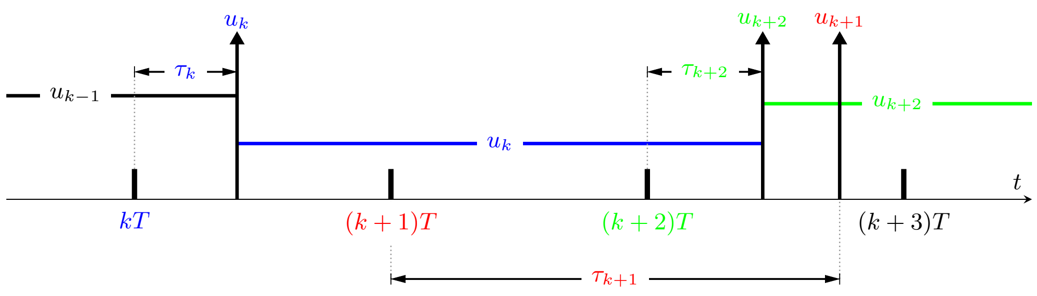


Figure 2.9: Timing diagram showing the message rejection mechanism.

large delay case as depicted in figure 2.9. A message rejection mechanism is implemented at the zero-order hold which drops a packet whenever newer information is already available to deal with this out-of-order arrivals. This message rejection is illustrated by the colored trace in figure 2.9 which represents the output of the zero-order hold $u^*(t)$.

These observations lead to the mathematical representation of $u^*(t)$ summarized in lemma 2.3.

Lemma 2.3. *Consider a networked control system as depicted in figure 2.6. The output $u^*(t)$ of the zero-order hold can be mathematically formulated by*

$$u^*(t) = u_{k+j-\delta} \quad \textit{for} \quad kT + t_k^j \leq t < kT + t_k^{j+1}, \quad j = 0, 1, \dots, \delta \tag{2.9}$$

where the arrival times $t_k^j \in [0, T]$ are given by

$$\begin{aligned} t_k^j = \min\{ & \max[0, \tau_{k+j-\delta} - (\delta - j)T], \\ & \max[0, \tau_{k+j-\delta+1} - (\delta - j - 1)T], \\ & \dots, \max[0, \tau_k], T\}. \end{aligned} \tag{2.10}$$

Note that the arrival times fulfill the property

$$0 = t_k^0 \leq t_k^1 \leq \dots \leq t_k^\delta \leq t_k^{\delta+1} = T. \tag{2.11}$$

Proof. The proof follows directly from the proof of lemma 1 in [22] with $\bar{d} = \delta$, $\bar{\delta} = 0$, $\underline{d} = 0$, $h_k = T,\ \forall k$ and $m_k = 0, \forall k$. □

2.1.3 The Discrete Time Model of the Networked Control System

The full state space model (lifted model) of the single-input networked control system is summarized in theorem 2.5.

Assumption 2.4. *The sampling time T is chosen small enough to ensure that the inter-sample behavior of the perturbation $\boldsymbol{f}(t)$ is negligible. Therefore, the perturbation*

$\boldsymbol{f}(t)$ can be assumed as piece-wise constant, i.e.

$$\boldsymbol{f}(t) = \boldsymbol{f}(kT) = \boldsymbol{f}_k, \qquad kT \leq t < (k+1)T, \quad k \in \mathbb{N}_0. \tag{2.12}$$

Further assumptions on the boundedness of the perturbation's magnitude or change rates will be made later.

Theorem 2.5. Consider the networked control system depicted in figure 2.1 with plant (2.1) and assume that assumptions 1.4, 2.2 and 2.4 hold. The networked control system can then be represented by the discrete time model

$$\boldsymbol{\xi}_{k+1} = \hat{\boldsymbol{A}}(\boldsymbol{\theta}_k)\boldsymbol{\xi}_k + \hat{\boldsymbol{b}}(\boldsymbol{\theta}_k)u_k + \hat{\boldsymbol{b}}_f f_k \tag{2.13}$$

with the vector of arrival times

$$\boldsymbol{\theta}_k = \begin{bmatrix}\boldsymbol{\theta}_k^0 & \boldsymbol{\theta}_k^1 & \cdots & \boldsymbol{\theta}_k^\delta\end{bmatrix} \quad \text{and} \quad \boldsymbol{\theta}_k^j = \begin{bmatrix} t_k^j & t_k^{j+1}\end{bmatrix}^{\mathrm{T}} \quad \text{where} \quad j = 0, 1, \ldots, \delta, \tag{2.14}$$

the lifted state vector

$$\boldsymbol{\xi}_k = \begin{bmatrix}\boldsymbol{x}_k^{\mathrm{T}} & u_{k-1} & u_{k-2} & \cdots & u_{k-\delta}\end{bmatrix}^{\mathrm{T}}, \tag{2.15}$$

the system matrix

$$\hat{\boldsymbol{A}}(\boldsymbol{\theta}_k) = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{b}(\boldsymbol{\theta}_k^{\delta-1}) & \boldsymbol{b}(\boldsymbol{\theta}_k^{\delta-2}) & \cdots & \boldsymbol{b}(\boldsymbol{\theta}_k^1) & \boldsymbol{b}(\boldsymbol{\theta}_k^0) \\ \boldsymbol{0} & 0 & 0 & \cdots & 0 & 0 \\ \boldsymbol{0} & 1 & 0 & \cdots & 0 & 0 \\ \vdots & & \ddots & & \vdots & \vdots \\ \boldsymbol{0} & 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \tag{2.16}$$

and the input vectors

$$\hat{\boldsymbol{b}}(\boldsymbol{\theta}_k) = \begin{bmatrix} \boldsymbol{b}(\boldsymbol{\theta}_k^\delta) \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \hat{\boldsymbol{b}}_f = \begin{bmatrix} \boldsymbol{b}_f \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{2.17}$$

where

$$\boldsymbol{A} = e^{\boldsymbol{A}_c T}, \qquad \boldsymbol{b}(\boldsymbol{\theta}_k^j) = e^{\boldsymbol{A}_c T}\int_{t_k^j}^{t_k^{j+1}} e^{-\boldsymbol{A}_c s}\boldsymbol{b}_c ds, \qquad \boldsymbol{b}_f = \int_0^T e^{\boldsymbol{A}_c s}\boldsymbol{b}_c ds. \tag{2.18}$$

Proof. The trajectories of plant (2.1) are given by

$$\boldsymbol{x}(t) = e^{\boldsymbol{A}_c t}\boldsymbol{x}_0 + \int_0^t e^{\boldsymbol{A}_c(t-\sigma)}\boldsymbol{b}_c\left[u^*(\sigma) + f(\sigma)\right]d\sigma. \tag{2.19}$$

Applying the discretization with sampling time T results in

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}((k+1)T) = e^{\boldsymbol{A}_c T} \underbrace{\left(e^{\boldsymbol{A}_c kT}\boldsymbol{x}_0 + \int_0^{kT} e^{\boldsymbol{A}_c(kT-\sigma)}\boldsymbol{b}_c \left[u^*(\sigma) + f(\sigma)\right] d\sigma \right)}_{\boldsymbol{x}_k=\boldsymbol{x}(kT)} \tag{2.20}$$

$$+ \int_{kT}^{(k+1)T} e^{\boldsymbol{A}_c((k+1)T-\sigma)}\boldsymbol{b}_c \left[u^*(\sigma) + f(\sigma)\right] d\sigma \tag{2.21}$$

$$= e^{\boldsymbol{A}_c T}\boldsymbol{x}_k + \int_{kT}^{(k+1)T} e^{\boldsymbol{A}_c((k+1)T-\sigma)}\boldsymbol{b}_c \left[u^*(\sigma) + f(\sigma)\right] d\sigma. \tag{2.22}$$

Substituting $\sigma = kT + s$ and exploiting assumption 2.4 gives

$$\boldsymbol{x}_{k+1} = e^{\boldsymbol{A}_c T}\boldsymbol{x}_k + e^{\boldsymbol{A}_c T}\int_0^T e^{-\boldsymbol{A}_c s}\boldsymbol{b}_c[u^*(kT+s) + \underbrace{f(kT+s)}_{f_k}]ds. \tag{2.23}$$

Considering the arrival times, as described in lemma 2.3 and illustrated in figure 2.10,

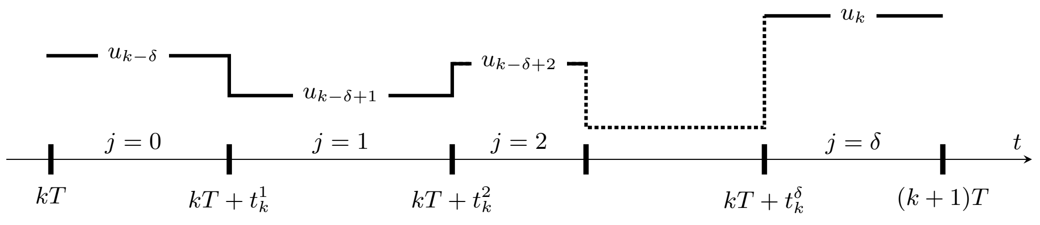


Figure 2.10: Control signal $u^*(t)$ in the interval of two sampling instances.

results in

$$\boldsymbol{x}_{k+1} = \underbrace{e^{\boldsymbol{A}_c T}}_{\boldsymbol{A}}\boldsymbol{x}_k + \underbrace{e^{\boldsymbol{A}_c T}\int_{0=t_k^0}^{t_k^1} e^{-\boldsymbol{A}_c s}\boldsymbol{b}_c ds}_{\boldsymbol{b}(\boldsymbol{\theta}_k^0)}\, u_{k-\delta} + \underbrace{e^{\boldsymbol{A}_c T}\int_{t_k^1}^{t_k^2} e^{-\boldsymbol{A}_c s}\boldsymbol{b}_c ds}_{\boldsymbol{b}(\boldsymbol{\theta}_k^1)}\, u_{k-\delta+1} +$$

$$+ \cdots + \underbrace{e^{\boldsymbol{A}_c T}\int_{t_k^\delta}^{T} e^{-\boldsymbol{A}_c s}\boldsymbol{b}_c ds}_{\boldsymbol{b}(\boldsymbol{\theta}_k^\delta)}\, u_k + \underbrace{\int_0^T e^{\boldsymbol{A}_c s}\boldsymbol{b}_c ds}_{\boldsymbol{b}_f}\, f_k \tag{2.24}$$

$$= \boldsymbol{A}\boldsymbol{x}_k + \sum_{j=0}^{\delta} \boldsymbol{b}(\boldsymbol{\theta}_k^j) u_{k+j-\delta} + \boldsymbol{b}_f f_k.$$

Finally, the lifted model (2.13) follows from (2.24) by introducing the lifted state vector (2.15). □

2.2 Modeling of multi-input Networked Control Systems

As previously indicated, the results from section 2.1 will now be extended to the multi-input case. Using assumption 2.2 for each feedback channel of the multi-input networked control system individually results in assumption 2.6.

Assumption 2.6. *One of the following conditions holds for $i = 1, 2, \dots, m$.*

1. *A static control law is used in the i^th controller node, i.e. $u_{i,k} = g_i(\boldsymbol{x}_k)$.*
2. *A dynamic control law is used in the i^th controller node, i.e. $u_{i,k} = g_i(\boldsymbol{x}_k, \boldsymbol{x}_{k-1}, \dots)$, and the time delay from the sensor to the controller fulfills $\tau^s_{i,k-1} - \tau^s_{i,k} \leq T$.*

If assumption 2.6 holds, the three individual time delays of the i^th channel ($\tau^s_{i,k}$, $\tau^c_{i,k}$ and $\tau^a_{i,k}$) can be considered as a single time delay

$$\tau_{i,k} = \tau^s_{i,k} + \tau^c_{i,k} + \tau^a_{i,k} \tag{2.25}$$

for analysis purposes. This time delay is called the i^th round trip time and is considered at the actuator side of the i^th channel. As a result, the block diagram of the networked control system in figure 1.5 can be simplified to the one in figure 2.11.

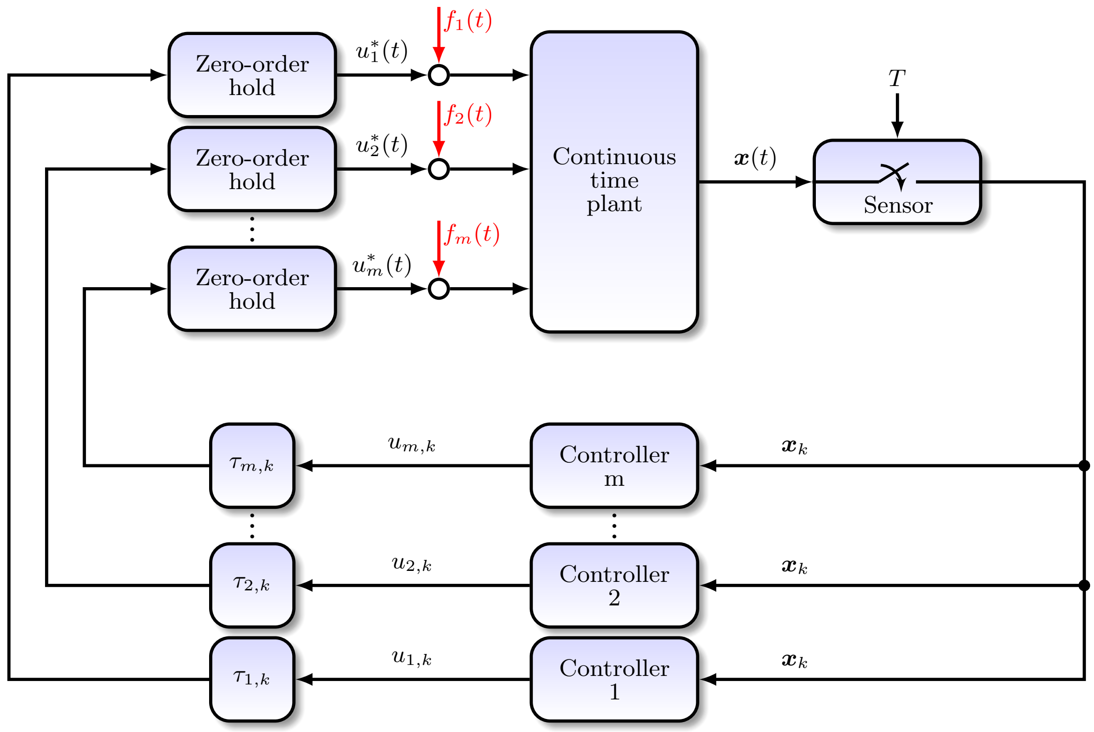


Figure 2.11: Architecture in which the round trip times are considered as single delays at the actuator side of the plant.

Reformulating the multi-input plant given in (1.5) by splitting the input vector for each channel results in

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{A}_c\boldsymbol{x}(t) + \sum_{i=1}^{m} \boldsymbol{b}_{c,i}(u_i^* + f_i) \tag{2.26}$$

where $\boldsymbol{B}_c = \begin{bmatrix}\boldsymbol{b}_{c,1} & \boldsymbol{b}_{c,2} & \cdots & \boldsymbol{b}_{c,m}\end{bmatrix}$.

The discrete time lifted model of the multi-input networked control system is presented in theorem 2.7.

Theorem 2.7. Consider the networked control system depicted in figure 1.5 with plant (1.5) and assume that assumptions 1.4, 2.4 and 2.6 hold. The networked control system can then be represented by the discrete time model

$$\boldsymbol{\xi}_{k+1} = \hat{\boldsymbol{A}}(\boldsymbol{\theta}_{1,k}, \boldsymbol{\theta}_{2,k}, \ldots, \boldsymbol{\theta}_{m,k})\boldsymbol{\xi}_k + \hat{\boldsymbol{B}}(\boldsymbol{\theta}_{1,k}, \boldsymbol{\theta}_{2,k}, \ldots, \boldsymbol{\theta}_{m,k})\boldsymbol{u}_k + \hat{\boldsymbol{B}}_f\boldsymbol{f}_k \tag{2.27}$$

with the vectors of arrival times

$$\begin{aligned}
\boldsymbol{\theta}_{i,k} &= \begin{bmatrix}\boldsymbol{\theta}_{i,k}^0 & \boldsymbol{\theta}_{i,k}^1 & \cdots & \boldsymbol{\theta}_{i,k}^{\delta_i}\end{bmatrix}^{\mathrm{T}}, \\
\boldsymbol{\theta}_{i,k}^j &= \begin{bmatrix}t_{i,k}^j & t_{i,k}^{j+1}\end{bmatrix}^{\mathrm{T}}, \\
t_{i,k}^j &= \min\{\max\left[0, \tau_{i,k+j-\delta_i} - (\delta_i - j)\,T\right], \\
&\qquad \max\left[0, \tau_{i,k+j-\delta_i+1} - (\delta_i - j - 1)\,T\right], \\
&\qquad \ldots, \max\left[0, \tau_{i,k}\right], T\}. \\
j &= 0, 1, \ldots, \delta_i, \quad i = 1, 2, \ldots, m,
\end{aligned} \tag{2.28}$$

the lifted state vector

$$\boldsymbol{\xi}_k = \begin{bmatrix}\boldsymbol{x}_k^{\mathrm{T}} & u_{1,k-1} & u_{1,k-2} & \cdots & u_{1,k-\delta_1} \cdots & u_{m,k-1} & \cdots & u_{m,k-\delta_m}\end{bmatrix}^{\mathrm{T}}, \tag{2.29}$$

the system matrix

$$\hat{\boldsymbol{A}}(\boldsymbol{\theta}_{1,k}, \boldsymbol{\theta}_{2,k}, \ldots, \boldsymbol{\theta}_{m,k}) = \begin{bmatrix}
\boldsymbol{A} & \boldsymbol{B}_1(\boldsymbol{\theta}_{1,k}) & \boldsymbol{B}_2(\boldsymbol{\theta}_{2,k}) & \cdots & \boldsymbol{B}_m(\boldsymbol{\theta}_{m,k}) \\
\boldsymbol{0} & \boldsymbol{E}_{\delta_1\times\delta_1} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\
\boldsymbol{0} & \boldsymbol{0} & \boldsymbol{E}_{\delta_2\times\delta_2} & \cdots & \boldsymbol{0} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{E}_{\delta_m\times\delta_m}
\end{bmatrix} \tag{2.30}$$

and the input vectors

$$\hat{\boldsymbol{B}}(\boldsymbol{\theta}_{1,k}, \boldsymbol{\theta}_{2,k}, \ldots, \boldsymbol{\theta}_{m,k}) = \begin{bmatrix}
\boldsymbol{b}_1(\boldsymbol{\theta}_{1,k}^{\delta_1}) & \boldsymbol{b}_2(\boldsymbol{\theta}_{2,k}^{\delta_2}) & \cdots & \boldsymbol{b}_m(\boldsymbol{\theta}_{m,k}^{\delta_m}) \\
\boldsymbol{e}_{\delta_1\times 1} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\
\boldsymbol{0} & \boldsymbol{e}_{\delta_2\times 1} & \cdots & \boldsymbol{0} \\
\vdots & \vdots & \ddots & \vdots \\
\boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{e}_{\delta_m\times 1}
\end{bmatrix} \qquad \hat{\boldsymbol{B}}_f = \begin{bmatrix}\boldsymbol{B}_f \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0}\end{bmatrix} \tag{2.31}$$

where

$$\boldsymbol{B}_i(\boldsymbol{\theta}_{i,k}) = \begin{bmatrix} \boldsymbol{b}_i(\boldsymbol{\theta}_{i,k}^{\delta_i-1}) & \boldsymbol{b}_i(\boldsymbol{\theta}_{i,k}^{\delta_i-2}) & \cdots & \boldsymbol{b}_i(\boldsymbol{\theta}_{i,k}^{0}) \end{bmatrix} \tag{2.32a}$$

$$\boldsymbol{b}_i(\boldsymbol{\theta}_{i,k}^{j}) = e^{\boldsymbol{A}_c T} \int_{t_{i,k}^{j}}^{t_{i,k}^{j+1}} e^{-\boldsymbol{A}_c s} \boldsymbol{b}_{c,i} ds \tag{2.32b}$$

$$\boldsymbol{E}_{\delta_i \times \delta_i} = \begin{bmatrix} \boldsymbol{0}_{1\times(\delta_i-1)} & 0 \\ \boldsymbol{I}_{(\delta_i-1)} & \boldsymbol{0}_{(\delta_i-1)\times 1} \end{bmatrix} \tag{2.32c}$$

$$\boldsymbol{e}_{\delta_i \times 1} = \begin{bmatrix} 1 \\ \boldsymbol{0}_{\delta_i-1\times 1} \end{bmatrix} \tag{2.32d}$$

$$\boldsymbol{B}_f = \int_0^T e^{\boldsymbol{A}_c s} \boldsymbol{B}_c ds \tag{2.32e}$$

$$\boldsymbol{A} = e^{\boldsymbol{A}_c T} \tag{2.32f}$$

Proof. The trajectories of the multi-input plant (2.26) are given by

$$\boldsymbol{x}(t) = e^{\boldsymbol{A}_c t}\boldsymbol{x}_0 + \sum_{i=1}^{m} \int_0^t e^{\boldsymbol{A}_c(t-\sigma)} \boldsymbol{b}_{c,i} \left[u_i^*(\sigma) + f_i(\sigma)\right] d\sigma. \tag{2.33}$$

Applying the discretization with sampling time T results in

$$\boldsymbol{x}_{k+1} = e^{\boldsymbol{A}_c T}\boldsymbol{x}_k + \sum_{i=1}^{m} \int_{kT}^{(k+1)T} e^{\boldsymbol{A}_c((k+1)T-\sigma)} \boldsymbol{b}_{c,i} \left[u_i^*(\sigma) + f_i(\sigma)\right] d\sigma. \tag{2.34}$$

Substituting $\sigma = kT + s$ gives

$$\boldsymbol{x}_{k+1} = e^{\boldsymbol{A}_c T}\boldsymbol{x}_k + \sum_{i=1}^{m} e^{\boldsymbol{A}_c T} \int_0^T e^{-\boldsymbol{A}_c s} \boldsymbol{b}_{c,i} [u_i^*(kT+s) + \underbrace{f_i(kT+s)}_{f_{i,k}}] ds. \tag{2.35}$$

Applying lemma 2.3 to each input channel individually results in the arrival times given in (2.28). Considering these arrival times, (2.35) can be represented by

$$\boldsymbol{x}_{k+1} = e^{\boldsymbol{A}_c T}\boldsymbol{x}_k + \sum_{i=1}^{m} \sum_{j=0}^{\delta_i} \underbrace{e^{\boldsymbol{A}_c T} \int_{t_{i,k}^{j}}^{t_{i,k}^{j+1}} e^{\boldsymbol{A}_c s} \boldsymbol{b}_{c,i} ds}_{\boldsymbol{b}_i(\boldsymbol{\theta}_{i,k}^{j})} \, u_{i,k+j-\delta_i} + \underbrace{\int_0^T e^{\boldsymbol{A}_c s} \boldsymbol{B}_c ds}_{\boldsymbol{B}_f} \, \boldsymbol{f}_k. \tag{2.36}$$

The lifted model (2.27) follows directly from (2.36) using the lifted state vector (2.29). □

2.3 Numerical Simulation

The lifted model (2.27) is verified by comparing the results of two numerical simulations. In the first simulation, the TrueTime toolbox introduced in [23] was used to implement a continuous time simulation. More details on the simulation environment are given in chapter 7. In the second simulation, the same networked control system is simulated by using the lifted model (2.27) which provides discrete time signals. The results of the two simulations should exactly match at the sampling instances.

Example 2.8. The plant for this simulation is chosen as the third order system

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 & -11 & -6 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} (\boldsymbol{u} + \boldsymbol{f}) \tag{2.37}$$

and the sampling time as

$$T = 0.65\,\mathrm{s}. \tag{2.38}$$

It is assumed that the time delays $\boldsymbol{\tau}_k^s = \begin{bmatrix}\tau_{1,k}^s & \tau_{2,k}^s\end{bmatrix}^{\mathrm{T}}$, $\boldsymbol{\tau}_k^c = \begin{bmatrix}\tau_{1,k}^c & \tau_{2,k}^c\end{bmatrix}^{\mathrm{T}}$ $\boldsymbol{\tau}_k^a = \begin{bmatrix}\tau_{1,k}^a & \tau_{2,k}^a\end{bmatrix}^{\mathrm{T}}$ are respectively bounded by

$$\begin{bmatrix}0\\0\end{bmatrix} \le \boldsymbol{\tau}_k^s \le \begin{bmatrix}0.5\\0.1\end{bmatrix}, \qquad \begin{bmatrix}0\\0\end{bmatrix} \le \boldsymbol{\tau}_k^c \le \begin{bmatrix}0.5\\1.7\end{bmatrix}, \qquad \begin{bmatrix}0\\0\end{bmatrix} \le \boldsymbol{\tau}_k^a \le \begin{bmatrix}0.3\\0.8\end{bmatrix}. \tag{2.39}$$

The delay values are generated as uniformly distributed random numbers within the specified boundaries. Due to the fact that the used control law

$$\boldsymbol{u}_k = \begin{bmatrix} -0.01 & 0.027 & 0.087 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.473 & 1.163 & 0.027 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{\xi}_k \tag{2.40}$$

is static, assumption 2.6 is fulfilled. Hence, the three individual delays in each channel can be treated as one single delay and the lifted model in theorem 2.7 is valid. Note that this controller is tuned based on the plant only (without considering the delays). Consequently, stability is not guaranteed for all possible delay configurations. However, it is used for validation purposes.

Considering the given delay bounds (2.39) and the sampling time (2.38), the round trip times are bounded by

$$\begin{bmatrix}0\\0\end{bmatrix} \le \boldsymbol{\tau}_k = \boldsymbol{\tau}_k^s + \boldsymbol{\tau}_k^c + \boldsymbol{\tau}_k^a \le \begin{bmatrix}2\\4\end{bmatrix} T, \quad \forall k. \tag{2.41}$$

For simulation purposes, the perturbation is assumed to be known as

$$\boldsymbol{f}_k = \begin{bmatrix} 10\sin(0.3kT) \\ 20\sin(0.2kT) \end{bmatrix}. \tag{2.42}$$

The continuous time simulation provides data for all signals at every point in time whereas the discrete time simulation only provides data for the distinct sampling time instances.

Additionally, the three delays $\boldsymbol{\tau}_k^s$, $\boldsymbol{\tau}_k^c$ and $\boldsymbol{\tau}_k^a$ are considered separately in the continuous time simulation while these delays are lumped together to the round trip time $\boldsymbol{\tau}_k$ for the discrete time simulation. However, the discrete time representation offers a basis for the controller design.

The simulation results provided by the continuous time simulation for the first and the second feedback channel are shown in figures 2.12 and 2.13, respectively. In these figures, the first three subplots show the three components of the sampled state vector $\boldsymbol{x}_k$ in blue. The dashed red lines in these subplots refer to the data received by the i^{th} controller node, which is affected by the network-induced delay $\tau_{i,k}^s$. The fourth subplots show the controller outputs without considering the computational time in blue and the controller outputs affected by the computational time $\tau_{i,k}^c$ in red. Finally, the last subplots depict the control signals additionally affected by the networked induced delays $\tau_{i,k}^a$ in blue. These are the signals received at the zero-order hold elements. The red lines in these last subplots refer to the outputs of the zero-order hold elements. Comparing the red and the blue traces in these subplots clearly shows the effect of the message rejection as older samples are dropped whenever newer information is already available.

The first few steps of this simulation will be shown explicitly to demonstrate the procedure for evaluating the discrete time simulation. Assume that the first round trip times are given by

$$\boldsymbol{\tau}_0 = \begin{bmatrix} 0.538 \\ 2.349 \end{bmatrix}, \quad \boldsymbol{\tau}_1 = \begin{bmatrix} 0.461 \\ 1.073 \end{bmatrix}, \quad \boldsymbol{\tau}_2 = \begin{bmatrix} 0.789 \\ 1.367 \end{bmatrix}, \quad \boldsymbol{\tau}_3 = \begin{bmatrix} 0.796 \\ 1.688 \end{bmatrix}, \quad \boldsymbol{\tau}_4 = \begin{bmatrix} 0.213 \\ 1.065 \end{bmatrix}.$$

The vectors of arrival times are computed according to (2.28), which leads to

$$\boldsymbol{\theta}_{1,0} = \begin{bmatrix} 0 & 0 \\ 0 & 0.538 \\ 0.538 & 0.65 \end{bmatrix}, \quad \boldsymbol{\theta}_{1,1} = \begin{bmatrix} 0 & 0 \\ 0 & 0.461 \\ 0.461 & 0.65 \end{bmatrix}, \quad \boldsymbol{\theta}_{1,2} = \begin{bmatrix} 0 & 0 \\ 0 & 0.65 \\ 0.65 & 0.65 \end{bmatrix},$$

$$\boldsymbol{\theta}_{1,3} = \begin{bmatrix} 0 & 0.139 \\ 0.139 & 0.65 \\ 0.65 & 0.65 \end{bmatrix}, \quad \boldsymbol{\theta}_{1,4} = \begin{bmatrix} 0 & 0.146 \\ 0.146 & 0.213 \\ 0.213 & 0.65 \end{bmatrix}$$

for the first input channel and

$$\boldsymbol{\theta}_{2,0} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0.65 \\ 0.65 & 0.65 \end{bmatrix}, \quad \boldsymbol{\theta}_{2,1} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0.65 \\ 0.65 & 0.65 \\ 0.65 & 0.65 \end{bmatrix}, \quad \boldsymbol{\theta}_{2,2} = \begin{bmatrix} 0 & 0 \\ 0 & 0.423 \\ 0.423 & 0.423 \\ 0.423 & 0.65 \\ 0.65 & 0.65 \end{bmatrix},$$

$$\boldsymbol{\theta}_{2,3} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0.65 \\ 0.65 & 0.65 \\ 0.65 & 0.65 \end{bmatrix}, \quad \boldsymbol{\theta}_{2,4} = \begin{bmatrix} 0 & 0 \\ 0 & 0.067 \\ 0.067 & 0.65 \\ 0.65 & 0.65 \\ 0.65 & 0.65 \end{bmatrix}$$

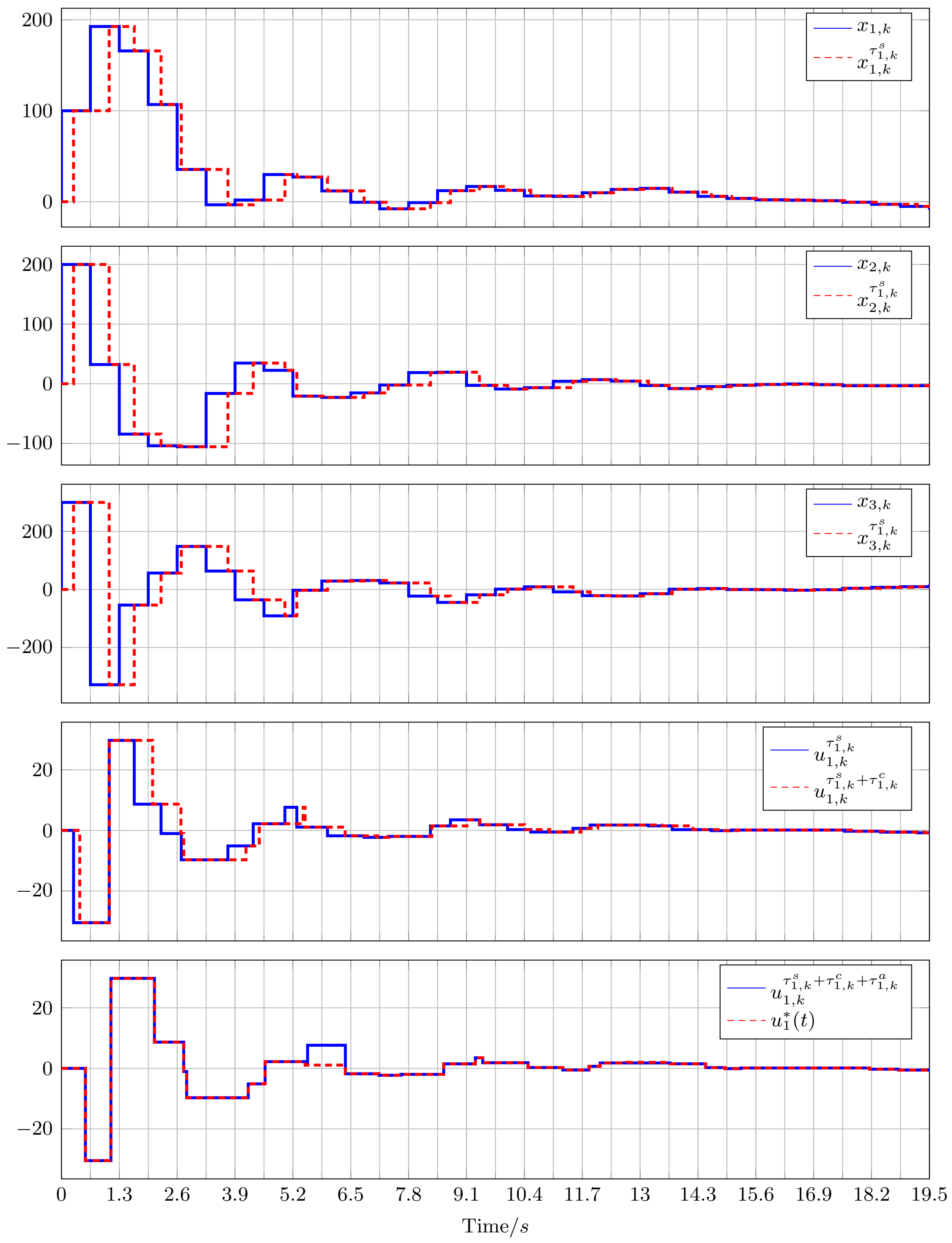


Figure 2.12: Example 2.8: The signals in the first feedback channel obtained with the continuous time simulation.

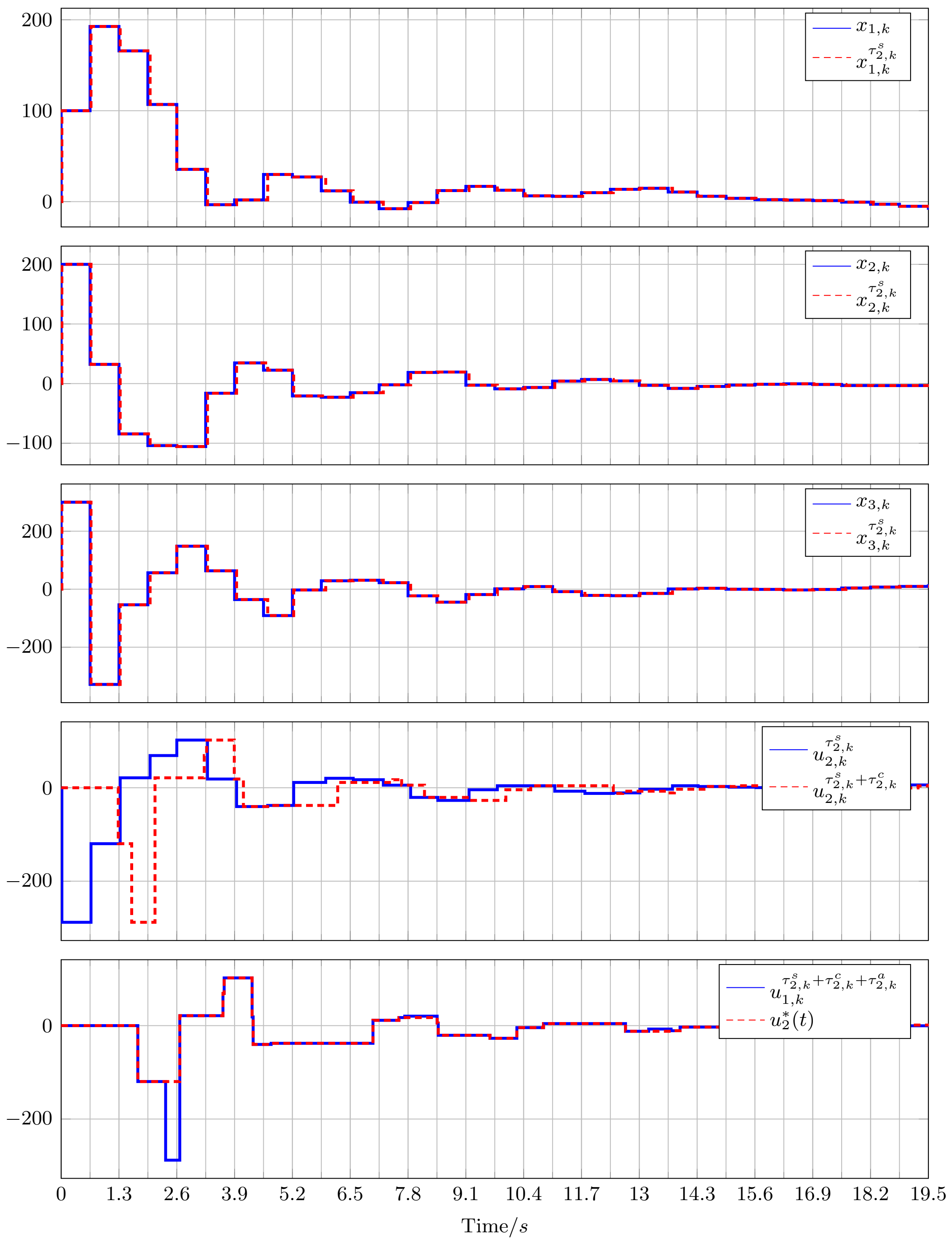


Figure 2.13: Example 2.8: The signals in the second feedback channel obtained with the continuous time simulation.

for the second input channel. With these vectors of arrival times, the system matrix and input vector of the lifted model (2.27) can be computed using (2.30), (2.31) and (2.32a)–(2.32e) for every time step, e.g. in the fourth step, the lifted model is given by

$$\boldsymbol{\xi}_5 = \hat{\boldsymbol{A}}(\boldsymbol{\theta}_{1,4}, \boldsymbol{\theta}_{2,4})\boldsymbol{\xi}_4 + \hat{\boldsymbol{B}}(\boldsymbol{\theta}_{1,4}, \boldsymbol{\theta}_{2,4})\boldsymbol{u}_4 + \hat{\boldsymbol{B}}_f \boldsymbol{f}_4 \tag{2.43}$$

with

$$\hat{\boldsymbol{A}}(\boldsymbol{\theta}_{1,4}, \boldsymbol{\theta}_{2,4}) = \begin{bmatrix} 0.891 & 0.428 & 0.06 & 0.003 & 0.008 & 0 & 0.141 & 0.028 & 0 \\ -0.358 & 0.235 & 0.071 & 0.007 & 0.012 & 0 & 0.41 & 0.018 & 0 \\ -0.424 & -1.135 & -0.189 & -0.01 & -0.027 & 0 & -0.685 & -0.08 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\hat{\boldsymbol{B}}(\boldsymbol{\theta}_{1,4}, \boldsymbol{\theta}_{2,4}) = \begin{bmatrix} 0.007 & 0.041 & 0.107 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$$

$$\hat{\boldsymbol{B}}_f = \begin{bmatrix} 0.018 & 0.06 & 0.071 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.169 & 0.428 & -0.765 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}} .$$

The lifted state vector in the fourth step

$$\boldsymbol{\xi}_4 = \begin{bmatrix} 35.51 & -105.609 & 148.274 & -1.043 & 8.665 & 68.94 & 21.254 & -119.754 & -288.021 \end{bmatrix}^{\mathrm{T}} \tag{2.44}$$

results from the previous iterations with initial condition

$$\boldsymbol{\xi}_0 = \begin{bmatrix} 100 & 200 & 300 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}} . \tag{2.45}$$

Applying control law (2.40) and perturbation (2.42) in (2.43) with $\boldsymbol{\xi}_4$ given in (2.44) yields

$$\boldsymbol{\xi}_5 = \begin{bmatrix} -3.355 & -16.148 & 63.409 & -9.714 & -1.043 & 102.098 & 68.94 & 21.254 & -119.754 \end{bmatrix}^{\mathrm{T}} .$$

Using this iterative approach up to $k = 30$, the discrete time simulation data for the lifted states $\boldsymbol{\xi}_k$ is obtained up to $t = 19.5\,\mathrm{s}$. Figure 2.14 shows the simulation results of the three plant states for the discrete time simulation in blue and for the continuous time simulation in dashed red. Although the discrete time model does not provide any information on the inter-sample behavior, the discrete time data matches the continuous time data at the sampling instances which verifies the effectiveness of the lifted model exactly. To achieve a higher resolution of the discrete time simulation, the sampling time T can be decreased.

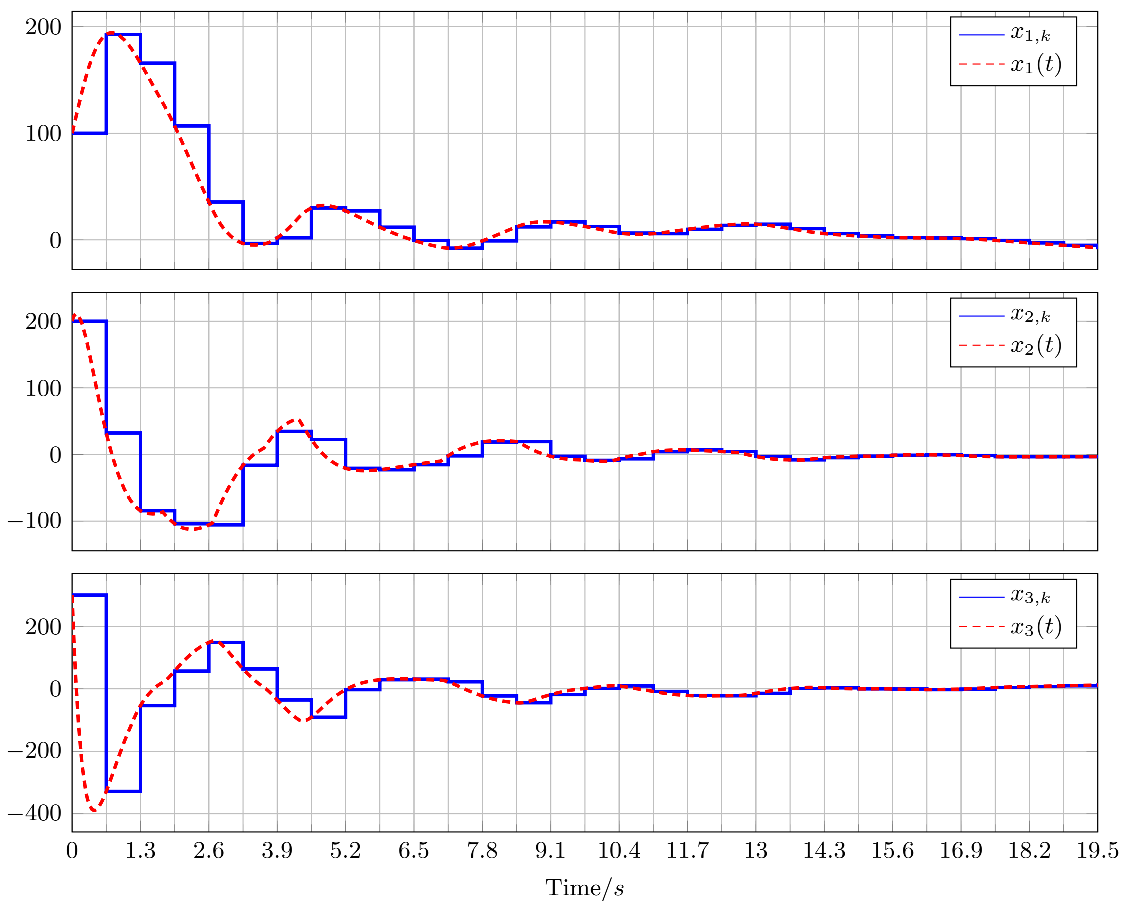


Figure 2.14: Example 2.8: Comparison of the states obtained with the discrete time simulation in blue and the continuous time simulation in dashed red.

Chapter 3

Buffered Networked Control System

Contents

In this chapter, challenges arising from the effects of time-varying delays on the networked control system are discussed and a method to achieve constant round trip times is described.

3.1 Challenges with Time-Varying Delays

A big challenge of the time-varying case is to prove stability of the closed loop system. The first intention could be to show stability of the networked control system with the control law

$$\boldsymbol{u}_k = \hat{\boldsymbol{K}}\boldsymbol{\xi}_k \tag{3.1}$$

by showing that the closed-loop system matrix

$$\hat{\boldsymbol{A}}_{cl}(\boldsymbol{\theta}_{1,k}, \boldsymbol{\theta}_{2,k}, \dots, \boldsymbol{\theta}_{m,k}) = \hat{\boldsymbol{A}}(\boldsymbol{\theta}_{1,k}, \boldsymbol{\theta}_{2,k}, \dots, \boldsymbol{\theta}_{m,k}) - \hat{\boldsymbol{B}}(\boldsymbol{\theta}_{1,k}, \boldsymbol{\theta}_{2,k}, \dots, \boldsymbol{\theta}_{m,k})\hat{\boldsymbol{K}} \tag{3.2}$$

is schur (i.e. has eigenvalues within the unit circle exclusively) for all possible $\boldsymbol{\theta}_{i,k}$. It turns out that this is not a sufficient condition as illustrated by means of numerical simulations in the following two subsections.

Scenario 1

In this subsection a simulation example for a system is shown that behaves unstable even though the closed loop system matrix (3.2) is schur for all possible delay configurations.

Example 3.1. Consider the continuous time plant

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \begin{bmatrix} -2 & 8 \\ 2 & 1 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 3 \\ 0.7 \end{bmatrix} u^* \tag{3.3}$$

with sampling time $T = 0.08$s. Furthermore, assume that the round trip time τ_k can exclusively equal the two distinct values

$$\tau_k \in \{\bar{\tau}_1, \bar{\tau}_2\}, \quad \text{with } \bar{\tau}_1 = 0.01 \text{ and } \bar{\tau}_2 = 0.04. \tag{3.4}$$

As both possible time delays are smaller than the sampling time T, a lifted model for each of the two cases can be derived using theorem 2.5 and is given by

$$\boldsymbol{\xi}_{k+1} = \underbrace{\begin{bmatrix} 0.9 & 0.627 & 0.031 \\ 0.157 & 1.135 & 0.012 \\ 0 & 0 & 0 \end{bmatrix}}_{\hat{\boldsymbol{A}}_1} \boldsymbol{\xi}_k + \underbrace{\begin{bmatrix} 0.212 \\ 0.066 \\ 1 \end{bmatrix}}_{\hat{\boldsymbol{b}}_1} u_k \quad \text{for } \tau_k = \bar{\tau}_1 = 0.01 \tag{3.5}$$

and

$$\boldsymbol{\xi}_{k+1} = \underbrace{\begin{bmatrix} 0.9 & 0.627 & 0.123 \\ 0.157 & 1.135 & 0.045 \\ 0 & 0 & 0 \end{bmatrix}}_{\hat{\boldsymbol{A}}_2} \boldsymbol{\xi}_k + \underbrace{\begin{bmatrix} 0.12 \\ 0.033 \\ 1 \end{bmatrix}}_{\hat{\boldsymbol{b}}_2} u_k \quad \text{for } \tau_k = \bar{\tau}_2 = 0.04. \tag{3.6}$$

Applying the static control law

$$u_k = \hat{\boldsymbol{k}}^{\mathrm{T}} \boldsymbol{\xi}_k = \begin{bmatrix} 8 & 1 & 0 \end{bmatrix} \boldsymbol{\xi}_k$$

results in the closed loop dynamic matrices

$$\hat{\boldsymbol{A}}_{cl,1} = \hat{\boldsymbol{A}}_1 - \hat{\boldsymbol{b}}_1 \hat{\boldsymbol{k}}^{\mathrm{T}} = \begin{bmatrix} -0.797 & 0.415 & 0.031 \\ -0.37 & 1.069 & 0.012 \\ -8 & -1 & 0 \end{bmatrix} \tag{3.7}$$

$$\hat{\boldsymbol{A}}_{cl,2} = \hat{\boldsymbol{A}}_2 - \hat{\boldsymbol{b}}_2 \hat{\boldsymbol{k}}^{\mathrm{T}} = \begin{bmatrix} -0.062 & 0.507 & 0.123 \\ -0.111 & 1.102 & 0.045 \\ -8 & -1 & 0 \end{bmatrix} \tag{3.8}$$

with the corresponding absolute values of its eigenvalues

$$\left| \mathrm{eig}\left(\hat{\boldsymbol{A}}_{cl,1} \right) \right| = \begin{bmatrix} 0.485 \\ 0.485 \\ 0.967 \end{bmatrix} \tag{3.9}$$

$$\left| \mathrm{eig}\left(\hat{\boldsymbol{A}}_{cl,2} \right) \right| = \begin{bmatrix} 0.973 \\ 0.973 \\ 0.966 \end{bmatrix}. \tag{3.10}$$

As (3.9) and (3.10) indicate, both system matrices are schur. This means that the networked control system is asymptotically stable for $\tau_k = \bar{\tau}_1, \forall k$ or $\tau_k = \bar{\tau}_2, \forall k$. This is also depicted in figure 3.1 which shows the simulation results of the two plant states with initial condition $\boldsymbol{\xi}_0 = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix}^{\mathrm{T}}$ and the two constant delay cases. The blue lines correspond to the case in which the time delay equals $\tau_k = \bar{\tau}_1, \forall k$ and the red lines to the case in which the time delay equals $\tau_k = \bar{\tau}_2, \forall k$.

The next question is what happens if the time delay can jump arbitrarily between the two values $\bar{\tau}_1$ and $\bar{\tau}_2$. Take, for instance, the pattern where the time delay alternates between the two possible values in every sampling step, i.e.

$$\tau_{2k} = \bar{\tau}_1 = 0.01, \qquad \tau_{2k+1} = \bar{\tau}_2 = 0.04. \tag{3.11}$$

In this case, the difference equation of the closed loop system is given by

$$\boldsymbol{\xi}_{k+1} = \begin{cases} \hat{\boldsymbol{A}}_{cl,1} \boldsymbol{\xi}_k & \text{if } k \in \{2h | h \in \mathbb{N}\} \\ \hat{\boldsymbol{A}}_{cl,2} \boldsymbol{\xi}_k & \text{if } k \in \{2h+1 | h \in \mathbb{N}\} \end{cases}. \tag{3.12}$$

Considering every second sampling step only, simplifies (3.12) to

$$\boldsymbol{\xi}_{2h+2} = \hat{\boldsymbol{A}}_{cl,2} \hat{\boldsymbol{A}}_{cl,1} \boldsymbol{\xi}_{2h} \quad \text{with } h \in \mathbb{N}. \tag{3.13}$$

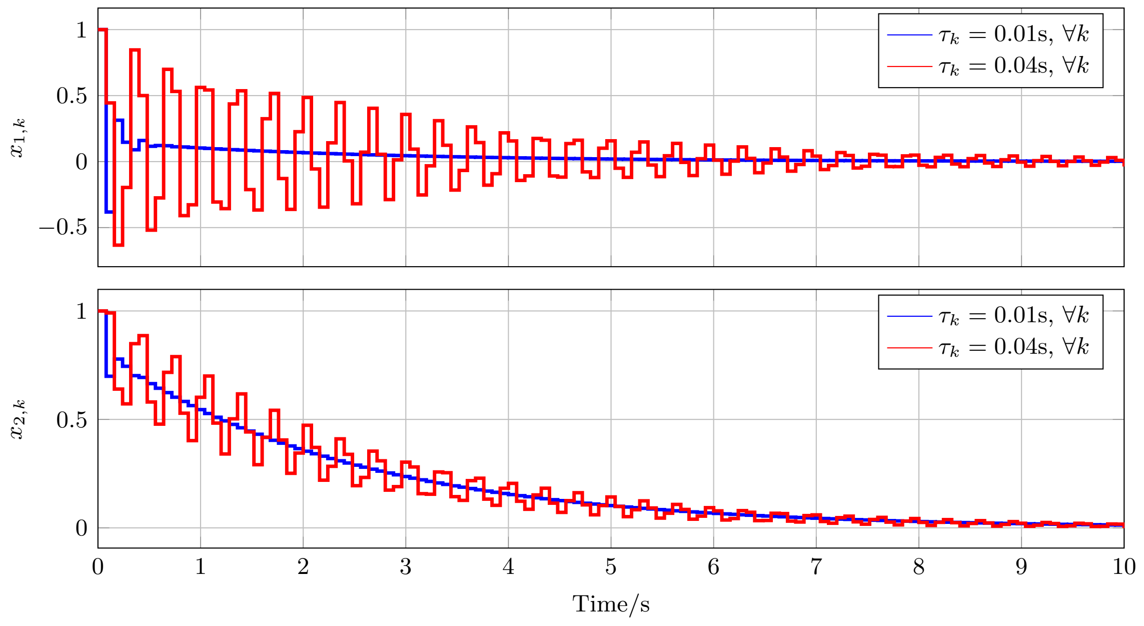


Figure 3.1: Example 3.1: State variables with constant time delays $\bar{\tau}_1$ and $\bar{\tau}_2$.

From (3.13) it is clear that the networked control system is asymptotically stable for the specific delay configuration (3.11) if the system matrix $\hat{\boldsymbol{A}}_{cl,2}\hat{\boldsymbol{A}}_{cl,1}$ is schur. In the present simulation example, this system matrix and the absolute values of its eigenvalues are given by

$$\hat{\boldsymbol{A}}_{cl,2}\hat{\boldsymbol{A}}_{cl,1} = \begin{bmatrix} -1.122 & 0.393 & 0.004 \\ -0.677 & 1.087 & 0.01 \\ 6.744 & -4.388 & -0.262 \end{bmatrix}, \qquad \left| \operatorname{eig}\left(\hat{\boldsymbol{A}}_{cl,2}\hat{\boldsymbol{A}}_{cl,1} \right) \right| = \begin{bmatrix} 0.935 \\ 0.22 \\ 1.012 \end{bmatrix}. \tag{3.14}$$

As (3.14) indicates, not all eigenvalues of $\hat{\boldsymbol{A}}_{cl,2}\hat{\boldsymbol{A}}_{cl,1}$ are within the unit circle. Hence, the networked control system is unstable for the delay pattern given in (3.11). This is also illustrated in figure 3.2 which shows the simulation results of the plant states.

This simulation example shows that the networked control system might be asymptotically stable for a set of constant time delays but this does not imply stability for time-varying delays.

Scenario 2

In this section, a simulation example is given where the networked control system is unstable for a set of constant time delays but is asymptotically stable for a specific sequence of time varying delays.

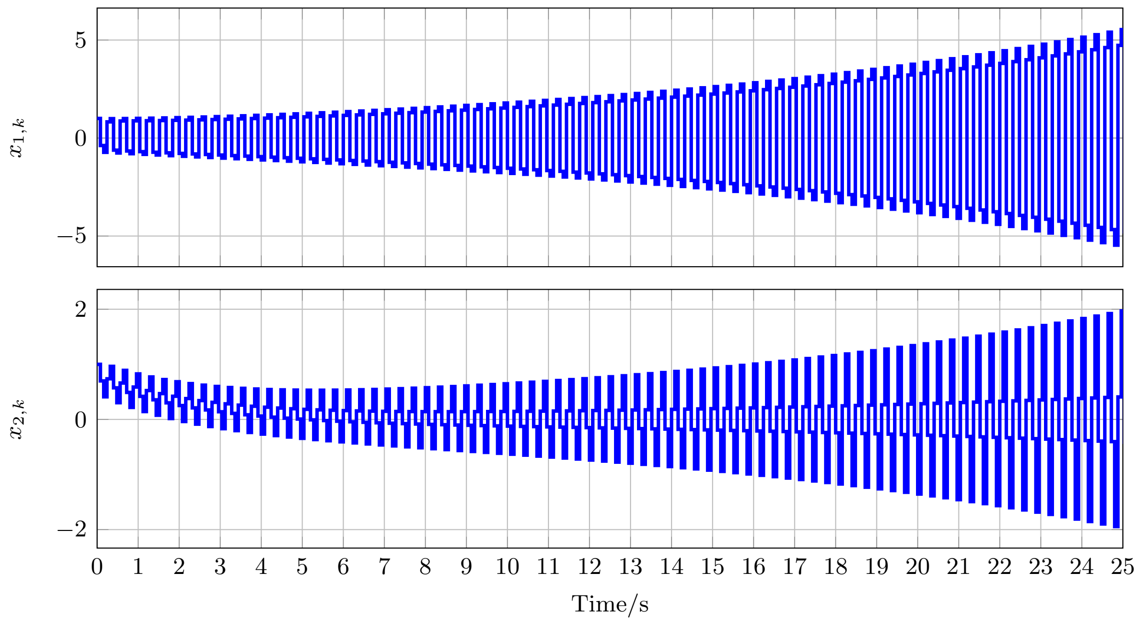


Figure 3.2: Example 3.1: State variables with time delays alternating between $\bar{\tau}_1$ and $\bar{\tau}_2$ in every sampling step.

Example 3.2. Consider the continuous time plant

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \begin{bmatrix} 7 & -23 \\ 28 & -9 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 4 \\ -1 \end{bmatrix} u^* \tag{3.15}$$

with sampling time $T = 1.1\text{s}$. Furthermore, assume that the round trip time τ_k can exclusively equal the two distinct values, i.e.

$$\tau_k \in \{\bar{\tau}_1, \bar{\tau}_2\} \quad \text{with } \bar{\tau}_1 = 0.62 \text{ and } \bar{\tau}_2 = 1.05. \tag{3.16}$$

Applying the static control law

$$u_k = \begin{bmatrix} 3 & 8 & 0 \end{bmatrix} \boldsymbol{\xi}_k \tag{3.17}$$

results in the closed loop system matrices and the corresponding magnitudes of its eigenvalues

$$\hat{\boldsymbol{A}}_{cl,1} = \begin{bmatrix} 0.228 & -0.177 & 0.164 \\ -0.111 & -1.344 & 0.011 \\ -3 & -8 & 0 \end{bmatrix}, \qquad \left|\text{eig}\left(\hat{\boldsymbol{A}}_{cl,1}\right)\right| = \begin{bmatrix} 0.63 \\ 0.63 \\ 1.232 \end{bmatrix} \tag{3.18}$$

$$\hat{\boldsymbol{A}}_{cl,2} = \begin{bmatrix} 0.228 & -0.177 & 0.164 \\ -0.111 & -1.344 & 0.011 \\ -3 & -8 & 0 \end{bmatrix}, \qquad \left|\text{eig}\left(\hat{\boldsymbol{A}}_{cl,2}\right)\right| = \begin{bmatrix} 0.274 \\ 1.203 \\ 1.203 \end{bmatrix}. \tag{3.19}$$

The networked control system is unstable for both constant delay cases since neither $\hat{\boldsymbol{A}}_{cl,1}$ nor $\hat{\boldsymbol{A}}_{cl,2}$ are schur. This unstable behavior is also reflected by a numerical simulation.

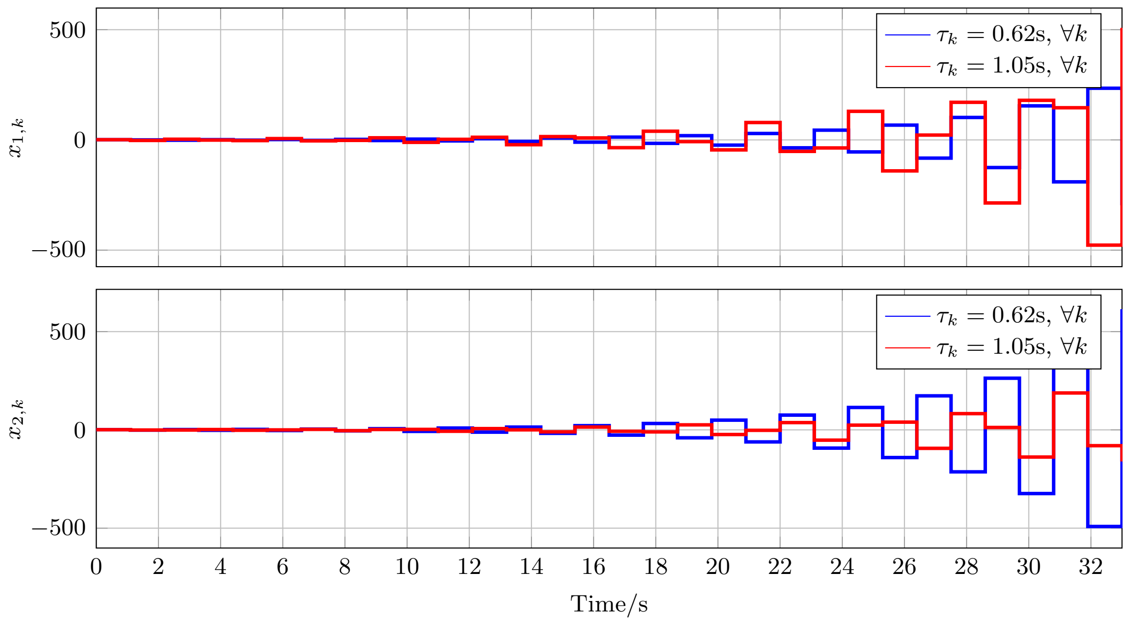


Figure 3.3: Example 3.2: State variables with constant time delays $\bar{\tau}_1$ and $\bar{\tau}_2$.

The simulation results in figure 3.3 show the plant states for the two constant delay cases $\tau_k = \bar{\tau}_1, \forall k$ in red and $\tau_k = \bar{\tau}_2, \forall k$ in blue.

The case in which the time delay alternates between the two possible time delays in every sampling step, i.e.

$$\tau_{2k} = \bar{\tau}_1 = 0.62, \qquad \tau_{2k+1} = \bar{\tau}_2 = 1.05, \tag{3.20}$$

can again be analyzed by considering system (3.13). The system matrix $\hat{\boldsymbol{A}}_{cl,2}\hat{\boldsymbol{A}}_{cl,1}$ and the magnitudes of its eigenvalues are given by

$$\hat{\boldsymbol{A}}_{cl,2}\hat{\boldsymbol{A}}_{cl,1} = \begin{bmatrix} 0.309 & 3.271 & -0.096 \\ -0.142 & 0.336 & 0.009 \\ 0.205 & 11.28 & -0.579 \end{bmatrix}, \qquad \left|\text{eig}\left(\hat{\boldsymbol{A}}_{cl,2}\hat{\boldsymbol{A}}_{cl,1}\right)\right| = \begin{bmatrix} 0.497 \\ 0.625 \\ 0.625 \end{bmatrix}. \tag{3.21}$$

Although the network control system is unstable for the two constant delay cases, it turns out that the network control system is asymptotically stable if the time delay alternates between the two delays in every sampling step since the system matrix in (3.21) is schur. This stabilizing behavior is also evident in the simulation results for the plant states as depicted in figure 3.2.

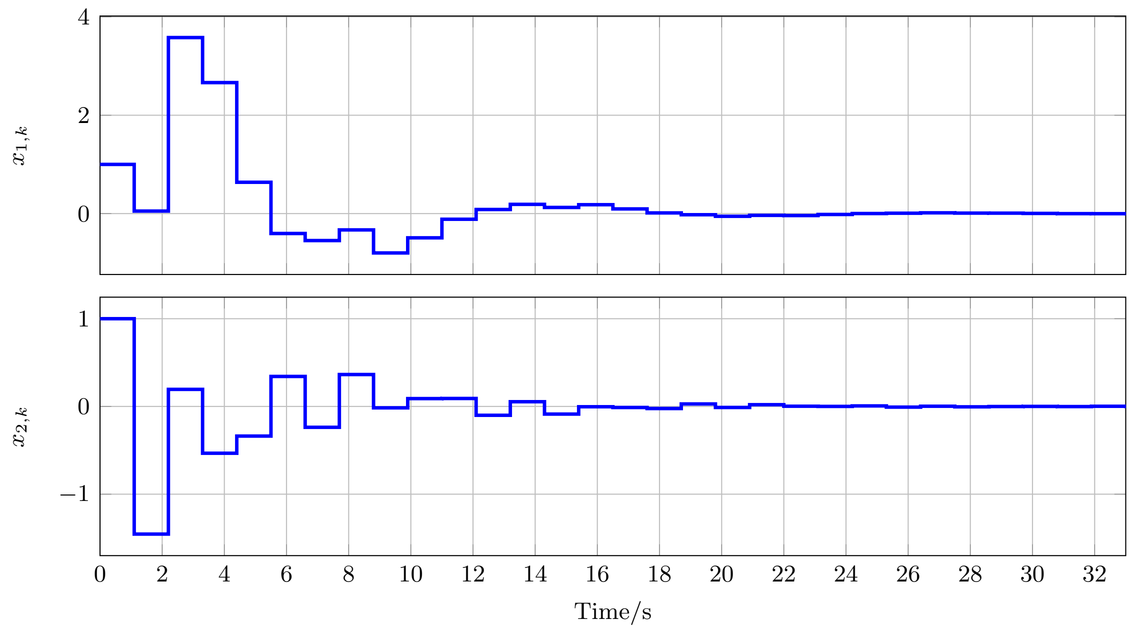


Figure 3.4: Example 3.2: State variables with time delays alternating between $\bar{\tau}_1$ and $\bar{\tau}_2$ in every sampling step.

3.2 The Buffered Network Control System

As motivated in section 3.1, the stability proof for network control systems affected by time-varying delays is quite challenging. There are some methods in literature (e.g. [22, 6]) which deal with this time-varying case. However, these methods are mathematically demanding, difficult to implement and can lead to conservative control laws. Consequently, a buffering mechanism is used in this thesis which ensures constant round trip times. This is achieved by implementing buffers directly at the receiving end of each feedback channel as indicated in the block diagram shown in figure 3.5.

The sensor attaches a timestamp to each sent message which is not altered by the controller and therefore received by the i^{th} buffer delayed by $\tau_{i,k}$. Due to a time synchronization mechanism (e.g. one of the methods proposed in [24, 25]) between the buffers and the sensor, the i^{th} buffer is capable of computing the time delay of the current packet $\tau_{i,k}$. As an upper bound of this time delay $\delta_i T$ is known, the i^{th} buffer can introduce the additional time delay

$$\tau_{i,k}^{b} = \delta_i T - \tau_{i,k} \tag{3.22}$$

before forwarding the data to the zero-order hold. This mechanism ensures that the round trip time (the delay from the sensor to the zero-order hold) is constantly the maximum time delay $\delta_i T$. Using these constant round trip times in (2.28) results in the arrival times

$$t_{i,k}^{j} = \min\{\max\left[0, jT\right], \max\left[0, (j+1)\,T\right], \ldots, \max\left[0, \delta_i T\right], T\} \qquad j = 0, 1, \ldots, \delta_i + 1$$

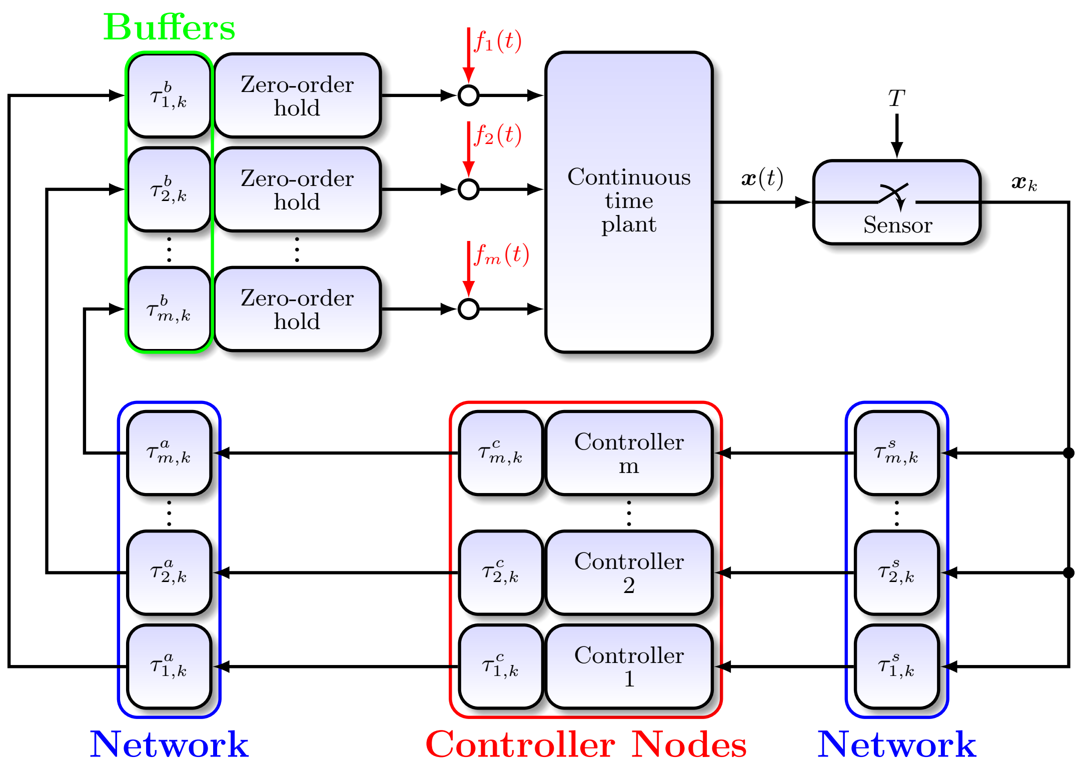


Figure 3.5: Buffered network control system.

which evaluates to

$$t_{i,k}^{0}=0,\qquad t_{i,k}^{1}=T,\qquad t_{i,k}^{2}=T,\qquad \cdots,\qquad t_{i,k}^{\delta_i+1}=T.$$

Applying these arrival times to (2.36) results in the discrete time model of the networked control system

$$\boldsymbol{x}_{k+1}=\boldsymbol{A}\boldsymbol{x}_k+\sum_{i=1}^{m}\boldsymbol{b}_i u_{i,k-\delta_i}+\boldsymbol{B}\boldsymbol{f}_k \tag{3.23}$$

with

$$\boldsymbol{A}=e^{\boldsymbol{A}_cT},\qquad \boldsymbol{B}=\begin{bmatrix}\boldsymbol{b}_1 & \boldsymbol{b}_2 & \cdots & \boldsymbol{b}_m\end{bmatrix}=\int_0^T e^{\boldsymbol{A}_cs}\boldsymbol{B}_c ds. \tag{3.24}$$

Defining the lifted state vector

$$\boldsymbol{\xi}_k=\begin{bmatrix}\boldsymbol{x}_k^{\mathrm{T}} & u_{1,k-1} & u_{1,k-2} & \cdots & u_{1,k-\delta_1} & \cdots & u_{m,k-1} & \cdots & u_{m,k-\delta_m}\end{bmatrix}^{\mathrm{T}} \tag{3.25}$$

results in the lifted model

$$\boldsymbol{\xi}_{k+1}=\hat{\boldsymbol{A}}\boldsymbol{\xi}_k+\hat{\boldsymbol{B}}\boldsymbol{u}_k+\hat{\boldsymbol{B}}_f\boldsymbol{f}_k \tag{3.26}$$

with

$$\hat{\boldsymbol{A}}=\begin{bmatrix}
\boldsymbol{A} & \boldsymbol{0}_{n\times\delta_1-1} & \boldsymbol{b}_1 & \boldsymbol{0}_{n\times\delta_2-1} & \boldsymbol{b}_2 & \cdots & \boldsymbol{0}_{n\times\delta_m-1} & \boldsymbol{b}_m\\
\boldsymbol{0}_{1\times n} & \boldsymbol{0} & 0 & \boldsymbol{0} & 0 & \cdots & \boldsymbol{0} & 0\\
\boldsymbol{0}_{\delta_1-1\times n} & \boldsymbol{I}_{\delta_1-1} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0}\\
\boldsymbol{0}_{1\times n} & \boldsymbol{0} & 0 & \boldsymbol{0} & 0 & \cdots & \boldsymbol{0} & 0\\
\boldsymbol{0}_{\delta_2-1\times n} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I}_{\delta_2-1} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0}\\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots\\
\boldsymbol{0}_{1\times n} & \boldsymbol{0} & 0 & \boldsymbol{0} & 0 & \cdots & \boldsymbol{0} & 0\\
\boldsymbol{0}_{\delta_m-1\times n} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{I}_{\delta_m-1} & \boldsymbol{0}
\end{bmatrix} \tag{3.27}$$

$$\hat{\boldsymbol{B}}=\begin{bmatrix}
\boldsymbol{0}_{n\times1} & \boldsymbol{0}_{n\times1} & \cdots & \boldsymbol{0}_{n\times1}\\
1 & 0 & \cdots & 0\\
\boldsymbol{0}_{\delta_1-1\times1} & \boldsymbol{0} & \cdots & \boldsymbol{0}\\
0 & 1 & \cdots & 0\\
\boldsymbol{0}_{\delta_2-1\times1} & \boldsymbol{0} & \cdots & \boldsymbol{0}\\
\vdots & \vdots & \ddots & \vdots\\
0 & 0 & \cdots & 1\\
\boldsymbol{0}_{\delta_m-1\times1} & \boldsymbol{0} & \cdots & \boldsymbol{0}
\end{bmatrix}=\begin{bmatrix}\hat{\boldsymbol{b}}_1 & \hat{\boldsymbol{b}}_2 & \cdots & \hat{\boldsymbol{b}}_m\end{bmatrix} \tag{3.28}$$

$$\hat{\boldsymbol{B}}_f=\begin{bmatrix}\boldsymbol{B}\\ \boldsymbol{0}\\ \boldsymbol{0}\\ \boldsymbol{0}\\ \boldsymbol{0}\\ \vdots\\ \boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix}=\begin{bmatrix}\hat{\boldsymbol{b}}_{f,1} & \hat{\boldsymbol{b}}_{f,2} & \cdots & \hat{\boldsymbol{b}}_{f,m}\end{bmatrix}. \tag{3.29}$$

For cases in which a non-integer upper bound of the time delay $\tilde{\delta}_i$ is known, i.e.

$$\tau_{i,k} \leq \tilde{\delta}_i T < \delta_i T \quad \forall k \tag{3.30}$$

with $\tilde{\delta}_i \in \mathbb{R}^+$ and $\delta_i = \left\lceil \tilde{\delta}_i \right\rceil$, the delay introduced by the buffer can be chosen as

$$\tau^b_{i,k} = \tilde{\delta}_i T - \tau_{i,k}, \tag{3.31}$$

which results in a constant round trip time of $\tilde{\delta}_i T$ that is not an integer multiple of the sampling time. This constant time delay ensures that exactly one new sample arrives at the plant in every sampling step (see figure 3.6 for a timing diagram with constant time delay $\tau_{i,k} + \tau^b_{i,k} = 2.3T$). The constant time period from the actual sampling instant to the time instant at which the new control signal is applied is symbolized by τ^*_i.

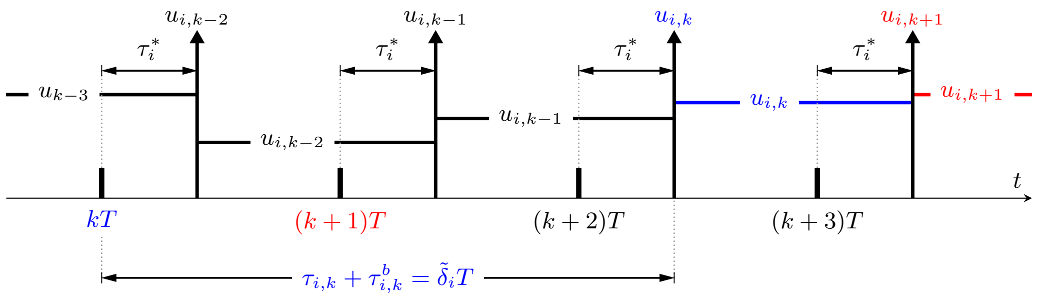


Figure 3.6: Timing diagram for the ith channel with constant time delay $\tau_{i,k} + \tau^b_{i,k} = 2.3T$ achieved by the buffering mechanism.

In this case, the constant time delay is given by

$$\tilde{\delta}_i T = (\delta_i - 1)T + \tau^*_i. \tag{3.32}$$

Using these constant round trip times in (2.28) results in the arrival times

$$t^j_{i,k} = \min\{\max[0, (j-1)T + \tau^*_i], \max[0, jT + \tau^*_i], \ldots, \max[0, (\delta_i - 1)T + \tau^*_i], T\}$$
$$j = 0, 1, \ldots, \delta_i + 1$$

which evaluates to

$$t^0_{i,k} = 0, \qquad t^1_{i,k} = \tau^*_i, \qquad t^2_{i,k} = T, \qquad \cdots, \qquad t^{\delta_i+1}_{i,k} = T.$$

Applying these arrival times to (2.36) results in the discrete time model of the networked control system

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \sum_{i=1}^{m} \left(\bar{\boldsymbol{b}}_i u_{i,k-\delta_i} + \tilde{\boldsymbol{b}}_i u_{i,k-\delta_i+1} \right) + \boldsymbol{B}\boldsymbol{f}_k \tag{3.33}$$

with $\boldsymbol{A}$, $\boldsymbol{B}$ as in (3.24) and

$$\bar{\boldsymbol{B}} = \begin{bmatrix} \bar{\boldsymbol{b}}_1 & \bar{\boldsymbol{b}}_2 & \cdots & \bar{\boldsymbol{b}}_m \end{bmatrix} = e^{\boldsymbol{A}_c T} \int_0^{\tau_i^*} e^{\boldsymbol{A}_c s} \boldsymbol{B}_c ds \tag{3.34}$$

$$\tilde{\boldsymbol{B}} = \begin{bmatrix} \tilde{\boldsymbol{b}}_1 & \tilde{\boldsymbol{b}}_2 & \cdots & \tilde{\boldsymbol{b}}_m \end{bmatrix} = \boldsymbol{B} - \bar{\boldsymbol{B}} = e^{\boldsymbol{A}_c T} \int_{\tau_i^*}^{T} e^{\boldsymbol{A}_c s} \boldsymbol{B}_c ds \tag{3.35}$$

$$\tau_i^* = (\tilde{\delta}_i - \delta_i + 1)T. \tag{3.36}$$

Again, using the lifted state vector (3.25) results in the lifted model

$$\boldsymbol{\xi}_{k+1} = \hat{\boldsymbol{A}} \boldsymbol{\xi}_k + \hat{\boldsymbol{B}} \boldsymbol{u}_k + \hat{\boldsymbol{B}}_f \boldsymbol{f}_k \tag{3.37}$$

with $\hat{\boldsymbol{B}}$, $\hat{\boldsymbol{B}}_f$ as in (3.28) and (3.29) and

$$\hat{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{0}_{n\times\delta_1-2} & \tilde{\boldsymbol{B}}_1 & \boldsymbol{0}_{n\times\delta_2-2} & \tilde{\boldsymbol{B}}_2 & \cdots & \boldsymbol{0}_{n\times\delta_m-2} & \tilde{\boldsymbol{B}}_m \\ \boldsymbol{0}_{1\times n} & \boldsymbol{0} & 0 & \boldsymbol{0} & 0 & \cdots & \boldsymbol{0} & 0 \\ \boldsymbol{0}_{\delta_1-1\times n} & \boldsymbol{I}_{\delta_1-1} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0}_{1\times n} & \boldsymbol{0} & 0 & \boldsymbol{0} & 0 & \cdots & \boldsymbol{0} & 0 \\ \boldsymbol{0}_{\delta_2-1\times n} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I}_{\delta_2-1} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \boldsymbol{0}_{1\times n} & \boldsymbol{0} & 0 & \boldsymbol{0} & 0 & \cdots & \boldsymbol{0} & 0 \\ \boldsymbol{0}_{\delta_m-1\times n} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{I}_{\delta_m-1} & \boldsymbol{0} \end{bmatrix} \tag{3.38}$$

$$\tilde{\boldsymbol{B}}_i = \begin{bmatrix} \bar{\boldsymbol{b}}_i & \tilde{\boldsymbol{b}}_i \end{bmatrix}.$$

As the time delays induced by the networked feedback and the buffer are explicitly considered in the discrete time models given in (3.26) and (3.37), stabilizing control laws for buffered networked control systems can be designed based on these models.

Chapter 4

Discrete Time Sliding Mode Control

Contents

This chapter gives a short introduction to the concepts of sliding mode control with special focus on the differences between continuous time and discrete time methods.

The theory of sliding modes originates from the theory of variable structure systems, which are systems whose structure can change depending on the system's current state values. A simple example of a variable structure system (see [26]) is a unity feedback loop with the control law

$$u(t) = \begin{cases} K_1 e(t) & \text{for } |e(t)| > \varepsilon \\ K_2 \int_{t_\varepsilon}^{t} e(t) dt & \text{else} \end{cases} \tag{4.1}$$

with the error $e(t)$, $\varepsilon \in \mathbb{R}^+$ and $|e(t)| < \varepsilon$ for $t \geq t_\varepsilon$. This control law combines the advantages of fast convergence for large initial values of the proportional controller with the advantage of the zero steady state error of the integrating controller. Additionally, the disadvantage of high overshoot with the integrating controller is suppressed as the integrating control law is active only if the error $e(t)$ is small. One can imagine that there are numerous possible combinations and switching rules, which results in many methods and algorithms. One specific class of algorithms, however, received special attention in the last decades, the so called sliding mode control. The most significant advantage of this class of algorithms is the possibility to enforce motions that are insensitive to a wide class of disturbances.

4.1 Sliding Modes – Continuous vs. Discrete Time

To understand the concept of continuous time sliding modes, consider the second order system

$$\begin{aligned} \frac{\mathrm{d}x_1}{\mathrm{d}t} &= x_2 \\ \frac{\mathrm{d}x_2}{\mathrm{d}t} &= u + f \end{aligned} \tag{4.2}$$

with state variables x_1, x_2, input $u(t)$ and bounded perturbation $|f(t)| < 1$. Furthermore, consider the control law [26]

$$u = -cx_2 - \operatorname{sign}(\sigma) \tag{4.3}$$

with the so-called sliding variable

$$\sigma = cx_1 + x_2 \tag{4.4}$$

and $c > 0$. Note that

$$\operatorname{sign}(\sigma) = \begin{cases} 1 & \text{for } \sigma > 0 \\ \begin{bmatrix} -1 & 1 \end{bmatrix} & \text{for } \sigma = 0 \\ -1 & \text{for } \sigma < 0 \end{cases} . \tag{4.5}$$

There are some mathematical tools in literature to formally analyze controllers using this set-valued function. The interested reader is referred to [27, 28, 29, 30]. To gain insight in the properties of this system, numerical simulations with the three perturbations

$$f(t) = 0, \qquad f(t) = -0.6, \qquad f(t) = 0.8 \sin(5t) \tag{4.6}$$

are performed. The resulting trajectories, starting from the initial conditions $\boldsymbol{x}_0^{(1)} = \begin{bmatrix}-5 & -5\end{bmatrix}^{\mathrm{T}}$ and $\boldsymbol{x}_0^{(2)} = \begin{bmatrix}5 & 5\end{bmatrix}^{\mathrm{T}}$ with $c = 0.5$, are depicted in figure 4.1. For the initial condition $\boldsymbol{x}_0^{(2)}$ and the different perturbations, the time evolution of the states x_1, x_2 and the sliding variable σ are shown in figure 4.2.

These figures show the most important property of sliding mode approaches, i.e. that the sliding variable σ approaches $\sigma = 0$ in finite time and stays at zero independently of the perturbation (see figure 4.2). Figure 4.1 illustrates this property even more clearly because the trajectories are only influenced by the disturbance from the initial condition until the sliding surface (i.e. $\sigma = 0$) is reached in finite time (reaching phase). After reaching the sliding surface, the states "slide" along the sliding surface to the origin unaffected by the disturbance (sliding phase). This property can also be illustrated mathematically. Considering the derivative of the sliding variable σ and the plant dynamics (4.2) results in

$$\frac{\mathrm{d}\sigma}{\mathrm{d}t} = -\operatorname{sign}(\sigma) + f. \tag{4.7}$$

Since $|f| < 1$, the perturbation is always dominated by the $\operatorname{sign}(\sigma)$ part. Therefore, $\sigma = 0$ is achieved in finite time and maintained for the remaining time (see [26]). Thus,

$$\sigma = 0 = \frac{\mathrm{d}x_1}{\mathrm{d}t} + cx_1 \Rightarrow \frac{\mathrm{d}x_1}{\mathrm{d}t} = -cx_1 \tag{4.8}$$

are the remaining dynamics in sliding mode. From (4.8) it can be concluded that the motion enforced in sliding mode is insensitive to the disturbance $f(t)$ as it does not appear in (4.8).

In real world applications, the control law has to be implemented in a discrete time setting. Hence, the properties of discrete time sliding modes and the differences to its continuous time counterpart have to be discussed. For a discrete time setting, consider the second order system

$$\begin{aligned} x_{1,k+1} &= x_{1,k} + Tx_{2,k} \\ x_{2,k+1} &= x_{2,k} + T(u_k + f_k) \end{aligned} \tag{4.9}$$

which is the Euler discretized version of (4.2) with sampling time T.
Performing the exact discretization of (4.8) gives

$$x_{1,k+1} = e^{cT} x_{1,k}. \tag{4.10}$$

The discrete time sliding variable

$$\sigma_k = \frac{1 - e^{cT}}{T} x_{1,k} + x_{2,k} \tag{4.11}$$

is then defined as the deviation between (4.10) and the first state in (4.9). The forward increment of (4.11) using (4.9) is given by

$$\sigma_{k+1} = \frac{1 - e^{cT}}{T} x_{1,k} + (2 - e^{cT}) x_{2,k} + T(u_k + f_k). \tag{4.12}$$

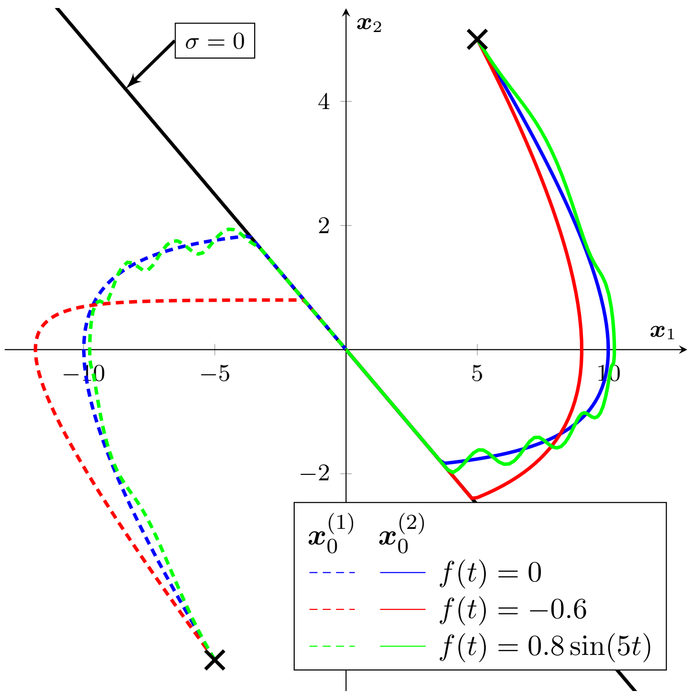


Figure 4.1: Trajectories of system (4.2) affected by different perturbations with controller (4.3) and (4.4) where $c = 0.5$ and initial conditions $\boldsymbol{x}_0^{(1)}$ and $\boldsymbol{x}_0^{(2)}$ are used.

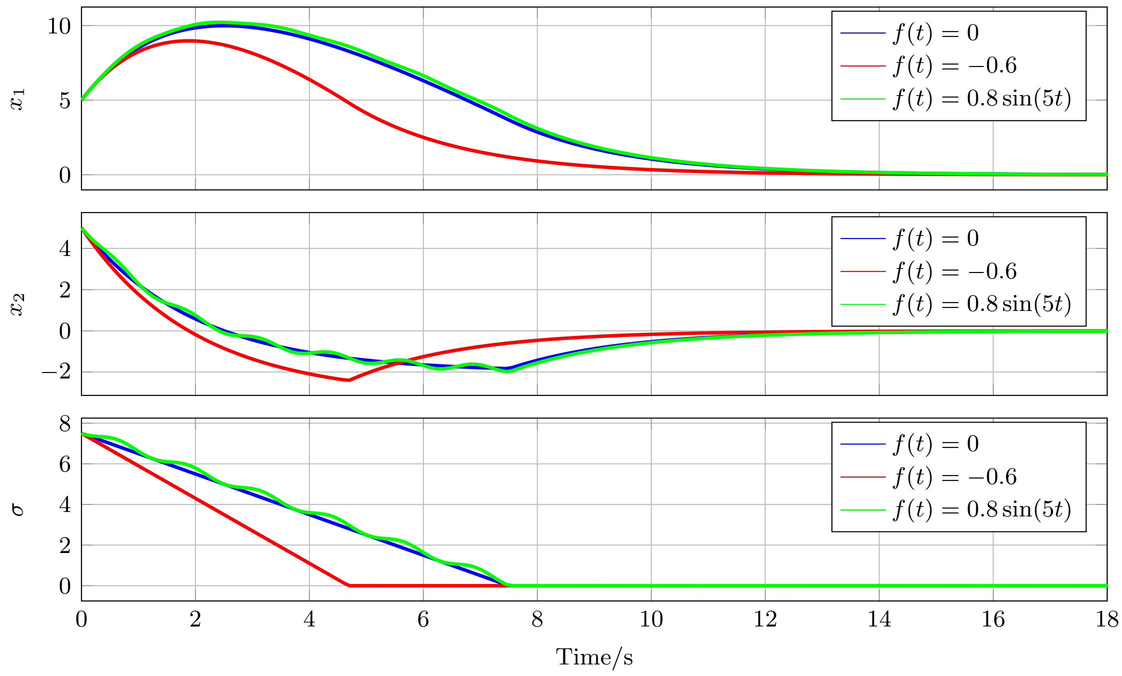


Figure 4.2: State variables x_1, x_2 and sliding variable σ of system (4.2) affected by different perturbations with controller (4.3) and (4.4) where $c = 0.5$ and initial condition $\boldsymbol{x}_0^{(2)}$ is used.

Performing Euler discretization of (4.7) yields

$$\sigma_{k+1} = \sigma_k - T \operatorname{sign}(\sigma_k) + T f_k. \tag{4.13}$$

Applying (4.12) to (4.13) results in the control law

$$u_k = -\frac{1}{T}\left(\frac{1-e^{cT}}{T} x_{1,k} + \left(2 - e^{cT}\right) x_{2,k} - \sigma_k\right) - \operatorname{sign}(\sigma_k). \tag{4.14}$$

Simulation results of the closed loop system with $c = -0.05$, $f_k = 0.8\sin(kT)$ and initial conditions $\boldsymbol{x}_0^{(1)} = \begin{bmatrix}-5 & -5\end{bmatrix}^{\mathrm{T}}$ and $\boldsymbol{x}_0^{(2)} = \begin{bmatrix}5 & 5\end{bmatrix}^{\mathrm{T}}$ are shown in figures 4.3 and 4.4 for different sampling times T. As these figures illustrate, it is not possible to achieve an ideal sliding mode in discrete time setting as in the continuous time case in which the motion is completely insensitive to the disturbance. Therefore, the concept of quasi-sliding mode is introduced for discrete time systems.

Definition 4.1. *A system performs a quasi-sliding mode if the sliding variable σ_k approaches a band around zero in a finite number of steps k^* and remains inside this band for $k \geq k^*$, i.e.*

$$|\sigma_k| \leq \Delta \quad \text{for} \quad k \geq k^*. \tag{4.15}$$

The band Δ is denoted as quasi-sliding mode band.
In the multi-input case, the quasi-sliding mode band is defined for each component of the sliding variable individually, i.e.

$$|\sigma_{i,k}| \leq \Delta_i \quad \text{for} \quad k \geq k_i^*, \quad i = 1, 2, \ldots m. \tag{4.16}$$

Definition 4.2. *A system performs an ideal quasi-sliding mode if the sliding variable σ_k approaches zero in a finite number of steps k^* and remains at zero for all subsequent steps, i.e.*

$$\sigma_k = 0 \quad \text{for} \quad k \geq k^*. \tag{4.17}$$

The remaining disturbance-insensitive dynamics are denoted as ideal quasi-sliding mode dynamics.

Remark 4.3. In literature, however, a different definition of the quasi-sliding mode [31] exists. According to this definition, the sliding variable should cross and re-cross the zero line in every successive sampling step inside the quasi-sliding mode band. From a practical point of view, such a motion is not desired since it leads to high-frequency oscillations. Thus, definition 4.1 is used in this dissertation.

The simulation results depicted in figures 4.3 and 4.4 show that the accuracy significantly increases with decreasing sampling time T. With decreasing sampling time, the switching frequency around the sliding variable increases. As a result, an infinitesimally small sampling time leads to an infinite switching frequency with an infinitesimally small quasi-sliding mode band. As the continuous sliding mode can be considered to be switching around the sliding surface with infinite frequency and infinitesimal amplitude, this corresponds to the continuous sliding mode.

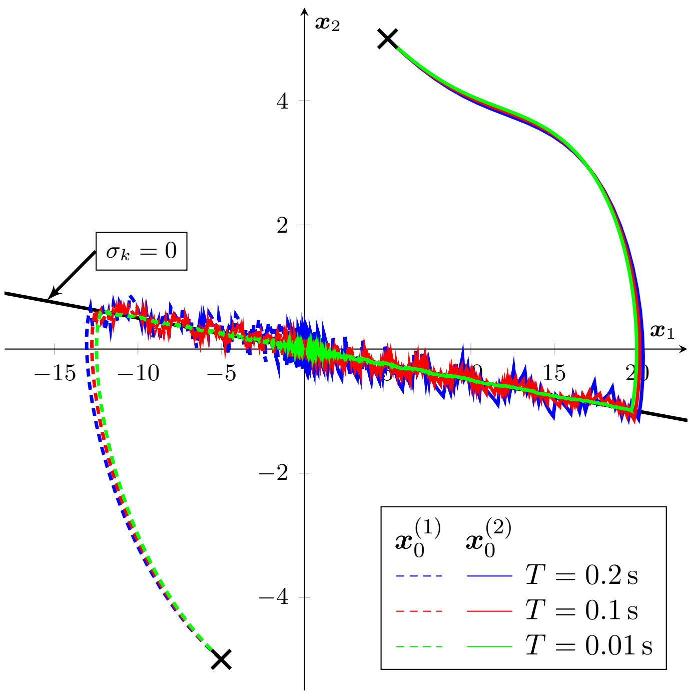


Figure 4.3: Trajectories of system (4.9) with controller (4.11) and (4.14) for different sampling times T and initial conditions $x_0^{(1)}$ and $x_0^{(2)}$.

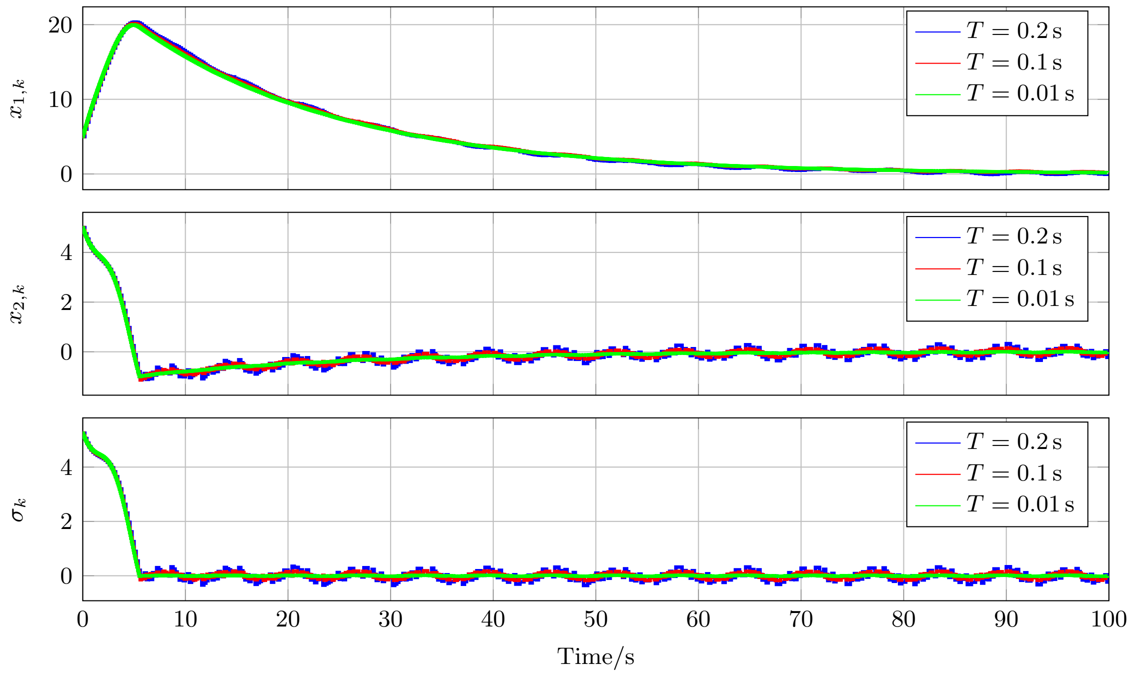


Figure 4.4: State variables and sliding variable σ_k of system (4.9) with controller (4.11) and (4.14) for different sampling times T and initial condition $x_0^{(2)}$.

4.2 Sliding Variable Design

Consider the discrete time system

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k \tag{4.18}$$

with state vector $\boldsymbol{x}_k \in \mathbb{R}^n$ and vector of inputs $\boldsymbol{u}_k \in \mathbb{R}^m$. The classical choice of the sliding variable for such systems is a linear combination of the states, i.e.

$$\boldsymbol{\sigma}_k = \boldsymbol{M}\boldsymbol{x}_k \tag{4.19}$$

to ensure that the ideal quasi-sliding mode dynamics are asymptotically stable. In this section, two approaches to design sliding variables are presented. The first approach is based on the Ackermann formula and can be used to design sliding surfaces for single-input systems. The second approach is based on a state transformation and is also applicable in the multi-input case. Note that both methods result in sliding variables of relative degree $\gamma = 1$, i.e. σ_{k+1} directly depends on u_k.

4.2.1 Ackermann's Formula for Sliding Surface Design

Consider the discrete time single-input system

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{b}u_k \tag{4.20}$$

with $\boldsymbol{x}_k \in \mathbb{R}^n$ and $u_k \in \mathbb{R}$. Matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$ are of appropriate dimensions.

Ackermann's formula [32] is adapted in such a way that the sliding variable

$$\sigma_k = \boldsymbol{m}^{\mathrm{T}}\boldsymbol{x}_k \tag{4.21}$$

can be designed by specifying the $n-1$ eigenvalues of the ideal quasi-sliding mode dynamics. The result is summarized in theorem 4.4.

Theorem 4.4. Consider plant (4.20) and sliding variable (4.21). If

$$\boldsymbol{m}^{\mathrm{T}} = \boldsymbol{t}_1^{\mathrm{T}} P_1(\boldsymbol{A}) \tag{4.22}$$

with

$$\begin{aligned} P_1(\lambda) &= (\lambda-\lambda_1)(\lambda-\lambda_2)\cdots(\lambda-\lambda_{n-1}) = p_0 + p_1\lambda + \cdots + p_{n-2}\lambda^{n-2} + \lambda^{n-1} \\ \boldsymbol{t}_1^{\mathrm{T}} &= \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{b} & \boldsymbol{A}\boldsymbol{b} & \cdots & \boldsymbol{A}^{n-1}\boldsymbol{b} \end{bmatrix}^{-1} \end{aligned} \tag{4.23}$$

then $\lambda_1, \lambda_2, \ldots, \lambda_{n-1}$ are the eigenvalues of the ideal quasi-sliding mode dynamics (see [33]).

Proof. In ideal quasi-sliding mode, the sliding variable is zero for all subsequent steps, i.e.

$$\sigma_k = \boldsymbol{m}^{\mathrm{T}}\boldsymbol{x}_k = 0 \tag{4.24}$$

$$\sigma_{k+1} = \boldsymbol{m}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{m}^{\mathrm{T}}\boldsymbol{b}u_k = 0 \tag{4.25}$$

Applying (4.22) and (4.23) gives

$$\begin{aligned}
\boldsymbol{m}^{\mathrm{T}}\boldsymbol{b} &= \boldsymbol{t}_1^{\mathrm{T}} P_1(\boldsymbol{A})\boldsymbol{b} \\
&= \boldsymbol{t}_1^{\mathrm{T}} \underbrace{\left(p_0\boldsymbol{b} + p_1\boldsymbol{A}\boldsymbol{b} + \cdots p_{n-2}\boldsymbol{A}^{n-2}\boldsymbol{b} + \boldsymbol{A}^{n-1}\boldsymbol{b}\right)}_{P_1(\boldsymbol{A})\boldsymbol{b}} \\
&= \boldsymbol{t}_1^{\mathrm{T}} \begin{bmatrix}\boldsymbol{b} & \boldsymbol{A}\boldsymbol{b} & \cdots & \boldsymbol{A}^{n-1}\end{bmatrix} \begin{bmatrix}p_0 & p_1 & \cdots & p_{n-2} & 1\end{bmatrix}^{\mathrm{T}} \qquad (4.26) \\
&= \underbrace{\begin{bmatrix}0 & \cdots & 0 & 1\end{bmatrix} \begin{bmatrix}\boldsymbol{b} & \boldsymbol{A}\boldsymbol{b} & \cdots & \boldsymbol{A}^{n-1}\end{bmatrix}^{-1}}_{\boldsymbol{t}_1^{\mathrm{T}}} \begin{bmatrix}\boldsymbol{b} & \boldsymbol{A}\boldsymbol{b} & \cdots & \boldsymbol{A}^{n-1}\end{bmatrix} \begin{bmatrix}p_0 & \cdots & 1\end{bmatrix}^{\mathrm{T}} \\
&= 1.
\end{aligned}$$

Solving (4.25) for u_k and exploiting (4.26) results in

$$u_k = -\boldsymbol{m}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}_k. \tag{4.27}$$

Using (4.22) in (4.27) gives

$$u_k = -\boldsymbol{t}_1^{\mathrm{T}} \underbrace{P_1(\boldsymbol{A})\boldsymbol{A}}_{P(\boldsymbol{A})} \boldsymbol{x}_k = -\boldsymbol{k}^{\mathrm{T}}\boldsymbol{x}_k \tag{4.28}$$

with

$$P(\lambda) = P_1(\lambda)\lambda = (\lambda - \lambda_1)(\lambda - \lambda_2)\cdots(\lambda - \lambda_{n-1})\lambda \tag{4.29}$$

$$\boldsymbol{k}^{\mathrm{T}} = \boldsymbol{t}_1^{\mathrm{T}} P(\boldsymbol{A}). \tag{4.30}$$

Consequently, the control law given in (4.28) can be seen as a classical state feedback controller, designed with Ackermann's formula where the eigenvalues of $\boldsymbol{A} - \boldsymbol{b}\boldsymbol{k}^{\mathrm{T}}$ are placed at $\lambda_1, \lambda_2, \ldots, \lambda_{n-1}, 0$. Note that there is an eigenvalue at zero in addition to the ones specified by $P_1(\lambda)$. Applying (4.27) to (4.20) yields

$$\boldsymbol{x}_{k+1} = (\boldsymbol{A} - \boldsymbol{b}\boldsymbol{m}^{\mathrm{T}}\boldsymbol{A})\boldsymbol{x}_k. \tag{4.31}$$

Applying the transformation

$$\begin{bmatrix}\boldsymbol{x}_k^* \\ \sigma_k\end{bmatrix} = \begin{bmatrix}\boldsymbol{I}_{n-1} & \boldsymbol{0} \\ \multicolumn{2}{c}{\boldsymbol{m}^{\mathrm{T}}}\end{bmatrix} \boldsymbol{x}_k = \boldsymbol{T}\boldsymbol{x}_k \tag{4.32}$$

to (4.31) and considering (4.25) results in

$$\begin{bmatrix}\boldsymbol{x}_{k+1}^* \\ \sigma_{k+1}\end{bmatrix} = \begin{bmatrix}\boldsymbol{A}_1 & \boldsymbol{a}_s \\ \boldsymbol{0} & 0\end{bmatrix} \begin{bmatrix}\boldsymbol{x}_k^* \\ \sigma_k\end{bmatrix} \tag{4.33}$$

with $\boldsymbol{A}_1 \in \mathbb{R}^{n-1 \times n-1}$ and $\boldsymbol{a}_s \in \mathbb{R}^{n-1}$. The transformation matrix $\boldsymbol{T}$ is invertible if the last element of $\boldsymbol{m}^{\mathrm{T}}$ is not zero. As $\boldsymbol{m}^{\mathrm{T}}$ is a nonzero vector, this property can always be enforced by reordering the states. The transformed system matrix in (4.33) has the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_{n-1}, 0$. Due to the specific structure of the transformed system matrix in (4.33), matrix $\boldsymbol{A}_1$ has the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_{n-1}$.
For $\sigma_k = 0 \; \forall k$, the difference equation

$$\boldsymbol{x}_{k+1}^* = \boldsymbol{A}_1 \boldsymbol{x}_k^* \tag{4.34}$$

results from (4.33). Consequently, the eigenvalues of the ideal quasi-sliding mode dynamics are given by $\lambda_1, \lambda_2, \ldots, \lambda_{n-1}$. $\square$

Further insights and examples are given in [28].

4.2.2 Transformation for Sliding Surface Design

As mentioned before, the Ackermann formula is applicable only in the single-input case. Hence, an alternative approach is shown for the multi-input case. Consider the controllable discrete time multi-input system

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k \tag{4.35}$$

with $\boldsymbol{x}_k \in \mathbb{R}^n$ and $\boldsymbol{u}_k \in \mathbb{R}^m$. Matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are of appropriate dimension. Furthermore, consider the LQ-factorization of the input matrix

$$\boldsymbol{B} = \boldsymbol{Q}\boldsymbol{L} \tag{4.36}$$

with an orthogonal matrix $\boldsymbol{Q}$ (i.e. $\boldsymbol{Q}^{-1} = \boldsymbol{Q}^{\mathrm{T}}$) and a lower triangular matrix $\boldsymbol{L}$ such that

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B} = \boldsymbol{L} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{B}_1 \end{bmatrix} \tag{4.37}$$

with $\boldsymbol{B}_1 \in \mathbb{R}^{m \times m}$ (see [34]).

Theorem 4.5. Consider the controllable plant (4.35) and the sliding variable

$$\boldsymbol{\sigma}_k = \boldsymbol{M}\boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{B}_1^{-1}\tilde{\boldsymbol{M}} & \boldsymbol{B}_1^{-1} \end{bmatrix} \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{x}_k \tag{4.38}$$

with $\tilde{\boldsymbol{M}} \in \mathbb{R}^{m \times (n-m)}$. The eigenvalues of the corresponding ideal quasi-sliding mode dynamics are specified by placing the $n-m$ eigenvalues of

$$\boldsymbol{A}_{11} - \boldsymbol{A}_{12}\tilde{\boldsymbol{M}} \tag{4.39}$$

with

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix} \qquad \begin{matrix} \boldsymbol{A}_{11} \in \mathbb{R}^{(n-m)\times(n-m)} \\ \boldsymbol{A}_{22} \in \mathbb{R}^{m \times m} \end{matrix} \qquad \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{B}_1 \end{bmatrix} \tag{4.40}$$

to the desired eigenvalues. The eigenvalues of (4.39) can be placed, for instance, by applying one of the methods proposed in [35].

Proof. Transforming system (4.35) with the transformation

$$\begin{bmatrix} \boldsymbol{z}_{1,k} \\ \boldsymbol{z}_{2,k} \end{bmatrix} = \boldsymbol{Q}^{\mathrm{T}} \boldsymbol{x}_k, \tag{4.41}$$

where $\boldsymbol{z}_{1,k} \in \mathbb{R}^{n-m}$ and $\boldsymbol{z}_{2,k} = \mathbb{R}^m$, results in

$$\begin{bmatrix} \boldsymbol{z}_{1,k+1} \\ \boldsymbol{z}_{2,k+1} \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix}}_{\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}} \begin{bmatrix} \boldsymbol{z}_{1,k} \\ \boldsymbol{z}_{2,k} \end{bmatrix} + \underbrace{\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{B}_1 \end{bmatrix}}_{\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B}} \boldsymbol{u}_k. \tag{4.42}$$

Considering sliding variable (4.38) during ideal quasi-sliding mode (i.e. $\boldsymbol{\sigma}_k = \boldsymbol{0}$), the relations

$$\boldsymbol{\sigma}_k = \begin{bmatrix} \boldsymbol{B}_1^{-1}\tilde{\boldsymbol{M}} & \boldsymbol{B}_1^{-1} \end{bmatrix} \boldsymbol{Q}^{\mathrm{T}} \boldsymbol{x}_k = \boldsymbol{0} \Leftrightarrow \boldsymbol{z}_{2,k} = -\tilde{\boldsymbol{M}} \boldsymbol{z}_{1,k} \tag{4.43}$$

hold. Applying (4.43) to (4.42) results in

$$\boldsymbol{z}_{1,k+1} = (\boldsymbol{A}_{11} - \boldsymbol{A}_{12}\tilde{\boldsymbol{M}})\boldsymbol{z}_{1,k}, \tag{4.44}$$

which gives the difference equation of the remaining dynamics during ideal quasi-sliding mode. Thus, placing eigenvalues of (4.44) by means of varying $\tilde{\boldsymbol{M}}$ places the eigenvalues of the remaining dynamic matrix during ideal quasi-sliding mode. □

Remark 4.6. Note that

$$\boldsymbol{M}\boldsymbol{B} = \boldsymbol{I}_m \tag{4.45}$$

holds for $\boldsymbol{M}$ if it is designed according to theorem 4.5. This can be verified by applying (4.38) and (4.40), which results in

$$\boldsymbol{M}\boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}_1^{-1}\tilde{\boldsymbol{M}} & \boldsymbol{B}_1^{-1} \end{bmatrix} \underbrace{\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B}}_{\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{B}_1 \end{bmatrix}} = \boldsymbol{I}_m. \tag{4.46}$$

For $\boldsymbol{m}^{\mathrm{T}}$ as designed in theorem 4.4, the relation $\boldsymbol{m}^{\mathrm{T}}\boldsymbol{b} = 1$ holds too. This has already been shown in (4.26).

4.3 Matching Condition

Motions during sliding mode are only insensitive to disturbances which fulfill a certain condition, the so-called matching condition. Consider the continuous time system

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{A}_c\boldsymbol{x} + \boldsymbol{B}_c\boldsymbol{u} + \boldsymbol{D}_c\boldsymbol{g}_c \tag{4.47}$$

with state vector $\boldsymbol{x} \in \mathbb{R}^n$, vector of inputs $\boldsymbol{u} \in \mathbb{R}^m$ and vector of disturbances $\boldsymbol{g}_c \in \mathbb{R}^p$. The matrices $\boldsymbol{B}_c$ and $\boldsymbol{D}_c$ are of appropriate dimension. In [36] it was shown that the motion during sliding mode is invariant to the disturbance $\boldsymbol{g}_c$ if the condition

$$\operatorname{rank}[\boldsymbol{B}_c, \boldsymbol{D}_c] = \operatorname{rank} \boldsymbol{B}_c, \tag{4.48}$$

the so-called matching condition, is satisfied. Equivalently, the disturbance can be expressed as

$$\boldsymbol{D}_c \boldsymbol{g}_c = \boldsymbol{B}_c \tilde{\boldsymbol{f}}. \tag{4.49}$$

Hence, (4.47) can be simplified to

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{A}_c \boldsymbol{x} + \boldsymbol{B}_c(\boldsymbol{u} + \tilde{\boldsymbol{f}}). \tag{4.50}$$

In the remainder of this section, it will be examined under which conditions the matched character of perturbation $\tilde{\boldsymbol{f}}$ is maintained for a discrete time implementation of the controller, i.e. $\boldsymbol{u}(t) = \boldsymbol{u}_k$ for $kT \leq t < (k+1)T$.

Discretizing (4.47) while exploiting (4.49) with sampling time T results in

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{g}_k \tag{4.51}$$

with

$$\boldsymbol{A} = e^{\boldsymbol{A}_c T}, \qquad \boldsymbol{B} = \int_0^T e^{\boldsymbol{A}_c \tau} \boldsymbol{B}_c d\tau, \qquad \boldsymbol{g}_k = e^{\boldsymbol{A}_c T} \int_0^T e^{-\boldsymbol{A}_c s} \boldsymbol{B}_c \tilde{\boldsymbol{f}}(kT + s) ds. \tag{4.52}$$

For the discrete time system to be insensitive to the disturbance $\boldsymbol{g}_k$ during ideal quasi-sliding mode, the same condition as in the continuous time case has to be satisfied, i.e.

$$\boldsymbol{g}_k = \boldsymbol{B}\boldsymbol{f}_k. \tag{4.53}$$

From (4.52) one can see that even if the matching condition is satisfied for the continuous time system, it is not guaranteed that the matching condition is satisfied for the discretized system for an arbitrary perturbation $\tilde{\boldsymbol{f}}$. However, in [37] it was proposed that the relation

$$\boldsymbol{g}_k = \boldsymbol{B}\tilde{\boldsymbol{f}}(kT) + \frac{1}{2}\boldsymbol{B}\boldsymbol{v}(kT)T + \mathcal{O}\left(T^3\right) \tag{4.54}$$

with $\boldsymbol{v}(t) = \frac{\mathrm{d}\tilde{\boldsymbol{f}}}{\mathrm{d}t}$ holds for a bounded and smooth perturbation $\tilde{\boldsymbol{f}}$. This means that the magnitude of the unmatched part in $\boldsymbol{g}_k$ is of order $\mathcal{O}\left(T^3\right)$. Therefore, the unmatched part is negligible for suitably small chosen sampling times T. Moreover, the perturbation in the discrete time model is given by

$$\boldsymbol{g}_k = \boldsymbol{B}\boldsymbol{f}_k \tag{4.55}$$

for piecewise constant perturbations, i.e.

$$\tilde{\boldsymbol{f}}(t) = \tilde{\boldsymbol{f}}(kT) = \boldsymbol{f}_k, \quad kT \leq t < (k+1)T. \tag{4.56}$$

Hence, it does not contain unmatched components.

As previously analyzed, no ideal sliding mode is possible for discrete time systems. As a result, the design of discrete time sliding mode controllers is dedicated to ensuring a small quasi-sliding mode band and thus reducing the influence of disturbances on the motions during quasi-sliding mode.

The existing literature on the design of discrete time sliding mode controllers can be divided into two categories. One category focuses on applying the reaching law approach to design sliding mode controllers purely in discrete time to meet certain specifications. These control approaches are not necessarily inspired by continuous time sliding mode controllers. Research in the second category uses a continuous time sliding mode controller and applies discretization techniques in order to keep desirable properties of the original controller. In this dissertation, control laws derived from both categories will be introduced and adapted for the networked control setup.

4.4 The Reaching Law Approach

Consider a discrete time multi-input system

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}(\boldsymbol{u}_k + \boldsymbol{f}_k) \tag{4.57}$$

with state vector $\boldsymbol{x}_k \in \mathbb{R}^n$, vector of inputs $\boldsymbol{u}_k \in \mathbb{R}^m$ and vector of bounded matched perturbations $\boldsymbol{f}_k \in \mathbb{R}^m$ with the perturbation being bounded by

$$\underline{\boldsymbol{f}} \leq \boldsymbol{f}_k \leq \bar{\boldsymbol{f}}, \ \forall k. \tag{4.58}$$

The system matrix $\boldsymbol{A}$ and input matrix $\boldsymbol{B}$ are of appropriate dimension. The sliding variable is designed as a linear combination of the states, i.e.

$$\boldsymbol{\sigma}_k = \boldsymbol{M}\boldsymbol{x}_k \tag{4.59}$$

with

$$\boldsymbol{M}\boldsymbol{B} = \boldsymbol{I}_m. \tag{4.60}$$

Remark 4.7. Note that (4.60) can always be achieved by choosing

$$\boldsymbol{M} = (\tilde{\boldsymbol{M}}\boldsymbol{B})^{-1}\tilde{\boldsymbol{M}} \tag{4.61}$$

for a given $\tilde{\boldsymbol{M}}$. The matrix $\tilde{\boldsymbol{M}}\boldsymbol{B}$ is always invertible if the sliding variable $\tilde{\boldsymbol{\sigma}}_k = \tilde{\boldsymbol{M}}\boldsymbol{x}_k$ is of uniform relative degree one. Additionally, the dynamical properties of the motion during ideal quasi-sliding mode are not altered by applying (4.61).

Using (4.57) and (4.59), the forward increment of the sliding variable is given by

$$\boldsymbol{\sigma}_{k+1} = \boldsymbol{M}\boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{u}_k + \boldsymbol{f}_k. \tag{4.62}$$

The idea of the reaching law approach is to explicitly specify the difference equation for the sliding variable. In general, this might be a function of the previous values of the sliding variable, the current time index k and a vector of initial conditions $\boldsymbol{\nu}_0$ for dynamical reaching laws. To obtain a perturbation with symmetrical bounds, the mean value of the bounds

$$\hat{\boldsymbol{f}} = \frac{1}{2}\left(\bar{\boldsymbol{f}} + \underline{\boldsymbol{f}}\right) \tag{4.63}$$

is considered in the reaching law. Consequently, the general reaching law is given by

$$\boldsymbol{\sigma}_{k+1} = \boldsymbol{\mathscr{F}}\left(\boldsymbol{\sigma}_k, \boldsymbol{\sigma}_{k-1}, \ldots, \boldsymbol{\sigma}_0, k, \boldsymbol{\nu}_0\right) - \hat{\boldsymbol{f}} + \boldsymbol{f}_k. \tag{4.64}$$

The resulting perturbation acting on $\boldsymbol{\sigma}_{k+1}$ is then symmetrically bounded by

$$\left|\boldsymbol{f}_k - \hat{\boldsymbol{f}}\right| \leq \frac{1}{2}\left(\bar{\boldsymbol{f}} - \underline{\boldsymbol{f}}\right) = \tilde{\boldsymbol{f}}. \tag{4.65}$$

Applying the reaching law (4.64) to (4.62) and solving for $\boldsymbol{u}_k$ results in the control law

$$\boldsymbol{u}_k = -\boldsymbol{M}\boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{\mathscr{F}}\left(\boldsymbol{\sigma}_k, \boldsymbol{\sigma}_{k-1}, \ldots, \boldsymbol{\sigma}_0, k, \boldsymbol{\nu}_0\right) - \hat{\boldsymbol{f}}. \tag{4.66}$$

From (4.62) one can conclude that each channel of the sliding variable is only influenced by one single-input channel. As a result, each channel of the sliding variable can be treated independently. This simplifies the multi-input problem to m single-input problems, i.e.

$$\sigma_{i,k+1} = \mathscr{F}_i\left(\sigma_{i,k}, \sigma_{i,k-1}, \ldots, \sigma_{i,0}, k, \nu_{i,0}\right) - \hat{f}_i + f_{i,k}, \qquad i = 1, 2, \ldots, m \tag{4.67}$$

with $\boldsymbol{\sigma}_k = \begin{bmatrix}\sigma_{1,k} & \sigma_{2,k} & \cdots & \sigma_{m,k}\end{bmatrix}^{\mathrm{T}}$ and $\boldsymbol{\mathscr{F}}(\cdot) = \begin{bmatrix}\mathscr{F}_1(\cdot) & \mathscr{F}_2(\cdot) & \cdots & \mathscr{F}_m(\cdot)\end{bmatrix}^{\mathrm{T}}$.

Several reaching laws are proposed in literature. In this dissertation, different reaching laws will be presented and subsequently integrated into the networked control architecture.

4.4.1 The Switching Reaching Law

One of the first published reaching laws, the switching reaching law which is summarized as

$$\sigma_{i,k+1} = \mathscr{F}_i\left(\sigma_{i,k}\right) - \hat{f}_i + f_{i,k} = \alpha_i\sigma_{i,k} - \rho_i \operatorname{sign}\left(\sigma_{i,k}\right) - \hat{f}_i + f_{i,k}, \quad i = 1, 2, \ldots, m \tag{4.68}$$

with parameters

$$0 \leq \alpha_i < 1, \qquad\qquad \rho_i > 0, \tag{4.69}$$

was proposed in [31] and some remarks are given in [38]. This reaching law is inspired by the continuous time “constant plus proportional” control law. Representing this reaching law in vector form

$$\begin{gathered}\boldsymbol{\sigma}_{k+1} = \boldsymbol{\Gamma}\boldsymbol{\sigma}_k - \boldsymbol{S}\operatorname{sign}\left(\boldsymbol{\sigma}_k\right) - \hat{\boldsymbol{f}} + \boldsymbol{f}_k \\ \boldsymbol{\Gamma} = \begin{bmatrix}\alpha_1 & 0 & \cdots & 0 \\ 0 & \alpha_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_m\end{bmatrix}, \quad \boldsymbol{S} = \begin{bmatrix}\rho_1 & 0 & \cdots & 0 \\ 0 & \rho_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \rho_m\end{bmatrix}\end{gathered} \tag{4.70}$$

and inserting it in (4.66) results in the control law

$$\boldsymbol{u}_k = (\boldsymbol{\Gamma}\boldsymbol{M} - \boldsymbol{M}\boldsymbol{A})\boldsymbol{x}_k - \boldsymbol{S}\operatorname{sign}\left(\boldsymbol{\sigma}_k\right) - \hat{\boldsymbol{f}}. \tag{4.71}$$

According to the definition of quasi-sliding mode in [31], the sliding variable should cross and re-cross zero in every successive sampling step after the sliding variable crossed the zero line for the first time. To ensure this property, the gains have to be chosen such that

$$\rho_i > \frac{1+\alpha_i}{1-\alpha_i}\tilde{f}_i, \tag{4.72}$$

see [38]. However, definition 4.1 does not require the sliding variable to cross and re-cross the zero line in every successive sampling step during quasi-sliding mode. Therefore, the gains can be chosen as

$$\rho_i > \tilde{f}_i, \tag{4.73}$$

which is generally significantly smaller. As the quasi-sliding mode band is given by

$$\Delta_i = \rho_i + \tilde{f}_i \tag{4.74}$$

as shown in [31], it is clear that smaller values of ρ_i lead to a smaller quasi-sliding mode band Δ_i. Hence, (4.73) is used in the remainder of this dissertation.

4.4.2 The Nonswitching Reaching Law

In [39], the very general reaching law

$$\sigma_{i,k+1} = \mathscr{F}_i\left(k+1, \sigma_{i,0}\right) - \hat{f}_i + f_{i,k} \tag{4.75}$$

was proposed where $\mathscr{F}_i\left(k, \sigma_{i,0}\right)$ is an a priori constructed function which fulfills the following properties:

- If $|\sigma_{i,0}| > 2\tilde{f}_i$, then

$$\begin{aligned}
&\mathscr{F}_i\left(0, \sigma_{i,0}\right) = \sigma_{i,0} \\
&\mathscr{F}_i\left(k, \sigma_{i,0}\right)\mathscr{F}_i\left(0, \sigma_{i,0}\right) \geq 0 && \forall k \geq 0 \\
&\mathscr{F}_i\left(k, \sigma_{i,0}\right) = 0 && \forall k \geq k_i^* \\
&\left|\mathscr{F}_i\left(k+1, \sigma_{i,0}\right)\right| < \left|\mathscr{F}_i\left(k, \sigma_{i,0}\right)\right| - 2\tilde{f}_i && \forall k < k_i^*
\end{aligned} \tag{4.76}$$

- Otherwise, i.e. if $|\sigma_{i,0}| \leq 2\tilde{f}_i$, then

$$\mathscr{F}_i\left(k, \sigma_{i,0}\right) = 0 \quad \forall k \geq 0. \tag{4.77}$$

The case $|\sigma_{i,0}| > 2\tilde{f}_i$ defines a function which starts at $\sigma_{i,0}$ and converges to zero in k_i^* number of steps. The last line in (4.76) ensures that the sliding variable $\sigma_{i,k}$ converges strictly monotonicly even in the worst case scenario, i.e.

$$\left|\sigma_{i,k+1}\right| < \left|\sigma_{i,k}\right|. \tag{4.78}$$

If $|\sigma_{i,0}| \leq 2\tilde{f}_i$, no reaching phase is needed and the reaching law has to evaluate to zero.

This reaching law ensures the strict monotonic convergence from any initial condition $\sigma_{i,0}$ to the quasi-sliding mode band

$$\Delta_i = \tilde{f}_i \tag{4.79}$$

in the a priori chosen number of steps k_i^* as shown in [39].
Using this reaching law, the control law is given by

$$\boldsymbol{u}_k = -\boldsymbol{M}\boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{\mathscr{F}}\left(k+1, \boldsymbol{\sigma}_0\right) - \hat{\boldsymbol{f}}. \tag{4.80}$$

One possible choice of $\mathscr{F}_i\left(k, \sigma_{i,0}\right)$ was given in [39] as

$$\mathscr{F}_i\left(k, \sigma_{i,0}\right) = \begin{cases} \frac{k_i^*-k}{k_i^*}\sigma_{i,0} & \text{for } k < k_i^* \\ 0 & \text{else} \end{cases} \tag{4.81}$$

with

$$k_i^* < \frac{|\sigma_{i,0}|}{2\tilde{f}_i}. \tag{4.82}$$

It is easy to verify that this linearly decreasing/increasing function satisfies the required properties (4.76) and (4.77).
Comparing the quasi-sliding mode band (4.74) ensured by the switching reaching law with the quasi-sliding mode band (4.79) ensured by the nonswitching reaching law confirms that the magnitude of the quasi-sliding mode band is reduced to approximately half its value.

4.5 Discretized Super Twisting Algorithms

Apart from the reaching law approach, two discretized versions of the super twisting algorithm will be adapted to apply them in the networked control system case. These algorithms are not directly designed for a discrete time system. In fact, they are designed by discretization of continuous time algorithms. Consider a continuous time perturbed integrator

$$\begin{aligned} \frac{\mathrm{d}\sigma}{\mathrm{d}t} &= u + \varphi \\ \frac{\mathrm{d}\varphi}{\mathrm{d}t} &= v \end{aligned} \tag{4.83}$$

with $\sup(v) = L_\varphi < \infty$. Note that in (4.83) instead of the amplitude of perturbation φ, its derivative is bounded. Applying the super twisting algorithm

$$\begin{aligned} u &= -\alpha\sqrt{|\sigma|}\,\mathrm{sign}\left(\sigma\right) + \nu \\ \frac{\mathrm{d}\nu}{\mathrm{d}t} &= -\beta\,\mathrm{sign}\left(\sigma\right) \end{aligned} \tag{4.84}$$

proposed in [40] results in the closed loop system

$$\begin{aligned} \frac{\mathrm{d}\sigma}{\mathrm{d}t} &= -\alpha\sqrt{|\sigma|}\,\mathrm{sign}\left(\sigma\right) + \tilde{\nu} \\ \frac{\mathrm{d}\tilde{\nu}}{\mathrm{d}t} &= -\beta\,\mathrm{sign}\left(\sigma\right) + v \end{aligned} \tag{4.85}$$

where $\tilde{\nu} = \varphi + \nu$. The parameters α and β can be chosen by the well-established setting

$$\alpha = 1.5\sqrt{L_\varphi}, \qquad \beta = 1.1L_\varphi, \tag{4.86}$$

which was initially proposed in [41]. The stability of (4.85) using (4.86) was recently proven in [42]. As proposed in [43], these closed loop dynamics can be written in the pseudo-linear form

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\sigma \\ \tilde{\nu}\end{bmatrix} = \boldsymbol{P}(\sigma)\begin{bmatrix}\sigma \\ \tilde{\nu}\end{bmatrix} + \begin{bmatrix}0 \\ v\end{bmatrix} \tag{4.87}$$

with

$$\boldsymbol{P}(\sigma) = \begin{bmatrix}-\alpha\,|\sigma|^{-\frac{1}{2}} & 1 \\ -\beta\,|\sigma|^{-1} & 0\end{bmatrix}. \tag{4.88}$$

Its characteristic polynomial is (almost everywhere) given by

$$w(s) = s^2 + \alpha\,|\sigma|^{-\frac{1}{2}}\,s + \beta\,|\sigma|^{-1} \tag{4.89}$$

and the resulting state dependent eigenvalues are given by

$$s^{(1)}(\sigma) = p^{(1)}\,|\sigma|^{-\frac{1}{2}}, \qquad s^{(2)}(\sigma) = p^{(2)}\,|\sigma|^{-\frac{1}{2}} \tag{4.90}$$

with parameters $p^{(1)}, p^{(2)} \in \mathbb{C}$ such that

$$\alpha = -(p^{(1)} + p^{(2)}), \qquad \beta = p^{(1)}p^{(2)}. \tag{4.91}$$

Performing Euler forward discretization of (4.83) with sampling time T yields

$$\begin{aligned}\sigma_{k+1} &= \sigma_k + Tu_k + T\varphi_k \\ \varphi_{k+1} &= \varphi_k + Tv_k.\end{aligned} \tag{4.92}$$

In [44], a framework was proposed in which a control law

$$\begin{aligned}u_k &= \frac{1}{T}(\tilde{\ell}_k\sigma_k - \sigma_k) + \nu_k \\ \nu_{k+1} &= \nu_k + \ell_k\sigma_k\end{aligned} \tag{4.93}$$

with $\tilde{\ell}_k$ and ℓ_k is designed to ensure that the state dependent eigenvalues of the discrete time closed loop system match the state dependent eigenvalues $q_k^{(1)}$ and $q_k^{(2)}$ which are determined by applying discretization methods on (4.90). Combining (4.93) and (4.92) results in the discrete time closed loop system

$$\begin{bmatrix}\sigma_{k+1} \\ \tilde{\nu}_{k+1}\end{bmatrix} = \boldsymbol{P}_d\begin{bmatrix}\sigma_k \\ \tilde{\nu}_k\end{bmatrix} + \begin{bmatrix}0 \\ Tv_k\end{bmatrix} \tag{4.94}$$

with $\tilde{\nu}_k = \nu_k + \varphi_k$ and

$$\boldsymbol{P}_d = \begin{bmatrix}\tilde{\ell}_k & T \\ \ell_k & 1\end{bmatrix}. \tag{4.95}$$

Computing $\det(z\boldsymbol{I}_2 - \boldsymbol{P}_d)$ results in the characteristic polynomial of $\boldsymbol{P}_d$ as

$$w(z) = z^2 - (\tilde{\ell}_k + 1)z + \tilde{\ell}_k - T\ell_k. \tag{4.96}$$

Specifying the desired characteristic polynomial

$$(z - q_k^{(1)})(z - q_k^{(2)}) = z^2 - \left(q_k^{(1)} + q_k^{(2)}\right) z + q_k^{(1)} q_k^{(2)} \overset{!}{=} w(z) \tag{4.97}$$

gives

$$\begin{aligned} \tilde{\ell}_k &= q_k^{(1)} + q_k^{(2)} - 1 \\ \ell_k &= \frac{1}{T}\left(\tilde{\ell}_k - q_k^{(1)} q_k^{(2)}\right). \end{aligned} \tag{4.98}$$

In this dissertation, the explicit Euler discretized version of the super twisting algorithm where

$$q_k^{(1)} = 1 + Ts^{(1)}(\sigma_k), \qquad q_k^{(2)} = 1 + Ts^{(2)}(\sigma_k) \tag{4.99}$$

(see [45]) and the discrete time super twisting algorithm using the matching approach where

$$q_k^{(1)} = \begin{cases} e^{s^{(1)}(\sigma_k)T} & \sigma_k \neq 0 \\ 0 & \sigma_k = 0 \end{cases}, \qquad q_k^{(2)} = \begin{cases} e^{s^{(2)}(\sigma_k)T} & \sigma_k \neq 0 \\ 0 & \sigma_k = 0 \end{cases} \tag{4.100}$$

(see [44]) are considered. The discrete version of the super twisting using the matching approach has two main advantages. First, no discretization chattering appears using this algorithm and if $\varphi_k = 0$, the origin of the closed loop system is asymptotically stable. Second, overestimating the required controller gains has very little negative impact on the accuracy in terms of the magnitude of the sliding variable σ_k compared to the explicit Euler discretized version of the super twisting algorithm. These properties and some additional ones are extensively researched in [46].

Chapter 5

Centralized Sliding Mode Control for Buffered Networked Systems

Contents

This chapter is dedicated to combining networked control and discrete time sliding mode control by using the design of the sliding variable to specify the dynamics of the sliding motion. It is also shown that these types of sliding mode controllers are implementable in a centralized fashion only, i.e. that the control law is evaluated by one single controller node that provides the actuating signals for all m input channels. The results presented in this chapter are based on the results published in [18, 19]. The considered architecture is depicted in figure 5.1. As one single controller node computing all actuating signals is

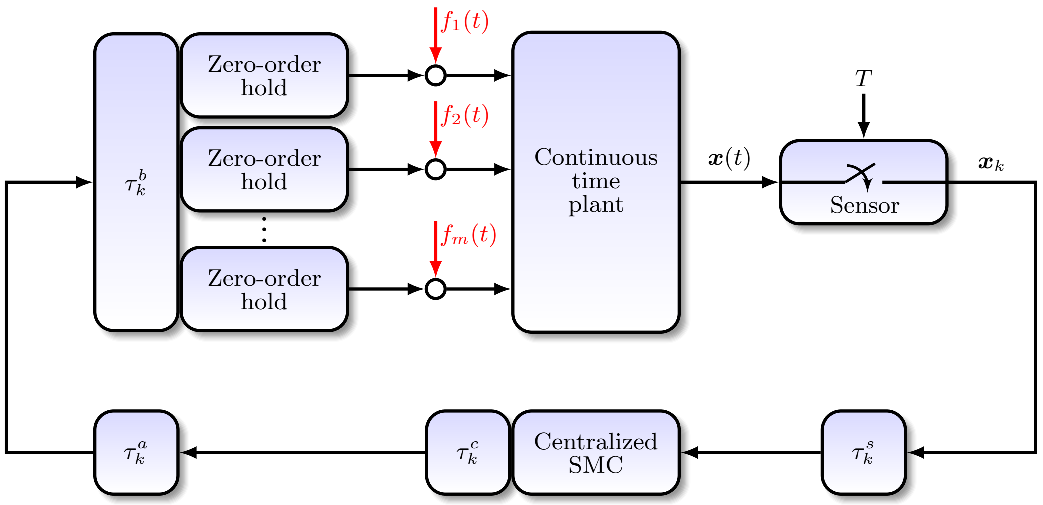


Figure 5.1: Block diagram of a centralized buffered networked control system.

implemented, only one buffer is necessary to ensure the constant round-trip time δT for all input channels simultaneously. Before applying any sliding mode control algorithms, the matching condition for this special networked control system has to be analyzed. Since the time delay is uniform over all input channels, the discrete time model of the centralized buffered network system results from (3.23) with $\delta_1 = \delta_2 = \cdots = \delta_m = \delta$ and is given by

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}(\boldsymbol{u}_{k-\delta} + \boldsymbol{f}_k) \tag{5.1}$$

with $\boldsymbol{A}$ and $\boldsymbol{B}$ given in (3.24). Using the lifted state vector

$$\boldsymbol{\xi}_k = \begin{bmatrix} \boldsymbol{x}_k^{\mathrm{T}} & u_{1,k-1} & u_{1,k-2} & \cdots & u_{1,k-\delta} & \cdots & u_{m,k-1} & \cdots & u_{m,k-\delta} \end{bmatrix}^{\mathrm{T}} \tag{5.2}$$

results in the lifted model

$$\boldsymbol{\xi}_{k+1} = \hat{\boldsymbol{A}}\boldsymbol{\xi}_k + \hat{\boldsymbol{B}}\boldsymbol{u}_k + \hat{\boldsymbol{B}}_f\boldsymbol{f}_k \tag{5.3}$$

with

$$\hat{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{0}_{n\times\delta-1} & \boldsymbol{b}_1 & \boldsymbol{0}_{n\times\delta-1} & \boldsymbol{b}_2 & \cdots & \boldsymbol{0}_{n\times\delta-1} & \boldsymbol{b}_m \\ \boldsymbol{0}_{1\times n} & \boldsymbol{0} & 0 & \boldsymbol{0} & 0 & \cdots & \boldsymbol{0} & 0 \\ \boldsymbol{0}_{\delta-1\times n} & \boldsymbol{I}_{\delta-1} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0}_{1\times n} & \boldsymbol{0} & 0 & \boldsymbol{0} & 0 & \cdots & \boldsymbol{0} & 0 \\ \boldsymbol{0}_{\delta-1\times n} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I}_{\delta-1} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \boldsymbol{0}_{1\times n} & \boldsymbol{0} & 0 & \boldsymbol{0} & 0 & \cdots & \boldsymbol{0} & 0 \\ \boldsymbol{0}_{\delta-1\times n} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{I}_{\delta-1} & \boldsymbol{0} \end{bmatrix} \tag{5.4}$$

$$\hat{\boldsymbol{B}} = \begin{bmatrix} \boldsymbol{0}_{n\times 1} & \boldsymbol{0}_{n\times 1} & \cdots & \boldsymbol{0}_{n\times 1} \\ 1 & 0 & \cdots & 0 \\ \boldsymbol{0}_{\delta-1\times 1} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ 0 & 1 & \cdots & 0 \\ \boldsymbol{0}_{\delta-1\times 1} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ \boldsymbol{0}_{\delta-1\times 1} & \boldsymbol{0} & \cdots & \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{b}}_1 & \hat{\boldsymbol{b}}_2 & \cdots & \hat{\boldsymbol{b}}_m \end{bmatrix} \tag{5.5}$$

$$\hat{\boldsymbol{B}}_f = \begin{bmatrix} \boldsymbol{B} \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{b}}_{f,1} & \hat{\boldsymbol{b}}_{f,2} & \cdots & \hat{\boldsymbol{b}}_{f,m} \end{bmatrix}. \tag{5.6}$$

Note that the lifted model for centralized buffered networked systems results directly from the lifted model (3.26) for the uniform time delay $\delta_i T = \delta T$, $\forall i$.

Comparing (5.5) and (5.6) shows that the matching condition

$$\operatorname{rank}[\hat{\boldsymbol{B}}, \hat{\boldsymbol{B}}_f] = \operatorname{rank} \hat{\boldsymbol{B}} \tag{5.7}$$

is never satisfied. For (5.1), however, it is satisfied. As a result, a sliding motion which is independent of the perturbation $\boldsymbol{f}_k$ is possible in the subspace $\boldsymbol{x}_k$ of $\boldsymbol{\xi}_k$. This is quite intuitive since, apart from $\boldsymbol{x}_k$, $\boldsymbol{\xi}_k$ also contains the delayed control signals, which in the ideal case compensates for future values of the perturbation. Thus, a motion of $\boldsymbol{\xi}_k$ which is independent of the perturbation $\boldsymbol{f}_k$ is neither possible nor desired.

5.1 Design of the Sliding Variable for Buffered Networked Systems

As mentioned before, the first step in a sliding mode based control design very often is the design of the sliding variable. The first intention could be to design a sliding variable, based on the methods given in section 4.2, as a linear combination of the lifted state vector, i.e.

$$\boldsymbol{\sigma}_k = \boldsymbol{M}\boldsymbol{\xi}_k. \tag{5.8}$$

As the matching condition is not satisfied and this definition specifies a sliding motion in the lifted states $\boldsymbol{\xi}_k$, this contradicts the previous consideration that a sliding motion is only possible in the subspace $\boldsymbol{x}_k$.
This can be mathematically verified by considering the forward increment of (5.8) and applying the lifted model (5.3), which results in

$$\boldsymbol{\sigma}_{k+1} = \boldsymbol{M}\hat{\boldsymbol{A}}\boldsymbol{\xi}_k + \boldsymbol{M}\hat{\boldsymbol{B}}\boldsymbol{u}_k + \boldsymbol{M}\hat{\boldsymbol{B}}_f\boldsymbol{f}_k. \tag{5.9}$$

Setting $\boldsymbol{\sigma}_{k+1} = \boldsymbol{0}$, using $\boldsymbol{M}\hat{\boldsymbol{B}} = \boldsymbol{I}_m$ and solving for $\boldsymbol{u}_k$ gives the so-called equivalent control

$$\boldsymbol{u}_k = -\boldsymbol{M}\hat{\boldsymbol{A}}\boldsymbol{\xi}_k - \boldsymbol{M}\hat{\boldsymbol{B}}_f\boldsymbol{f}_k \tag{5.10}$$

which enforces the sliding motion for the next sampling step. Using this control signal in the lifted model (5.3) results in a motion during sliding given by

$$\boldsymbol{\xi}_{k+1} = (\hat{\boldsymbol{A}} - \hat{\boldsymbol{B}}\boldsymbol{M}\hat{\boldsymbol{A}})\boldsymbol{\xi}_k - (\hat{\boldsymbol{B}}\boldsymbol{M}\hat{\boldsymbol{B}}_f - \hat{\boldsymbol{B}}_f)\boldsymbol{f}_k. \tag{5.11}$$

In order for this motion to be independent of the disturbance $\boldsymbol{f}_k$, the relation

$$\hat{\boldsymbol{B}}\boldsymbol{M}\hat{\boldsymbol{B}}_f - \hat{\boldsymbol{B}}_f = \boldsymbol{0} \tag{5.12}$$

has to be satisfied. The matrices $\hat{\boldsymbol{B}}$ and $\hat{\boldsymbol{B}}_f$ given in (5.5) and (5.6) can be written as

$$\hat{\boldsymbol{B}} = \begin{bmatrix} \boldsymbol{0}_{n\times m} \\ \boldsymbol{B}_2 \end{bmatrix} \qquad \hat{\boldsymbol{B}}_f = \begin{bmatrix} \boldsymbol{B} \\ \boldsymbol{0} \end{bmatrix} \tag{5.13}$$

with

$$\boldsymbol{B}_2 = \begin{bmatrix} 1 & \cdots & 0 \\ \boldsymbol{0}_{\delta-1\times 1} & \cdots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \\ \boldsymbol{0}_{\delta-1\times 1} & \cdots & \boldsymbol{0} \end{bmatrix}. \tag{5.14}$$

Using $\boldsymbol{M} = \begin{bmatrix} \boldsymbol{M}_1 & \boldsymbol{M}_2 \end{bmatrix}$ with $\boldsymbol{M}_1 \in \mathbb{R}^{m\times n}$ and $\boldsymbol{M}_2 \in \mathbb{R}^{m\times \delta m}$ together with (5.13) in (5.12) yields

$$\hat{\boldsymbol{B}}\boldsymbol{M}\hat{\boldsymbol{B}}_f - \hat{\boldsymbol{B}}_f = \begin{bmatrix} -\boldsymbol{B} \\ \boldsymbol{B}_2\boldsymbol{M}_1\boldsymbol{B} \end{bmatrix}. \tag{5.15}$$

From (5.15) it is clear that (5.12) can never be satisfied. Hence, the resulting movement for $\boldsymbol{\sigma}_k = \boldsymbol{0}$ is not independent of the perturbation $\boldsymbol{f}_k$. This result is illustrated by means of a numerical simulation in example 5.1.

Example 5.1. Consider the continuous time plant

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \begin{bmatrix} 0 & 1 \\ 3 & -2 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} (u^* + f) \tag{5.16}$$

with state vector $\boldsymbol{x} \in \mathbb{R}^2$, scalar input u^* and perturbation f. Constructing the lifted model for sampling time $T = 0.1$s and constant round trip time $\tau_k = T$ ensured by the buffer results in

$$\boldsymbol{\xi}_{k+1} = \begin{bmatrix} 1.014 & 0.091 & 0.005 \\ 0.273 & 0.832 & 0.091 \\ 0 & 0 & 0 \end{bmatrix} \boldsymbol{\xi}_k + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u_k + \begin{bmatrix} 0.005 \\ 0.091 \\ 0 \end{bmatrix} f_k. \tag{5.17}$$

Using one of the methods in section 4.2 and specifying the two eigenvalues to $\lambda_1 = \lambda_2 = e^{-T}$ to design the sliding variable results in

$$\sigma_k = \begin{bmatrix} 3.997 & 0.193 & 1 \end{bmatrix} \boldsymbol{\xi}_k. \tag{5.18}$$

Applying the control law (5.10), which equals

$$u_k = \begin{bmatrix} -4.106 & -0.524 & -0.036 \end{bmatrix} \boldsymbol{\xi}_k + (-0.036) f_k, \tag{5.19}$$

ensures that $\sigma_k = 0$ for $k > 0$. Figure 5.2 shows simulated trajectories which start from $\boldsymbol{\xi}_0 = \begin{bmatrix} 1 & 2 & 0 \end{bmatrix}^{\mathrm{T}}$ and are affected by different disturbances f_k. As this figure shows, all trajectories immediately approach the $\sigma_k = 0$ plane after the first step and remain on this plane. If the system is in sliding mode, the trajectories would, independently of the perturbation, slide along the $\sigma_k = 0$ plane towards the origin. In the shown picture, the trajectories move along this plane but the movements on this plane are affected by the perturbation and therefore the convergence to the origin is not ensured. Consequently, no sliding motion occurs although the sliding variable σ_k is zero.

Theorem 5.2. Consider the lifted model of a centralized buffered networked system (5.3)–(5.6). Using a sliding variable

$$\boldsymbol{\sigma}_k = \hat{\boldsymbol{M}} \boldsymbol{\xi}_k = \begin{bmatrix} \boldsymbol{M} & \boldsymbol{0} \end{bmatrix} \boldsymbol{\xi}_k = \underbrace{\begin{bmatrix} \boldsymbol{m}_1^{\mathrm{T}} \\ \boldsymbol{m}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{m}_m^{\mathrm{T}} \end{bmatrix}}_{\boldsymbol{M}} \boldsymbol{x}_k, \tag{5.20}$$

which is a linear combination of the plant states, only ensures a perturbation independent motion for $\boldsymbol{\sigma}_k = \boldsymbol{0}$.

Proof. To verify that $\boldsymbol{\sigma}_k = \boldsymbol{0}$ ensures a disturbance insensitive motion, consider the forward increment of (5.20) together with (5.1), which results in

$$\boldsymbol{\sigma}_{k+1} = \boldsymbol{M}\boldsymbol{A}\boldsymbol{x}_k + \underbrace{\boldsymbol{M}\boldsymbol{B}}_{\boldsymbol{I}_m}(\boldsymbol{u}_{k-\delta} + \boldsymbol{f}_k). \tag{5.21}$$

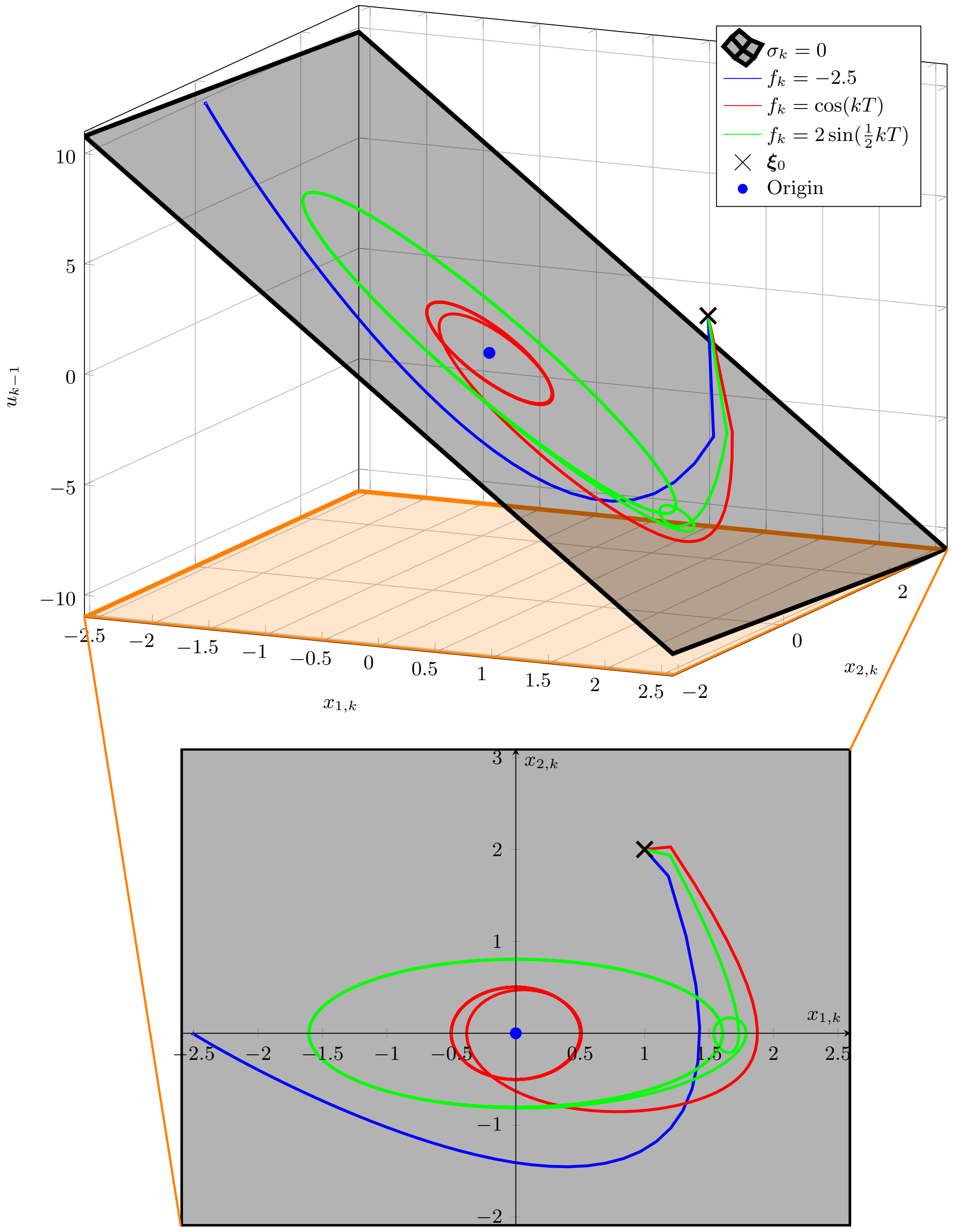


Figure 5.2: Example 5.1: Trajectories of system (5.17) with controller (5.19) affected by different perturbations f_k. In the lower section, the projection on the $x_{1,k}/x_{2,k}$ plane is illustrated.

Setting $\boldsymbol{\sigma}_{k+1} = \mathbf{0}$ and solving for the control signals results in

$$\boldsymbol{u}_{k-\delta} = -\boldsymbol{M}\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{f}_k \tag{5.22}$$

which enforces $\boldsymbol{\sigma}_k = \mathbf{0}$ for $k > 0$. Using this control signal in (5.1) yields

$$\boldsymbol{x}_{k+1} = (\boldsymbol{A} - \boldsymbol{B}\boldsymbol{M}\boldsymbol{A})\boldsymbol{x}_k + \boldsymbol{B}(\boldsymbol{f}_k - \boldsymbol{f}_k) \tag{5.23}$$

which obviously is independent of the perturbation $\boldsymbol{f}_k$. □

As the control signal (5.22) is not causal in the present form, it cannot be implemented. In order to circumvent this issue, consider the forward increments of the plant states

$$\begin{aligned}
\boldsymbol{x}_{k+2} &= \boldsymbol{A}^2\boldsymbol{x}_k + \boldsymbol{A}\boldsymbol{B}(\boldsymbol{u}_{k-\delta} + \boldsymbol{f}_k) + \boldsymbol{B}(\boldsymbol{u}_{k-\delta+1} + \boldsymbol{f}_{k+1}) && (5.24)\\
\boldsymbol{x}_{k+3} &= \boldsymbol{A}^3\boldsymbol{x}_k + \boldsymbol{A}^2\boldsymbol{B}(\boldsymbol{u}_{k-\delta} + \boldsymbol{f}_k) + \boldsymbol{A}\boldsymbol{B}(\boldsymbol{u}_{k-\delta+1} + \boldsymbol{f}_{k+1}) + \boldsymbol{B}(\boldsymbol{u}_{k-\delta+2} + \boldsymbol{f}_{k+2}) && (5.25)\\
&\vdots\\
\boldsymbol{x}_{k+\delta} &= \boldsymbol{A}^\delta\boldsymbol{x}_k + \sum_{i=0}^{\delta-1}\boldsymbol{A}^i\boldsymbol{B}(\boldsymbol{u}_{k-i-1} + \boldsymbol{f}_{k+\delta-1-i}). && (5.26)
\end{aligned}$$

Using (5.26), the control signal in (5.22) can be rewritten in the form

$$\boldsymbol{u}_k = -\boldsymbol{M}\boldsymbol{A}\underbrace{\left(\boldsymbol{A}^\delta\boldsymbol{x}_k + \sum_{i=0}^{\delta-1}\boldsymbol{A}^i\boldsymbol{B}(\boldsymbol{u}_{k-i-1} + \boldsymbol{f}_{k+\delta-1-i})\right)}_{\boldsymbol{x}_{k+\delta}} - \boldsymbol{f}_{k+\delta}, \tag{5.27}$$

which can be implemented if the perturbation $\boldsymbol{f}_k$ and its future values are known.

Example 5.3. Reconsider example 5.1 which results in the lifted model (5.17). Designing the sliding variable (5.20) according to section 4.2 and specifying one eigenvalue to $\lambda_1 = e^{-T}$ results in

$$\sigma_k = \begin{bmatrix}10.473 & 10.439 & 0\end{bmatrix}\boldsymbol{\xi}_k. \tag{5.28}$$

According to (5.27), the control signal is given by

$$u_k = \begin{bmatrix}-16.297 & -9.245 & -0.941\end{bmatrix}\boldsymbol{\xi}_k - 0.941 f_k - f_{k+1}. \tag{5.29}$$

Simulated trajectories of system (5.17), which is affected by different perturbations f_k, start from $\boldsymbol{\xi}_0 = \begin{bmatrix}1 & 2 & 0\end{bmatrix}^{\mathrm{T}}$ and are shown in figure 5.3. These trajectories show that at least two steps are required for the states to reach the sliding surface, which is intuitively clear as the sliding variable depends on the plant states only. Therefore, the first control signal u_0 reaches the plant in the second iteration, which then leads to $\sigma_k = 0$ for $k \geq 2$. These trajectories also show that the motion in the lifted states $\boldsymbol{\xi}_k$ is affected by the perturbation since u_{k-1} has to compensate for the perturbation f_k. However, the motion of the plant states $\boldsymbol{x}_k$ is insensitive to the perturbation once the sliding variable has approached zero. This is illustrated by the projection of the trajectories to the $x_{1,k}/x_{2,k}$ plane, which is also depicted in figure 5.3.

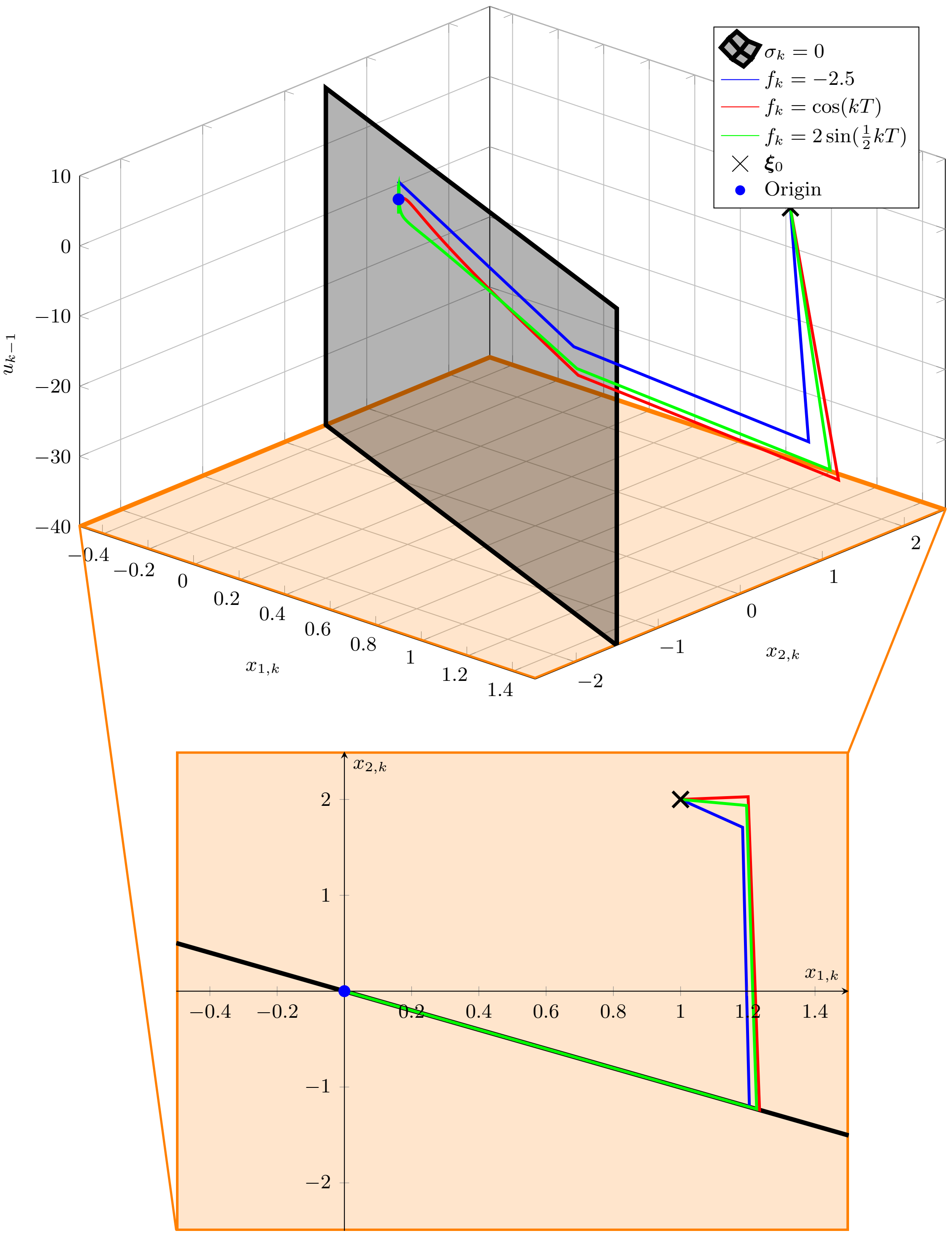


Figure 5.3: Example 5.3: Trajectories of system (5.17) with controller (5.29) affected by different perturbations f_k. In the lower section, the projection on the $x_{1,k}/x_{2,k}$ plane is illustrated.

Remark 5.4. Note that (5.27) is always evaluable in the centralized setup as all control signals are delayed by the same time delay δ. Why this approach is not suitable for a spatially distributed setup, in which the control law is distributed over several controller nodes, is investigated in this remark.
Applying (3.23) and exploiting $\boldsymbol{MB} = \boldsymbol{I}_m$ to compute the forward increments of the ith component of (5.20) results in

$$\begin{aligned}
\sigma_{i,k+1} &= \boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x}_k + \underbrace{\sum_{j=1}^{m} \boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{b}_j (u_{j,k-\delta_j} + f_{j,k})}_{u_{i,k-\delta_i}+f_{i,k}} = \boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x}_k + u_{i,k-\delta_i} + f_{i,k} \\
\sigma_{i,k+2} &= \boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A}^2 \boldsymbol{x}_k + \sum_{j=1}^{m} \boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A} \boldsymbol{b}_j (u_{j,k-\delta_j} + f_{j,k}) + u_{i,k-\delta_i+1} + f_{i,k+1} \\
&\vdots \\
\sigma_{i,k+\delta_i+1} &= \boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A}^{\delta_i+1} \boldsymbol{x}_k + \sum_{r=0}^{\delta_i-1} \sum_{j=1}^{m} \boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A}^{+\delta_i-r} \boldsymbol{b}_j (u_{j,k-\delta_j+r} + f_{j,k+r}) + u_{i,k} + f_{i,k+\delta_i}.
\end{aligned} \tag{5.30}$$

To force the ith component of the sliding variable to zero using the ith input, $\sigma_{i,k+\delta_i+1} = 0$ has to be solved for $u_{i,k}$, which results in

$$u_{i,k} = -\boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A}^{\delta_i+1} \boldsymbol{x}_k - \sum_{r=0}^{\delta_i-1} \sum_{j=1}^{m} \boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A}^{\delta_i-r} \boldsymbol{b}_j (u_{j,k-\delta_j+r} + f_{j,k+r}) - f_{i,k+\delta_i}. \tag{5.31}$$

From (5.31) one can conclude that the most recent control signals necessary to compute $u_{i,k}$ are $u_{j,k-\delta_j+\delta_i-1}$ as, in general, $\boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A} \boldsymbol{b}_j \neq 0$. As a result, $\delta_i = \delta_j$, $\forall i, j$ has to be satisfied. The ith control signal (5.31) would otherwise depend on the current or even future values of the control signal of other channels. This property could also be satisfied in a spatially distributed setup by choosing the same round-trip time for all buffers in a worst case sense. However, in the next section it will be shown that control laws based on sliding variables as in (5.20) are not suitable for a spatially distributed setup.

5.2 Reaching Law Based Networked Sliding Mode Control

As discussed in section 5.1, the sliding variable has to have the form (5.20), which means that the sliding variable is a linear combination of the plant states $\boldsymbol{x}_k$ only, as opposed to the full lifted states $\boldsymbol{\xi}_k$. In order to apply a reaching law, the forward increments of the sliding variable up to its relative degree are required. These elements are calculated using (5.20) and the lifted model of the buffered networked system (5.3), which results in

$$\boldsymbol{\sigma}_{k+1} = \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}\boldsymbol{\xi}_k + \underbrace{\hat{\boldsymbol{M}}\hat{\boldsymbol{B}}}_{=\boldsymbol{0}}\boldsymbol{u}_k + \hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f\boldsymbol{f}_k \tag{5.32}$$

$$\boldsymbol{\sigma}_{k+2} = \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^2\boldsymbol{\xi}_k + \underbrace{\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}\hat{\boldsymbol{B}}}_{=\boldsymbol{0}}\boldsymbol{u}_k + \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}\hat{\boldsymbol{B}}_f\boldsymbol{f}_k + \hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f\boldsymbol{f}_{k+1} \tag{5.33}$$

$$\vdots$$

$$\boldsymbol{\sigma}_{k+\delta} = \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^\delta\boldsymbol{\xi}_k + \underbrace{\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-1}\hat{\boldsymbol{B}}}_{=\boldsymbol{0}}\boldsymbol{u}_k + \sum_{i=0}^{\delta-1}\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-i-1}\hat{\boldsymbol{B}}_f\boldsymbol{f}_{k+i} \tag{5.34}$$

$$\boldsymbol{\sigma}_{k+\delta+1} = \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta+1}\boldsymbol{\xi}_k + \underbrace{\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta}\hat{\boldsymbol{B}}}_{=\boldsymbol{I}_m}\boldsymbol{u}_k + \boldsymbol{p}_k \tag{5.35}$$

with

$$\boldsymbol{p}_k = \begin{bmatrix} p_{1,k} & p_{2,k} & \cdots & p_{m,k}\end{bmatrix}^{\mathrm{T}} = \sum_{i=0}^{\delta}\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-i}\hat{\boldsymbol{B}}_f\boldsymbol{f}_{k+i}. \tag{5.36}$$

Since $\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^i\hat{\boldsymbol{B}} = \boldsymbol{0}$ for $i = 0, 1, \ldots, \delta - 1$ in (5.32)–(5.35), the vector relative degree of the sliding variable is $\delta + 1$. The perturbation which acts on $\boldsymbol{\sigma}_{k+\delta+1}$ is denoted as $\boldsymbol{p}_k$. For the sliding mode control design used in this chapter, the lower and upper bounds

$$\underline{\boldsymbol{f}} \leq \boldsymbol{f} \leq \bar{\boldsymbol{f}} \tag{5.37}$$

of the perturbation $\boldsymbol{f}_k$ are considered to be known. Using the known mean

$$\hat{\boldsymbol{f}} = \frac{1}{2}\left(\bar{\boldsymbol{f}} + \underline{\boldsymbol{f}}\right) \tag{5.38}$$

and amplitude

$$\tilde{\boldsymbol{f}} = \frac{1}{2}\left(\bar{\boldsymbol{f}} - \underline{\boldsymbol{f}}\right) \tag{5.39}$$

of perturbation $\boldsymbol{f}_k$, the bounds of perturbation $\boldsymbol{p}_k$ are given by

$$\underline{\boldsymbol{p}} \leq \boldsymbol{p}_k \leq \bar{\boldsymbol{p}} \tag{5.40}$$

with

$$\underline{\boldsymbol{p}} = \hat{\boldsymbol{p}} - \tilde{\boldsymbol{p}} \qquad \text{and} \qquad \bar{\boldsymbol{p}} = \hat{\boldsymbol{p}} + \tilde{\boldsymbol{p}}, \tag{5.41}$$

where the mean $\hat{\boldsymbol{p}}$ is given by

$$\hat{\boldsymbol{p}} = \begin{bmatrix} \hat{p}_1 & \hat{p}_2 & \cdots & \hat{p}_m\end{bmatrix}^{\mathrm{T}} = \sum_{i=0}^{\delta}\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-i}\hat{\boldsymbol{B}}_f\hat{\boldsymbol{f}} \tag{5.42}$$

and the amplitude $\tilde{\boldsymbol{p}}$ is given by

$$\tilde{\boldsymbol{p}} = \begin{bmatrix} \tilde{p}_1 & \tilde{p}_2 & \cdots & \tilde{p}_m \end{bmatrix}^{\mathrm{T}} = \sum_{i=0}^{\delta} \left| \hat{\boldsymbol{M}} \hat{\boldsymbol{A}}^{\delta-i} \hat{\boldsymbol{B}}_f \right| \tilde{\boldsymbol{f}}. \tag{5.43}$$

The absolute operator applied to a matrix is defined as the absolute operator applied to each element of the matrix as defined in the chapter Notation at the beginning of this dissertation. In order to use the reaching law approach for deriving a control law, a reaching law for relative degree $\delta + 1$ is needed.

Remark 5.5. In [47] as well as in [48], the switching and the nonswitching reaching laws are adapted for systems with higher relative degree. These methods require the knowledge of $\boldsymbol{\sigma}_{k+1}, \boldsymbol{\sigma}_{k+2}, \ldots, \boldsymbol{\sigma}_{k+\delta}$ which can be computed for discrete time systems in which the matching condition is satisfied as they depend solely on the current state and not on the perturbation. This is because the sliding variable has the same relative degree with respect to the input and with respect to the perturbation if the matching condition is satisfied, i.e. $\hat{\boldsymbol{B}}_f = \hat{\boldsymbol{B}}$. For centralized buffered networked systems, the relative degree of the sliding variable with respect to the input is δ and with respect to the perturbation is one. Hence, future values of the sliding variable depend on the unknown perturbation, see (5.32)–(5.35). Consequently, these approaches are not applicable to buffered networked control systems.

As discussed in remark 5.5, the applied reaching law must not depend on future samples of the sliding variable $\boldsymbol{\sigma}_k$. The resulting control law otherwise depends on $\boldsymbol{f}_{k+i}$. Therefore, the general reaching law for relative degree $\delta + 1$ can be formulated as

$$\boldsymbol{\sigma}_{k+\delta+1} = \mathscr{F}\left(\boldsymbol{\sigma}_k, \boldsymbol{\sigma}_{k-1}, \ldots, \boldsymbol{\sigma}_0, k, \boldsymbol{\nu}_0\right) - \hat{\boldsymbol{p}} + \boldsymbol{p}_k. \tag{5.44}$$

Applying this general reaching law to (5.35) and solving for $\boldsymbol{u}_k$ results in the control law

$$\boldsymbol{u}_k = -\hat{\boldsymbol{M}} \hat{\boldsymbol{A}}^{\delta+1} \boldsymbol{\xi}_k + \mathscr{F}\left(\boldsymbol{\sigma}_k, \boldsymbol{\sigma}_{k-1}, \ldots, \boldsymbol{\sigma}_0, k, \boldsymbol{\nu}_0\right) - \hat{\boldsymbol{p}}. \tag{5.45}$$

Note that the reaching law has to be designed in such a way that it deals with the perturbation $\boldsymbol{p}_k - \hat{\boldsymbol{p}}$ with magnitude $\tilde{\boldsymbol{p}}$.

Remark 5.6. Sliding mode controllers based on sliding variables as in (5.20) are not suitable for a spatially distributed setup even though a uniform time delay can be achieved by the buffers. Consider the ith component of (5.45) using (3.27) and (5.20), which results in

$$u_{i,k} = -\boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A}^{\delta+1} \boldsymbol{x}_k + \sum_{j=0}^{\delta} \boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A}^{\delta-j} \boldsymbol{B} \boldsymbol{u}_{k-\delta+j} + \mathscr{F}_i\left(\sigma_{i,k}, \sigma_{i,k-1}, \ldots, \sigma_{i,0}, k, \nu_{i,0}\right) - \hat{p}_i. \tag{5.46}$$

For a control law to be implemented in a spatially distributed setup, the control law for the ith input is required to solely rely on the measured state vector $\boldsymbol{x}_k$ and the history of the ith control samples $\begin{bmatrix} u_{i,k-1} & u_{i,k-2} & \cdots & u_{i,0} \end{bmatrix}^{\mathrm{T}}$ and not on control samples of other input channels. Otherwise, the ith controller node has to receive the results from all other controller nodes from the previous step. This communication link would also be affected by network-induced delays. For delays larger than one sampling step, this data cannot be provided in time. To ensure that each controller node uses only local available information, the ith line of $\boldsymbol{M}$ has to fulfill

$$\boldsymbol{m}_i^{\mathrm{T}} \boldsymbol{A}^l \boldsymbol{b}_j = 0 \qquad \text{with} \qquad \begin{array}{l} j \neq i \\ l = 0, 1, \ldots, \delta. \end{array} \tag{5.47}$$

It is clear that (5.47) cannot be satisfied in general, especially for $n > \delta$. In addition, asymptotic stability of the ideal quasi-sliding motion has to be ensured. Since all these requirements are too restrictive to be of any practical relevance, the controllers developed in this chapter are designed for the centralized setup only.

5.2.1 The Switching Reaching Law

Applying the switching reaching law (4.70) to $\boldsymbol{\sigma}_{k+\delta+1}$ as stated in (4.70) results in

$$\begin{aligned} &\boldsymbol{\sigma}_{k+\delta+1} = \boldsymbol{\Gamma}\boldsymbol{\sigma}_k - \boldsymbol{S}\,\mathrm{sign}\left(\boldsymbol{\sigma}_k\right) - \hat{\boldsymbol{p}} + \boldsymbol{p}_k \\ &\boldsymbol{\Gamma} = \begin{bmatrix} \alpha_1 & 0 & \cdots & 0 \\ 0 & \alpha_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_m \end{bmatrix}, \quad \boldsymbol{S} = \begin{bmatrix} \rho_1 & 0 & \cdots & 0 \\ 0 & \rho_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \rho_m \end{bmatrix}, \quad \begin{array}{l} 0 \leq \alpha_i < 1 \\ \rho_i > \tilde{p}_i \end{array}. \end{aligned} \tag{5.48}$$

This reaching law can be analyzed by defining a down-sampled version of the sliding variable

$$\boldsymbol{s}_l = \boldsymbol{\sigma}_{(\delta+1)l+o} \tag{5.49}$$

with $l \in \mathbb{Z}$ and an offset $o \in [0, 1, \ldots, \delta]$. Using (5.49) and (5.48) to compute $\boldsymbol{s}_{l+1}$ with $k = (\delta+1)l + o$ results in

$$\boldsymbol{s}_{l+1} = \boldsymbol{\sigma}_{(\delta+1)l+o+\delta+1} = \boldsymbol{\Gamma}\boldsymbol{\sigma}_{(\delta+1)l+o} - \boldsymbol{S}\,\mathrm{sign}\left(\boldsymbol{\sigma}_{(\delta+1)l+o}\right) - \hat{\boldsymbol{p}} + \boldsymbol{p}_{(\delta+1)l+o} \tag{5.50}$$

$$= \boldsymbol{\Gamma}\boldsymbol{s}_l - \boldsymbol{S}\,\mathrm{sign}\left(\boldsymbol{s}_l\right) - \hat{\boldsymbol{p}} + \boldsymbol{p}_{(\delta+1)l+o}. \tag{5.51}$$

From (5.51), one can conclude that using (5.48) has the same effect as independently applying the reaching law in (4.70) to $\delta + 1$ consecutive elements of $\boldsymbol{\sigma}_k$ as (5.51) holds independently of the offset o. Using reaching law (5.48) in (5.45) results in the control law

$$\boldsymbol{u}_k = \left(\boldsymbol{\Gamma}\hat{\boldsymbol{M}} - \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta+1}\right)\boldsymbol{\xi}_k - \boldsymbol{S}\,\mathrm{sign}\left(\hat{\boldsymbol{M}}\boldsymbol{\xi}_k\right) - \hat{\boldsymbol{p}}. \tag{5.52}$$

Applying (5.52) to the lifted model of the buffered networked system (5.3) in the perturbation-free case, i.e. $\boldsymbol{f}_k = \boldsymbol{0}$ and $\boldsymbol{S} = \boldsymbol{0}$, yields the nominal closed loop system

$$\boldsymbol{\xi}_{k+1} = \left[\hat{\boldsymbol{A}} - \hat{\boldsymbol{B}}(\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta+1} - \boldsymbol{\Gamma}\hat{\boldsymbol{M}})\right]\boldsymbol{\xi}_k. \tag{5.53}$$

Theorem 5.7. Consider the centralized buffered networked control system depicted in figure 5.1 with plant (1.5) and sampling time T. Also, assume that assumptions 1.2, 1.4, 2.4 and 2.6 are satisfied and perturbation $\boldsymbol{f}_k$ is bounded by the known bounds

$\underline{\boldsymbol{f}} \leq \boldsymbol{f}_k \leq \bar{\boldsymbol{f}}$ $\forall k$. Let the controller be given by (5.52) with

$$\begin{matrix} \rho_i > \tilde{p}_i \\ 0 \leq \alpha_i < 1 \end{matrix}, \qquad i = 1, 2, \dots, m \tag{5.54}$$

and $\hat{\boldsymbol{M}}$ in (5.20) designed such that the ideal quasi-sliding mode is asymptotically stable. The states $\boldsymbol{x}$ of (1.5) are then ultimately bounded as defined in [29]. In addition, the quasi-sliding mode band of the i$^{\text{th}}$ sliding variable is given by

$$\Delta_i = 2(\rho_i + \tilde{p}_i). \tag{5.55}$$

Moreover, $m(\delta+1)$ eigenvalues of the nominal closed loop system (5.53) are given by

$$z_{jm+i} = \sqrt[\delta+1]{\alpha_i} e^{\frac{2j\pi}{\delta+1}}, \qquad \begin{matrix} i = 1, 2, \dots, m \\ j = 0, 1, \dots, \delta \end{matrix} \tag{5.56}$$

and the remaining $n - m$ eigenvalues are specified during the design process of $\hat{\boldsymbol{M}}$.

Proof. Applying control law (5.52) ensures that the sliding variable follows the reaching law (5.48). Large enough values have to be chosen for the parameters ρ_i to dominate the amplitude $\tilde{p}_i$ of perturbation $p_{i,k}$. This is ensured by the first relation in (5.54). To calculate the width of the i$^{\text{th}}$ quasi-sliding mode band, set $\sigma_{i,k} = -\varepsilon$ with $\varepsilon > 0$ and compute $\sigma_{i,k+\delta+1}$ using reaching law (5.48), which results in

$$\sigma_{i,k+\delta+1} = -\alpha_i\varepsilon + \rho_i + p_{i,k} - \hat{p}_i. \tag{5.57}$$

The upper bound $\bar{\sigma}_i$ of (5.57) is then given by

$$\sigma_{i,k+\delta+1} \leq \rho_i + \tilde{p}_i = \bar{\sigma}_i. \tag{5.58}$$

As in (5.58), the lower bound is calculated by setting $\sigma_{i,k} = \varepsilon$ with $\varepsilon > 0$ and finding the lower bound of $\sigma_{i,k+\delta+1}$, which results in

$$\sigma_{i,k+\delta+1} \geq -\rho_i - \tilde{p}_i = \underline{\sigma}_i. \tag{5.59}$$

As a consequence, the width of the quasi-sliding mode band is given by (5.55). In (5.51), it was shown that the reaching law is independently applied to δ consecutive samples. Hence, Δ_i is the overall quasi-sliding mode band for the i$^{\text{th}}$ sliding variable.
To analyze the eigenvalues of the nominal closed loop system matrix, consider the state transformation using (5.34)

$$\begin{bmatrix} \boldsymbol{x}_{1,k} \\ \boldsymbol{\sigma}_k \\ \boldsymbol{\sigma}_{k+1} \\ \vdots \\ \boldsymbol{\sigma}_{k+\delta} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \mathcal{N}(\boldsymbol{M})^{\mathrm{T}} & \boldsymbol{0} \end{bmatrix} \\ \hat{\boldsymbol{M}} \\ \hat{\boldsymbol{M}}\hat{\boldsymbol{A}} \\ \vdots \\ \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta} \end{bmatrix} \boldsymbol{\xi}_k \tag{5.60}$$

where the columns of $\mathcal{N}(\boldsymbol{M})$ span the nullspace of $\boldsymbol{M}$. Since the reaching law (5.48) in the nominal case is given by

$$\boldsymbol{\sigma}_{k+\delta+1} = \boldsymbol{\Gamma}\boldsymbol{\sigma}_k, \tag{5.61}$$

applying transformation (5.60) to the nominal closed loop system (5.53) results in

$$\begin{bmatrix} \boldsymbol{x}_{1,k+1} \\ \boldsymbol{\sigma}_{k+1} \\ \boldsymbol{\sigma}_{k+2} \\ \vdots \\ \boldsymbol{\sigma}_{k+\delta+1} \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{A}}_{11} & \tilde{\boldsymbol{A}}_{12} \\ \begin{matrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{matrix} & \underbrace{\boxed{\begin{matrix} \mathbf{0} & \boldsymbol{I}_m & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{I}_m & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{\Gamma} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \end{matrix}}}_{\tilde{\boldsymbol{A}}_{22}} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{1,k} \\ \boldsymbol{\sigma}_k \\ \boldsymbol{\sigma}_{k+1} \\ \vdots \\ \boldsymbol{\sigma}_{k+\delta} \end{bmatrix}. \tag{5.62}$$

The transformed dynamics matrix in (5.62) is in block triangular form, therefore its eigenvalues are given by the eigenvalues of $\tilde{\boldsymbol{A}}_{11}$ together with the eigenvalues of $\tilde{\boldsymbol{A}}_{22}$. The $n-m$ eigenvalues of $\tilde{\boldsymbol{A}}_{11}$ are specified in the design process of $\boldsymbol{M}$ (see section 4.2). The remaining $m(\delta+1)$ eigenvalues of (5.62) are the eigenvalues of $\tilde{\boldsymbol{A}}_{22}$.
As the ith component of (5.48) in the nominal case equals

$$\sigma_{i,k+\delta+1} = \alpha_i \sigma_{i,k}, \tag{5.63}$$

the subsystem with system matrix $\tilde{\boldsymbol{A}}_{22}$ in (5.62) can be formulated using the state vector

$$\tilde{\boldsymbol{\xi}}_k = \begin{bmatrix} \sigma_{1,k} & \sigma_{1,k+1} & \cdots & \sigma_{1,k+\delta} & \cdots & \sigma_{m,k} & \sigma_{m,k+1} & \cdots & \sigma_{m,k+\delta} \end{bmatrix}^{\mathrm{T}} \tag{5.64}$$

such that

$$\tilde{\boldsymbol{\xi}}_{k+1} = \bar{\boldsymbol{A}}_{22}\tilde{\boldsymbol{\xi}}_k \tag{5.65}$$

with

$$\bar{\boldsymbol{A}}_{22} = \begin{bmatrix} \bar{\boldsymbol{A}}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \bar{\boldsymbol{A}}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \bar{\boldsymbol{A}}_m \end{bmatrix}, \quad \bar{\boldsymbol{A}}_i = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_i & 0 & 0 & \cdots & 0 \end{bmatrix}, \quad i = 1, 2, \ldots, m. \tag{5.66}$$

Consequently, the eigenvalues of $\tilde{\boldsymbol{A}}_{22}$ are the eigenvalues of all $\bar{\boldsymbol{A}}_i$ blocks. As these blocks are given in controllability canonical form, the characteristic polynomial is given by

$$z^{(\delta+1)} - \alpha_i. \tag{5.67}$$

Hence, the eigenvalues are given by (5.56). □

Remark 5.8. Note that only the magnitude of the $m(\delta+1)$ eigenvalues is influenced by $\boldsymbol{\Gamma}$. The phases of these eigenvalues solely depend on the time delay and can therefore not be changed by the elements of $\boldsymbol{\Gamma}$.

Example 5.9. The switching reaching law proposed in this section is now applied to the unstable second order system

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \begin{bmatrix} 0.2 & 0.3 \\ 0 & 0.1 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix} (u^* + f). \tag{5.68}$$

The constant sampling time is defined as $T = 0.1$s and the bounds of the summed network-induced delay are known to be $0 \leq \tau_k \leq 5T$. Considering these bounds, the buffer can be designed in a way to ensure a constant overall time delay of $5T$, which yields $\delta = 5$. Discretizing plant (5.68) and considering δ results in the discrete time representation

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} 1.02 & 0.03 \\ 0 & 1.01 \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} 0.01 \\ 0.01 \end{bmatrix} (u_{k-5} + f_k). \tag{5.69}$$

Introducing the lifted state vector

$$\boldsymbol{\xi}_k = \begin{bmatrix} \boldsymbol{x}_k^{\mathrm{T}} & u_{k-1} & u_{k-2} & u_{k-3} & u_{k-4} & u_{k-5} \end{bmatrix}^{\mathrm{T}} \tag{5.70}$$

results in the lifted model as in (5.3). Vector

$$\boldsymbol{m}^{\mathrm{T}} = \begin{bmatrix} 97.535 & 0.005 \end{bmatrix} \tag{5.71}$$

was designed by placing the pole of the remaining dynamics in ideal quasi-sliding mode to $\lambda_1 = e^{-0.2T}$ using Ackermann's formula described in section 4.2. The perturbation f_k was chosen as

$$f_k = \sin\left(\sqrt{2}kT\right) + \sin\left(\frac{3kT}{5}\right) + 1 \tag{5.72}$$

which is bounded by

$$\underline{f} = -1, \qquad \bar{f} = 3. \tag{5.73}$$

The mean $\hat{f} = 1$ and magnitude $\tilde{f} = 2$ of this perturbation lead to the mean $\hat{p}$, amplitude $\tilde{p}$, lower bound $\underline{p}$ and upper bound $\bar{p}$ of perturbation p_k using (5.41)–(5.43) with the values

$$\hat{p} = 6.777, \qquad \tilde{p} = 13.555, \qquad \underline{p} = -6.777, \qquad \bar{p} = 20.332. \tag{5.74}$$

To apply the switching reaching law (5.48), the linear parameter α_1 was chosen as $\alpha_1 = e^{-3T}$ and the gain $\rho_1 = 13.7 > \tilde{p}$. Using these settings, the quasi-sliding mode band given by (5.55) equals

$$\Delta = 54.51. \tag{5.75}$$

The simulation results of the sliding variable σ_k for the initial condition

$$\boldsymbol{x}_0 = \begin{bmatrix} 10 & 20 \end{bmatrix}^{\mathrm{T}} \tag{5.76}$$

are shown in figure 5.4. As this figure shows, it is not possible to influence the sliding variable in the first five sampling steps as the first control sample u_0 is received after a five steps delay. Starting with the sixth step, the control samples are applied and the sliding variable moves towards the quasi-sliding mode band depicted in gray. After $k^* = 66$, i.e. 6.6s, the sliding variable σ_k remains in the quasi-sliding mode band. To illustrate that reaching law (5.48) has the same effect as applying the classical switching reaching law to $\delta + 1$ consecutive elements of σ_k, the down-sampled sliding variables s_l with different offsets are shown in figure 5.5. This figure clearly shows the convergence typical of the switching reaching law of each of the down-sampled sliding variables. Simulation results of the plant states $\boldsymbol{x}_k$ as well as the actuating signal u_k are shown in figure 5.6. The actuating signal in particular reveals a disadvantage of this approach since the actuating signal is very large for u_{6k} and small for the remaining elements. This introduces high-frequency oscillations which usually are undesired in practical applications.

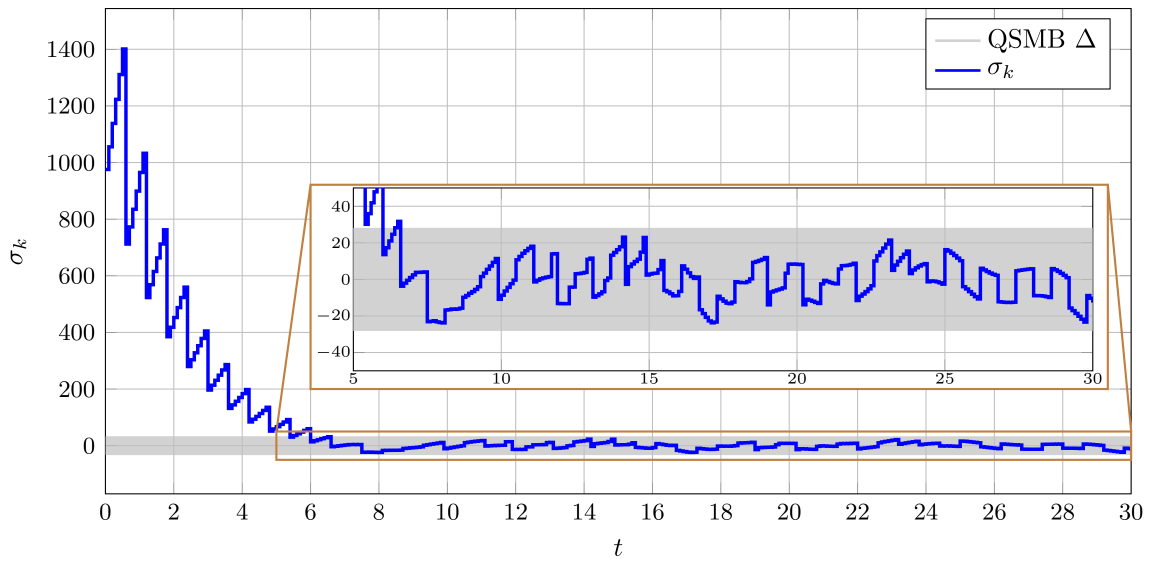


Figure 5.4: Example 5.9: Sliding variable σ_k using the switching reaching law (5.48).

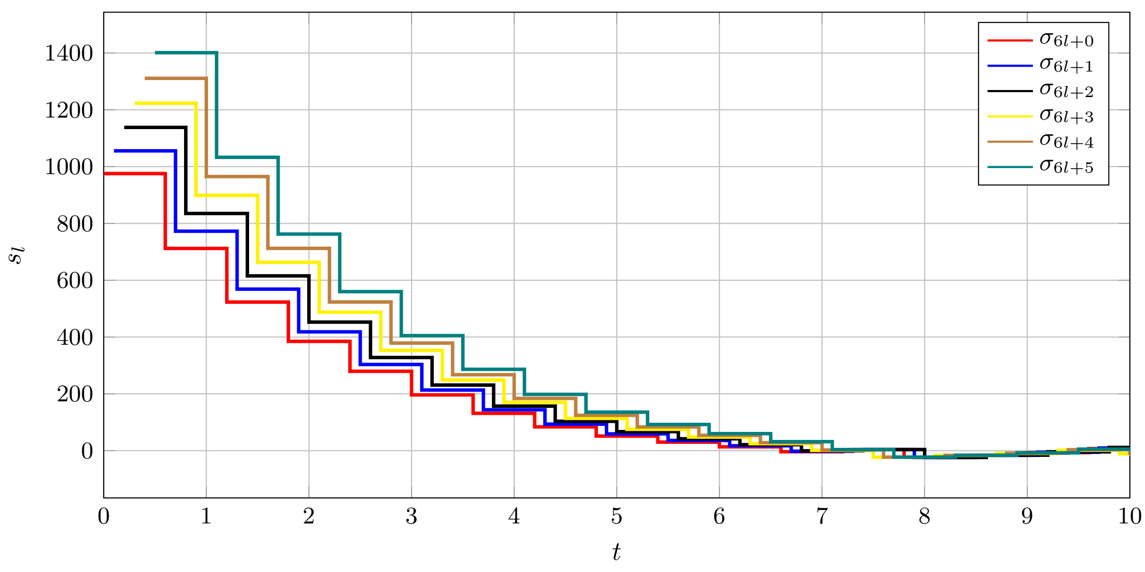


Figure 5.5: Example 5.9: Down-sampled sliding variables s_l with different offsets using the switching reaching law (5.48).

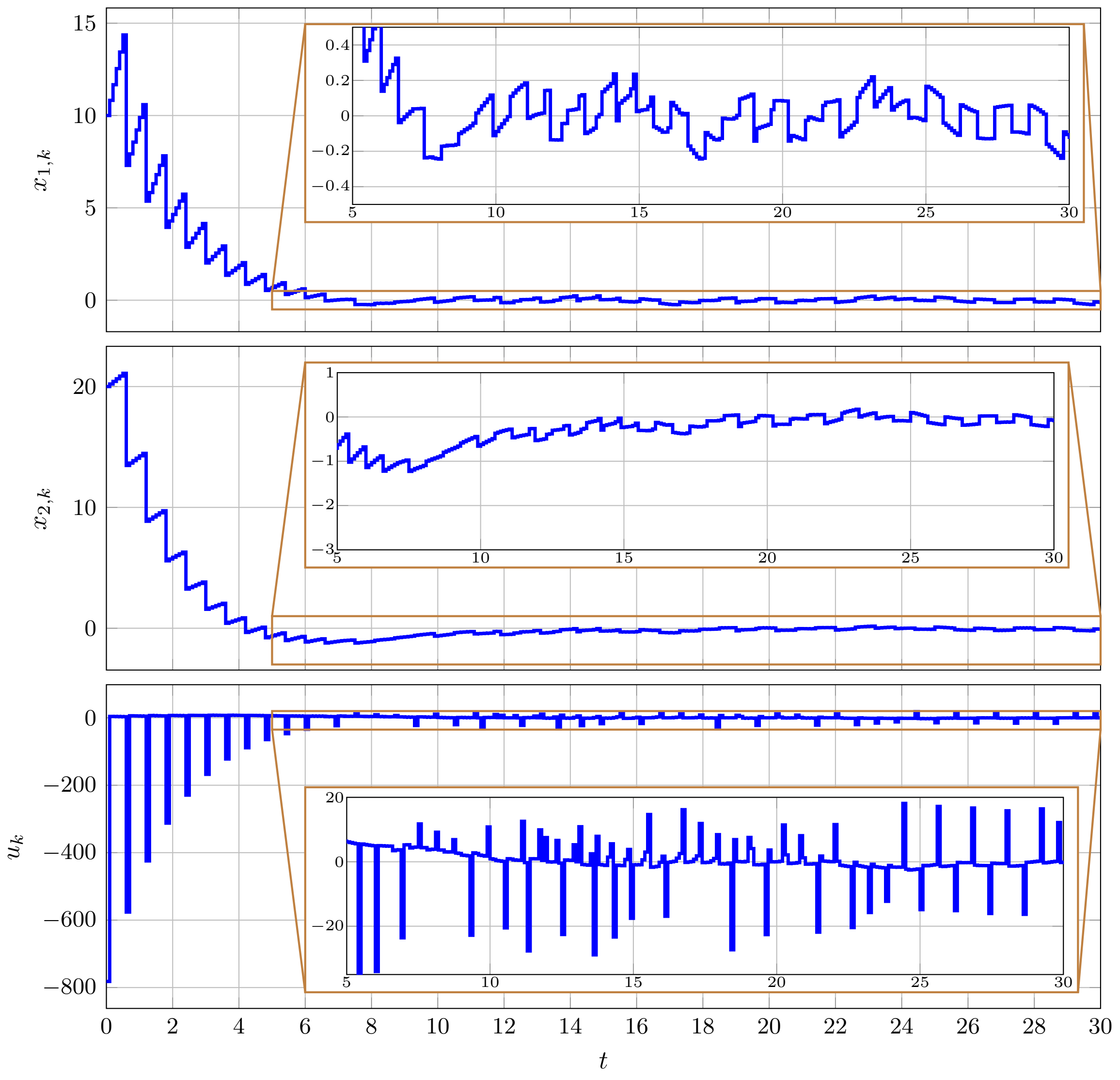


Figure 5.6: Example 5.9: Plant states $\boldsymbol{x}_k$ and the input signal u_k using the switching reaching law (5.48).

5.2.2 The Nonswitching Reaching Law

The nonswitching reaching law (4.81) and even the nonswitching reaching law for higher relative degrees as stated in [47] are not directly applicable in the networked case. In fact, they need to be adapted for the case of non-symmetrical perturbations and the initialization has to be adapted to ensure strict monotonic convergence of the sliding variable. Applying the nonswitching reaching law (4.75) and (4.81) to the i$^{\text{th}}$ component of the sliding variable results in

$$\sigma_{i,k+\delta+1} = \hat{\sigma}_{i,k} - \hat{p}_i + p_{i,k} \tag{5.77}$$

with

$$\hat{\sigma}_{i,k} = \begin{cases} \frac{k_i^* - k}{k_i^*}\hat{\sigma}_{i,0} & k < k_i^* \\ 0 & \text{else} \end{cases} \tag{5.78}$$

where k_i^* and $\hat{\sigma}_{i,0}$ have to be chosen in a way that

$$|\sigma_{i,k+1}| < |\sigma_{i,k}| \qquad \text{and} \tag{5.79a}$$

$$\operatorname{sign}(\sigma_{i,k+1}) = \operatorname{sign}(\sigma_{i,k}) \tag{5.79b}$$

are satisfied for $\delta \le k < k_i^*$ as given in (4.76) and (4.78). It is evident that (5.79) cannot be satisfied for $0 \le k < \delta$ in the networked control case as the first control signal is still propagating through the network and therefore the plant is operated in open loop in this period. In order to satisfy (5.79) for $\delta \le k \le k_i^*$, the future value of the sliding variable $\sigma_{i,\delta}$ with the smallest possible magnitude $\boldsymbol{\sigma}_{\text{est}}$ has to be predicted. This estimate is then used to set $\hat{\sigma}_{i,0}$ such that the magnitude of $\sigma_{i,\delta+1}$ is smaller than the magnitude of $\sigma_{i,\delta}$.

Using (5.4) and (5.34) and assuming that $\boldsymbol{u}_{-\delta} = \boldsymbol{u}_{-\delta+1} = \cdots = \boldsymbol{u}_{-1} = \boldsymbol{0}$, the prediction of the sliding variable $\boldsymbol{\sigma}_\delta$ can be written as

$$\boldsymbol{\sigma}_\delta = \boldsymbol{M}\boldsymbol{A}^\delta\boldsymbol{x}_0 + \sum_{i=0}^{\delta-1}\boldsymbol{M}\boldsymbol{A}^i\boldsymbol{B}\boldsymbol{f}_{\delta-1-i}. \tag{5.80}$$

Using the mean $\hat{\boldsymbol{f}}$ and magnitude $\tilde{\boldsymbol{f}}$ of perturbation $\boldsymbol{f}_k$, the mean

$$\hat{\boldsymbol{\sigma}}_{\text{est}} = \begin{bmatrix}\hat{\sigma}_{1,\text{est}} & \hat{\sigma}_{2,\text{est}} & \cdots & \hat{\sigma}_{m,\text{est}}\end{bmatrix}^{\text{T}} = \boldsymbol{M}\boldsymbol{A}^\delta\boldsymbol{x}_0 + \sum_{i=0}^{\delta-1}\boldsymbol{M}\boldsymbol{A}^i\boldsymbol{B}\hat{\boldsymbol{f}} \tag{5.81}$$

and the magnitude

$$\tilde{\boldsymbol{\sigma}}_{\text{est}} = \begin{bmatrix}\tilde{\sigma}_{1,\text{est}} & \tilde{\sigma}_{2,\text{est}} & \cdots & \tilde{\sigma}_{m,\text{est}}\end{bmatrix}^{\text{T}} = \sum_{i=0}^{\delta-1}\left|\boldsymbol{M}\boldsymbol{A}^i\boldsymbol{B}\right|\tilde{\boldsymbol{f}} \tag{5.82}$$

of this estimation can be computed. Using these values, the estimation of $\boldsymbol{\sigma}_\delta$ with the smallest possible amplitude is given by

$$\boldsymbol{\sigma}_{\text{est}} = \begin{bmatrix}\sigma_{1,\text{est}} & \sigma_{2,\text{est}} & \cdots & \sigma_{m,\text{est}}\end{bmatrix}^{\text{T}} \tag{5.83}$$

with

$$\sigma_{i,\mathrm{est}} = \begin{cases} \hat{\sigma}_{i,\mathrm{est}} - \tilde{\sigma}_{i,\mathrm{est}} \operatorname{sign}(\hat{\sigma}_{i,\mathrm{est}}) & \text{if } |\hat{\sigma}_{i,\mathrm{est}}| > \tilde{\sigma}_{i,\mathrm{est}} \\ 0 & \text{else} \end{cases} . \tag{5.84}$$

Applying (5.77) to (5.45) results in the control law

$$\boldsymbol{u}_k = -\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta+1}\boldsymbol{\xi}_k + \hat{\boldsymbol{\sigma}}_k - \hat{\boldsymbol{p}} \tag{5.85}$$

with $\hat{\boldsymbol{\sigma}}_k = \begin{bmatrix}\hat{\sigma}_{1,k} & \hat{\sigma}_{2,k} & \cdots & \hat{\sigma}_{m,k}\end{bmatrix}^{\mathrm{T}}$.

Theorem 5.10. Consider the centralized buffered networked control system depicted in figure 5.1 with plant (1.5) and sampling time T. Also, assume that assumptions 1.2, 1.4, 2.4 and 2.6 are satisfied and perturbation $\boldsymbol{f}_k$ is bounded by the known bounds $\underline{\boldsymbol{f}} \leq \boldsymbol{f}_k \leq \bar{\boldsymbol{f}}$ $\forall k$. Let the controller be given by (5.77), (5.78) and (5.85) with $\hat{\sigma}_{i,0}$ such that

$$\begin{cases} \tilde{p}_i < \hat{\sigma}_{i,0} < \sigma_{i,\mathrm{est}} - \tilde{p}_i & \text{if} \quad \sigma_{i,\mathrm{est}} - \tilde{p}_i > \tilde{p}_i \\ -\tilde{p}_i > \hat{\sigma}_{i,0} > \sigma_{i,\mathrm{est}} + \tilde{p}_i & \text{if} \quad \sigma_{i,\mathrm{est}} + \tilde{p}_i < -\tilde{p}_i \\ \hat{\sigma}_{i,0} = 0 & \text{else} \end{cases} , \tag{5.86}$$

$$k_i^* < \frac{|\hat{\sigma}_{i,0}|}{\tilde{r}_i} \tag{5.87}$$

with $\sigma_{i,\mathrm{est}}$ given by (5.84) and the exact limit

$$\tilde{\boldsymbol{r}} = \begin{bmatrix}\tilde{r}_1 & \tilde{r}_2 & \cdots & \tilde{r}_m\end{bmatrix}^{\mathrm{T}} = \left[\left|\hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f\right| + \left|\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta}\hat{\boldsymbol{B}}_f\right| + \sum_{i=1}^{\delta}\left|\hat{\boldsymbol{M}}\left(\hat{\boldsymbol{A}}^{i} - \hat{\boldsymbol{A}}^{i-1}\right)\hat{\boldsymbol{B}}_f\right|\right]\tilde{\boldsymbol{f}}. \tag{5.88}$$

Then, the states of the plant are ultimately bounded if $\hat{\boldsymbol{M}}$ is defined as proposed in theorem 5.2. The quasi-sliding mode band of the ith sliding variable is then given by

$$\Delta_i = 2\tilde{p}_i. \tag{5.89}$$

Proof. The initial value of the ith desired sliding variable $\hat{\sigma}_{i,0}$ has to be chosen in a way to satisfy (5.79a) for $k = \delta$, i.e.

$$|\sigma_{i,\delta+1}| = |\hat{\sigma}_{i,0} - \hat{p}_i + p_{i,\delta}| < |\sigma_{i,\delta}|\,. \tag{5.90}$$

Using the estimation of $\sigma_{i,\delta}$ with the smallest possible amplitude $\sigma_{i,\mathrm{est}}$ gives

$$|\hat{\sigma}_{i,0} - \hat{p}_i + p_{i,\delta}| < |\sigma_{i,\mathrm{est}}|\,. \tag{5.91}$$

Assume that $\sigma_{i,\mathrm{est}} > 0$. It therefore follows that $\sigma_{i,\delta+1} > 0$ due to (5.79b). Thus, the two inequalities

$$\sigma_{i,\delta+1} = \hat{\sigma}_{i,0} - \hat{p}_i + p_{i,\delta} > 0 \qquad \text{and} \tag{5.92a}$$

$$\hat{\sigma}_{i,0} - \hat{p}_i + p_{i,\delta} < \sigma_{i,\mathrm{est}} \tag{5.92b}$$

must be satisfied. Considering the worst case bound $|p_{i,k} - \hat{p}_i| < \tilde{p}_i$ and combining the two inequalities yields

$$\tilde{p}_i < \hat{\sigma}_{i,0} < \sigma_{i,\text{est}} - \tilde{p}_i. \tag{5.93}$$

This inequality corresponds to the first line in (5.86) and can only be satisfied if

$$\sigma_{i,\text{est}} - \tilde{p}_i > \tilde{p}_i. \tag{5.94}$$

The second line follows by analyzing (5.91) for the case in which $\sigma_{i,\text{est}} < 0$. The third line ensures that $\sigma_{i,\delta+1}$ is within the quasi-sliding mode band if $\sigma_{i,\text{est}}$ is close enough to that band.

The gradient of the i$^{\text{th}}$ desired sliding variable defined by k_i^* has to be chosen such that (5.79a) is satisfied. Using (5.77) and (5.78) to compute $\sigma_{i,k+\delta+2}$ results in

$$\sigma_{i,k+\delta+2} = \underbrace{\frac{k_i^* - k}{k_i^*}\hat{\sigma}_{i,0} - \hat{p}_i}_{\sigma_{i,k+\delta+1} - p_{i,k}} - \frac{1}{k_i^*}\hat{\sigma}_{i,0} + p_{i,k+1} \tag{5.95}$$

for $k < k_i^*$. Inserting the perturbations

$$\boldsymbol{p}_k = \sum_{i=0}^{\delta} \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-i}\hat{\boldsymbol{B}}_f \boldsymbol{f}_{k+i} = \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta}\hat{\boldsymbol{B}}_f \boldsymbol{f}_k + \sum_{i=1}^{\delta} \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-i}\hat{\boldsymbol{B}}_f \boldsymbol{f}_{k+i} \tag{5.96}$$

and

$$\boldsymbol{p}_{k+1} = \sum_{i=0}^{\delta} \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-i}\hat{\boldsymbol{B}}_f \boldsymbol{f}_{k+i+1} = \hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f \boldsymbol{f}_{k+\delta+1} + \sum_{i=1}^{\delta} \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-i+1}\hat{\boldsymbol{B}}_f \boldsymbol{f}_{k+i} \tag{5.97}$$

from (5.36) into (5.95) gives

$$\sigma_{i,k+\delta+2} = \sigma_{i,k+\delta+1} - \frac{1}{k_i^*}\hat{\sigma}_{i,0} + r_{i,k} \tag{5.98}$$

with

$$\begin{aligned} \boldsymbol{r}_k &= \begin{bmatrix} r_{1,k} & r_{2,k} & \cdots & r_{m,k} \end{bmatrix}^{\mathrm{T}} \\ &= \hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f \boldsymbol{f}_{k+\delta+1} - \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta}\hat{\boldsymbol{B}}_f \boldsymbol{f}_k + \sum_{i=1}^{\delta} \hat{\boldsymbol{M}}\left(\hat{\boldsymbol{A}}^{\delta-i+1} - \hat{\boldsymbol{A}}^{\delta-i}\right)\hat{\boldsymbol{B}}_f \boldsymbol{f}_{k+i}. \end{aligned} \tag{5.99}$$

Using the mean $\hat{\boldsymbol{f}}$ of perturbation $\boldsymbol{f}_k$ in (5.99) results in the mean

$$\hat{\boldsymbol{r}} = \left[\hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f - \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta}\hat{\boldsymbol{B}}_f + \sum_{i=1}^{\delta} \hat{\boldsymbol{M}}\left(\hat{\boldsymbol{A}}^{\delta-i+1} - \hat{\boldsymbol{A}}^{\delta-i}\right)\hat{\boldsymbol{B}}_f\right]\hat{\boldsymbol{f}} = \boldsymbol{0} \tag{5.100}$$

and using the amplitude $\tilde{\boldsymbol{f}}$ of perturbation $\boldsymbol{f}_k$ in (5.99) results in the amplitude $\tilde{\boldsymbol{r}}$ as given in (5.88). From (5.98), (5.100) and (5.88) one can conclude that (5.79a) is always satisfied if k_i^* is chosen as proposed in (5.87).

To derive the quasi-sliding mode band, consider the reaching law (5.77) for $k > k_i^*$ which is given by

$$\sigma_{i,k+\delta+1} = -\hat{p}_i + p_{i,k}. \tag{5.101}$$

Computing the difference of the upper and the lower bound of $\sigma_{i,k+\delta+1}$ in (5.101) results in the quasi-sliding mode band as given in (5.89). $\square$

Example 5.11. The nonswitching reaching law based controller is now applied to the networked control system described in example 5.9. Using the initial condition (5.76) to evaluate (5.81) and (5.82) gives

$$\hat{\sigma}_{\text{est}} = 1398.979, \qquad \tilde{\sigma}_{\text{est}} = 11.028. \tag{5.102}$$

Applying these values to (5.83) leads to

$$\sigma_{\text{est}} = 1387.951. \tag{5.103}$$

Therefore, a valid choice with respect to (5.86) is

$$\hat{\sigma}_0 = 1374. \tag{5.104}$$

Note that σ_{est} cannot be computed in advance. However, it is computed directly after the measurement $\boldsymbol{x}_0$ is available at the controller. Evaluating (5.88) results in

$$\tilde{r} = 5.055. \tag{5.105}$$

Consequently, relation (5.87) evaluates to

$$k^* < 271.827 \tag{5.106}$$

and is satisfied by

$$k^* = 50 \tag{5.107}$$

which was chosen to achieve a similar dynamical behavior as with the switching reaching law in example 5.9. Evaluating (5.89) results in the width of the quasi-sliding mode band

$$\Delta = 27.11. \tag{5.108}$$

Simulation results of the sliding variable σ_k with initial condition (5.76) are shown in figure 5.7. The green line in this figure shows the desired sliding variable $\hat{\sigma}_k$ which starts at $\hat{\sigma}_0$ and decreases monotonically until it reaches zero at $t = Tk^*$. The actual sliding variable σ_k increases in the first δ samples as the first control sample has not yet arrived at the plant. Next, the sliding variable σ_k follows $\hat{\sigma}_k$, as expected, with $\delta + 1$ delays and finally reaches the quasi-sliding mode band Δ and stays in it.

The simulation results for the plant states $\boldsymbol{x}_k$ and the actuating signal u_k are shown in figure 5.8. A comparison of the switching and the nonswitching reaching law for centralized buffered networked systems is shown in section 5.3.

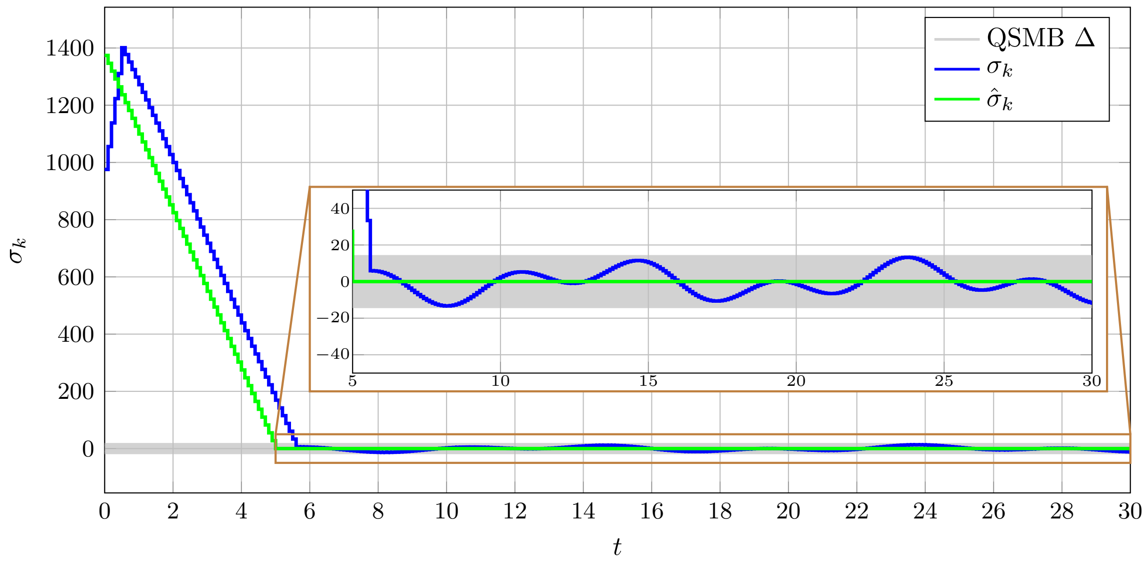


Figure 5.7: Example 5.11: Sliding variable σ_k and the desired sliding variable $\hat{\sigma}_k$ using the non-switching reaching law (5.77).

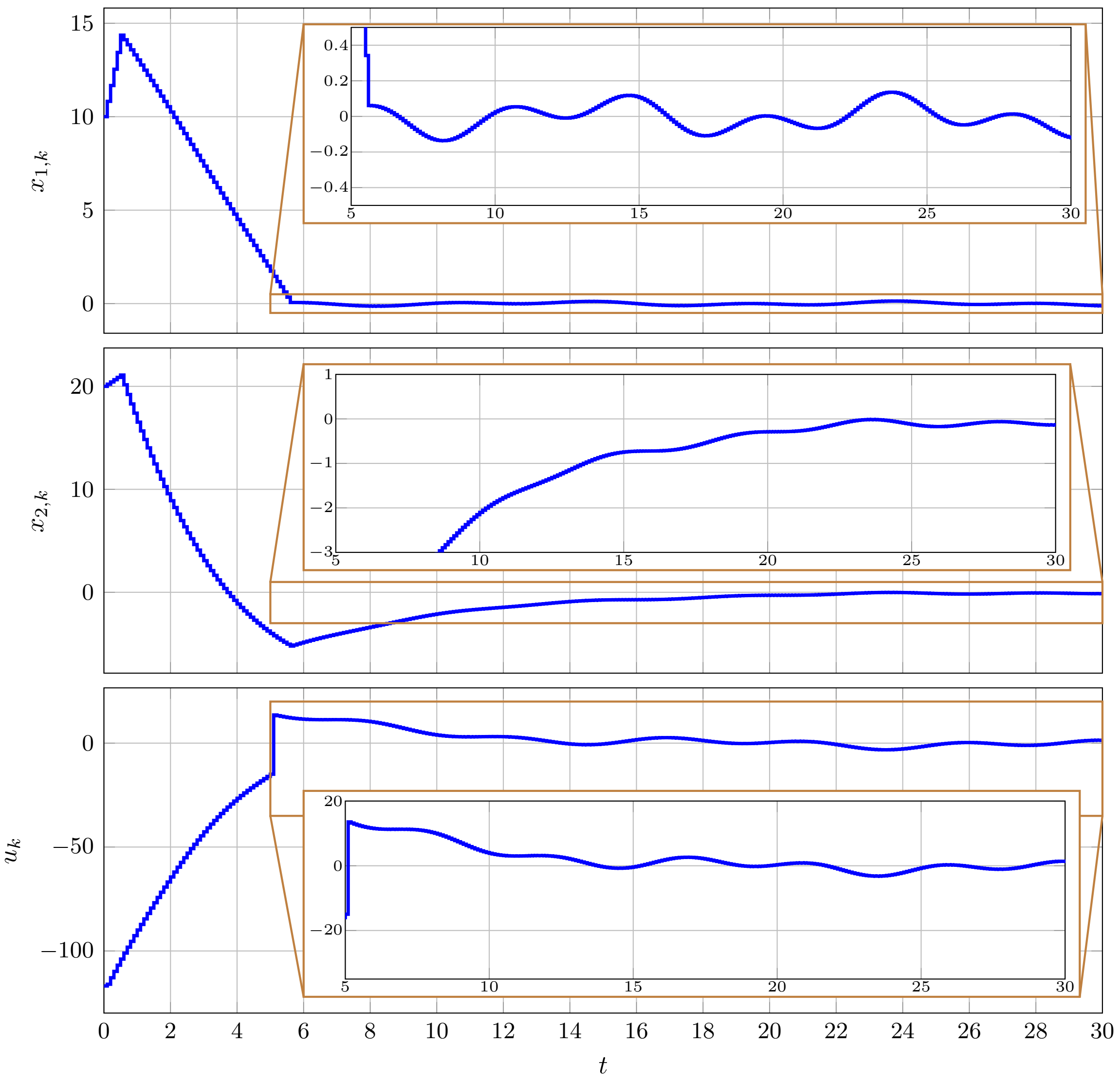


Figure 5.8: Example 5.11: Plant states $\boldsymbol{x}_k$ and the input signal u_k using the nonswitching reaching law (5.77).

5.2.3 The Predictive Switching Reaching Law

The switching reaching law given in section 5.2.1 has the drawback that $\delta+1$ consecutive elements converge independently of one another which could lead to high-frequency oscillations during the reaching phase (see simulation results in figure 5.4). As a consequence, the resulting control signal can switch between very high and very low values (see figure 5.6), which is typically undesired in practical applications. The reaching law proposed in this chapter is based on the switching reaching law but ensures exponential convergence of the sliding variable.

Considering the forward increment of the sliding variable derived from (5.34) and using the mean $\hat{\boldsymbol{f}}$ of perturbation $\boldsymbol{f}_k$ gives

$$\boldsymbol{\sigma}_{k+\delta} = \bar{\boldsymbol{\sigma}}_{k+\delta} + \sum_{j=0}^{\delta-1} \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j-1}\hat{\boldsymbol{B}}_f \tilde{\boldsymbol{f}}_{k+j} \tag{5.109}$$

with

$$\bar{\boldsymbol{\sigma}}_{k+\delta} = \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta}\boldsymbol{\xi}_k + \sum_{j=0}^{\delta-1} \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j-1}\hat{\boldsymbol{B}}_f \hat{\boldsymbol{f}} \tag{5.110}$$

$$\boldsymbol{f}_k = \tilde{\boldsymbol{f}}_k + \hat{\boldsymbol{f}}. \tag{5.111}$$

Since $\bar{\boldsymbol{\sigma}}_{k+\delta}$ contains all known quantities of $\boldsymbol{\sigma}_{k+\delta}$, it can be considered a prediction of $\boldsymbol{\sigma}_{k+\delta}$ with the mean perturbation $\hat{\boldsymbol{f}}$. The error made in this prediction is given by $\sum_{j=0}^{\delta-1} \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j-1}\hat{\boldsymbol{B}}_f \tilde{\boldsymbol{f}}_{k+j}$. Using the predicted sliding variable $\bar{\boldsymbol{\sigma}}_{k+\delta}$ in the reaching law introduces significant advantages as shown in this section.

Using the predicted sliding variable $\bar{\boldsymbol{\sigma}}_{k+\delta}$, the predictive switching reaching law is formulated as follows

$$\begin{gathered}\boldsymbol{\sigma}_{k+\delta+1} = \boldsymbol{\Gamma}\bar{\boldsymbol{\sigma}}_{k+\delta} - \boldsymbol{S}\,\mathrm{sign}\left(\bar{\boldsymbol{\sigma}}_{k+\delta}\right) - \hat{\boldsymbol{p}} + \boldsymbol{p}_k \\ \boldsymbol{\Gamma} = \begin{bmatrix} \alpha_1 & 0 & \cdots & 0 \\ 0 & \alpha_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_m \end{bmatrix}, \quad \boldsymbol{S} = \begin{bmatrix} \rho_1 & 0 & \cdots & 0 \\ 0 & \rho_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \rho_m \end{bmatrix}\end{gathered} \tag{5.112}$$

where $\alpha_i \in \mathbb{R}$, $0 \le \alpha_i < 1$, $i = 1, 2, \ldots, m$ and $\rho_i > 0$. In contrast to the switching reaching law proposed in section 5.2.1, the predictive switching reaching law does not use the current value of the sliding variable $\boldsymbol{\sigma}_k$ but its predicted future value $\bar{\boldsymbol{\sigma}}_{k+\delta}$. The perturbation acting on $\boldsymbol{\sigma}_{k+\delta+1}$ is obtained by inserting (5.36), (5.42), (5.109) and (5.111) into (5.112), which results in

$$\begin{aligned}\boldsymbol{\sigma}_{k+\delta+1} &= \boldsymbol{\Gamma}\boldsymbol{\sigma}_{k+\delta} - \boldsymbol{S}\,\mathrm{sign}\left(\bar{\boldsymbol{\sigma}}_{k+\delta}\right) + \boldsymbol{p}_k - \hat{\boldsymbol{p}} - \sum_{j=0}^{\delta-1} \boldsymbol{\Gamma}\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j-1}\hat{\boldsymbol{B}}_f \tilde{\boldsymbol{f}}_{k+j} \\ &= \boldsymbol{\Gamma}\boldsymbol{\sigma}_{k+\delta} - \boldsymbol{S}\,\mathrm{sign}\left(\bar{\boldsymbol{\sigma}}_{k+\delta}\right) + \boldsymbol{q}_k\end{aligned} \tag{5.113}$$

with

$$\tilde{\boldsymbol{p}}_k = \begin{bmatrix}\tilde{p}_{1,k} & \tilde{p}_{2,k} & \cdots & \tilde{p}_{m,k}\end{bmatrix}^{\mathrm{T}} = \boldsymbol{p}_k - \hat{\boldsymbol{p}} = \sum_{j=0}^{\delta} \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j}\hat{\boldsymbol{B}}_f\tilde{\boldsymbol{f}}_{k+j} \tag{5.114}$$

$$\begin{aligned}\boldsymbol{q}_k = \begin{bmatrix}q_{1,k} & q_{2,k} & \cdots & q_{m,k}\end{bmatrix}^{\mathrm{T}} &= \tilde{\boldsymbol{p}}_k - \sum_{j=0}^{\delta-1}\boldsymbol{\Gamma}\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j-1}\hat{\boldsymbol{B}}_f\tilde{\boldsymbol{f}}_{k+j} \\ &= \sum_{j=0}^{\delta-1}\left(\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j}\hat{\boldsymbol{B}}_f - \boldsymbol{\Gamma}\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j-1}\hat{\boldsymbol{B}}_f\right)\tilde{\boldsymbol{f}}_{k+j} + \hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f\tilde{\boldsymbol{f}}_{k+\delta}.\end{aligned} \tag{5.115}$$

The elements ρ_i of $\boldsymbol{S}$ have to be chosen in a way that they always dominate the i$^{\mathrm{th}}$ element of perturbation $\boldsymbol{q}_k$. The worst case magnitude of this perturbation is given by

$$\tilde{\boldsymbol{q}} = \sum_{j=0}^{\delta-1}\left|\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j}\hat{\boldsymbol{B}}_f - \boldsymbol{\Gamma}\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j-1}\hat{\boldsymbol{B}}_f\right|\tilde{\boldsymbol{f}} + \left|\hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f\right|\tilde{\boldsymbol{f}}. \tag{5.116}$$

Inserting (5.110) and (5.112) into (5.45) yields the control law

$$\boldsymbol{u}_k = -\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta+1}\boldsymbol{\xi}_k + \boldsymbol{\Gamma}\bar{\boldsymbol{\sigma}}_{k+\delta} - \boldsymbol{S}\,\mathrm{sign}\left(\bar{\boldsymbol{\sigma}}_{k+\delta}\right) - \hat{\boldsymbol{p}} \tag{5.117}$$

$$\begin{aligned}&= \left(\boldsymbol{\Gamma}\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta} - \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta+1}\right)\boldsymbol{\xi}_k - \boldsymbol{S}\,\mathrm{sign}\left(\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta}\boldsymbol{\xi}_k + \sum_{j=0}^{\delta-1}\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j-1}\hat{\boldsymbol{B}}_f\hat{\boldsymbol{f}}\right) \\ &\quad - \hat{\boldsymbol{p}} + \sum_{j=0}^{\delta-1}\boldsymbol{\Gamma}\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta-j-1}\hat{\boldsymbol{B}}_f\hat{\boldsymbol{f}}.\end{aligned} \tag{5.118}$$

Applying (5.118) to (5.3) in the perturbation free case, i.e. $\boldsymbol{f}_k = \boldsymbol{0}$ and $\boldsymbol{S} = \boldsymbol{0}$, results in the nominal closed loop system

$$\boldsymbol{\xi}_{k+1} = \left[\hat{\boldsymbol{A}} - \hat{\boldsymbol{B}}\left(\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta+1} - \boldsymbol{\Gamma}\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}^{\delta}\right)\right]\boldsymbol{\xi}_k. \tag{5.119}$$

Theorem 5.12. Consider the centralized buffered networked control system depicted in figure 5.1 with plant (1.5) and sampling time T. Also, assume that assumptions 1.2, 1.4, 2.4 and 2.6 are satisfied and perturbation $\boldsymbol{f}_k$ is bounded by the known bounds $\underline{\boldsymbol{f}} \leq \boldsymbol{f}_k \leq \bar{\boldsymbol{f}}\ \forall k$. Let the controller be given by (5.118) with the elements of $\boldsymbol{\Gamma}$ given by the solution of the optimization problem

$$\begin{aligned}\alpha_i = \arg\min_{\underline{\alpha}_i \leq \alpha_i^* \leq \bar{\alpha}_i} &\sum_{j=0}^{\delta-1}\left|\hat{\boldsymbol{m}}_i^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta-j}\hat{\boldsymbol{B}}_f - \alpha_i^*\hat{\boldsymbol{m}}_i^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta-j-1}\hat{\boldsymbol{B}}_f\right|\tilde{\boldsymbol{f}} \\ \text{s.t. } &\underline{\alpha}_i, \bar{\alpha}_i \in \mathbb{R},\ 0 \leq \underline{\alpha}_i \leq \bar{\alpha}_i < 1\end{aligned} \tag{5.120}$$

and the elements of $\boldsymbol{S}$ satisfying

$$\rho_i > \sum_{j=0}^{\delta-1} \left| \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{A}}^{\delta-j} \hat{\boldsymbol{B}}_f - \alpha_i \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{A}}^{\delta-j-1} \hat{\boldsymbol{B}}_f \right| \tilde{\boldsymbol{f}} + \left| \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{B}}_f \right| \tilde{\boldsymbol{f}}. \tag{5.121}$$

Then, the states of the plant are ultimately bounded if $\hat{\boldsymbol{M}}$ is designed as proposed in theorem 5.2. The quasi-sliding mode band for the i$^{\text{th}}$ sliding variable is then given by

$$\Delta_i = 2(\rho_i + \tilde{p}_i) \tag{5.122}$$

which is minimized by the specific choice of α_i. Moreover, the eigenvalues of the nominal closed loop system (5.119) are given by

- m eigenvalues at α_i,
- $m \cdot \delta$ eigenvalues at zero and
- the remaining $n - m$ eigenvalues are specified during the design process of $\hat{\boldsymbol{M}}$.

Proof. Applying control law (5.118) to (5.3) ensures that the dynamics of the sliding variable are given by the reaching law (5.112). Using (5.109), the reaching law can be reformulated to (5.113) with the worst case perturbation magnitude (5.116). Large enough values have to be chosen for the parameters ρ_i to dominate the amplitude of perturbation $q_{i,k}$, which is ensured by the inequality in (5.121).

To analyze the width of the i$^{\text{th}}$ quasi-sliding mode band Δ_i, consider the i$^{\text{th}}$ element of the reaching law in the form

$$\sigma_{i,k+\delta+1} = \alpha_i \sigma_{i,k+\delta} - \rho_i \operatorname{sign}(\bar{\sigma}_{i,k+\delta}) + q_{i,k}. \tag{5.123}$$

Furthermore, assuming $\bar{\sigma}_{i,k+\delta} = -\varepsilon$ with $\varepsilon > 0$ and using (5.109) yields

$$\sigma_{i,k+\delta} = -\varepsilon + \sum_{j=0}^{\delta-1} \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{A}}^{\delta-j-1} \hat{\boldsymbol{B}}_f \tilde{\boldsymbol{f}}_{k+j} \quad \text{with } \varepsilon > 0. \tag{5.124}$$

Applying (5.115) and (5.124) to (5.123) results in

$$\begin{aligned} \sigma_{i,k+\delta+1} &= \sum_{j=0}^{\delta-1} \alpha_i \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{A}}^{\delta-j-1} \hat{\boldsymbol{B}}_f \tilde{\boldsymbol{f}}_{k+j} - \alpha_i \varepsilon + \rho_i + \underbrace{\tilde{p}_{i,k} - \sum_{j=0}^{\delta-1} \alpha_i \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{A}}^{\delta-j-1} \hat{\boldsymbol{B}}_f \tilde{\boldsymbol{f}}_{k+j}}_{q_{i,k}} \\ &= -\alpha_i \varepsilon + \rho_i + \tilde{p}_{i,k} \end{aligned} \tag{5.125}$$

The upper bound of (5.123) is then given by

$$\sigma_{i,k+\delta+1} \leq \rho_i + \tilde{p}_i. \tag{5.126}$$

Analog to (5.126), the lower bound is calculated by assuming $\bar{\sigma}_{i,k+\delta} = \varepsilon$ with $\varepsilon > 0$ and using (5.109), which yields

$$\sigma_{i,k+\delta} = \varepsilon + \sum_{j=0}^{\delta-1} \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{A}}^{\delta-j-1} \hat{\boldsymbol{B}}_f \tilde{\boldsymbol{f}}_{k+j} \quad \text{with } \varepsilon > 0. \tag{5.127}$$

The lower bound of $\sigma_{i,k+\delta+1}$ is then given by

$$\sigma_{i,k+\delta+1} \geq -\rho_i - \tilde{p}_i. \tag{5.128}$$

As a consequence, the width of the quasi-sliding mode band is given by (5.122). Since ρ_i has to dominate the i$^{\text{th}}$ element of (5.116), the smallest possible magnitude of the quasi-sliding mode band is achieved by minimizing the i$^{\text{th}}$ element of (5.116) using α_i. This leads to the optimization problem given in (5.120).

To analyze the eigenvalues of the nominal closed loop system matrix, consider transformation (5.60). Since the reaching law (5.112) in the nominal case is given by

$$\boldsymbol{\sigma}_{k+\delta+1} = \boldsymbol{\Gamma} \boldsymbol{\sigma}_{k+\delta}, \tag{5.129}$$

applying transformation (5.60) to the nominal closed loop system (5.119) yields

$$\begin{bmatrix} \boldsymbol{x}_{1,k+1} \\ \boldsymbol{\sigma}_{k+1} \\ \boldsymbol{\sigma}_{k+2} \\ \vdots \\ \boldsymbol{\sigma}_{k+\delta+1} \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{A}}_{11} & \tilde{\boldsymbol{A}}_{12} \\ \begin{matrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{matrix} & \underbrace{\boxed{\begin{matrix} \boldsymbol{0} & \boldsymbol{I}_m & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I}_m & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{\Gamma} \end{matrix}}}_{\tilde{\boldsymbol{A}}_{22}} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{1,k} \\ \boldsymbol{\sigma}_{k} \\ \boldsymbol{\sigma}_{k+1} \\ \vdots \\ \boldsymbol{\sigma}_{k+\delta} \end{bmatrix}. \tag{5.130}$$

In this case also, the eigenvalues of the nominal closed loop system are the conjunction of the eigenvalues of $\tilde{\boldsymbol{A}}_{11}$, which are fixed by the design of $\hat{\boldsymbol{M}}$ (see section 4.2), and those of $\tilde{\boldsymbol{A}}_{22}$. As the i$^{\text{th}}$ component of (5.112) in the nominal case equals

$$\sigma_{i,k+\delta+1} = \alpha_i \sigma_{i,k+\delta}, \tag{5.131}$$

the subsystem with system matrix $\tilde{\boldsymbol{A}}_{22}$ in (5.130) can be represented using the state vector

$$\tilde{\boldsymbol{\xi}}_k = \begin{bmatrix} \sigma_{1,k} & \sigma_{1,k+1} & \cdots & \sigma_{1,k+\delta} & \cdots & \sigma_{m,k} & \sigma_{m,k+1} & \cdots & \sigma_{m,k+\delta} \end{bmatrix}^{\mathrm{T}} \tag{5.132}$$

in the form

$$\tilde{\boldsymbol{\xi}}_{k+1} = \bar{\boldsymbol{A}}_{22} \tilde{\boldsymbol{\xi}}_k \tag{5.133}$$

with

$$\bar{\boldsymbol{A}}_{22} = \begin{bmatrix} \bar{\boldsymbol{A}}_1 & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \bar{\boldsymbol{A}}_2 & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \bar{\boldsymbol{A}}_m \end{bmatrix} \qquad \bar{\boldsymbol{A}}_i = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha_i \end{bmatrix} \qquad i = 1, 2, \ldots, m. \tag{5.134}$$

Consequently, the eigenvalues of $\tilde{\boldsymbol{A}}_{22}$ are the eigenvalues of all $\bar{\boldsymbol{A}}_i$ blocks. As these blocks are given in controllability canonical form, the characteristic polynomial is given by

$$z^{(\delta+1)} - \alpha_i z^{\delta}. \tag{5.135}$$

Hence $m\delta$ eigenvalues are zero and the remaining ones are given by α_i. □

Remark 5.13. The optimization problem (5.120) can efficiently be solved by reformulating the optimization problem to end up in a linear program. To describe the necessary steps comprehensibly, the reformulation process is done for a simple example first and then extended to the optimization problem (5.120). Consider the optimization problem

$$\begin{gathered}\min_{\boldsymbol{\omega}} \sum_i |e_i| \\ \text{with } \boldsymbol{e} = \boldsymbol{h} + \boldsymbol{H}\boldsymbol{\omega}\end{gathered} \tag{5.136}$$

with $\boldsymbol{\omega} \in \mathbb{R}^a$, $\boldsymbol{H} \in \mathbb{R}^{b\times a}$ and where $\boldsymbol{h}$, $\boldsymbol{e}$ are of appropriate dimensions. Introducing additional optimization variables $\gamma_i \in \mathbb{R}$, the absolute value terms in (5.136) can be represented by the linear program

$$\begin{aligned} |e_i| &= \min_{\gamma_i} \gamma_i \\ \text{s.t.} & \begin{bmatrix} -1 \\ -1 \end{bmatrix} \gamma_i \leq \begin{bmatrix} e_i \\ -e_i \end{bmatrix}. \end{aligned} \tag{5.137}$$

Using (5.137) and the extended vector of optimization variables

$$\hat{\boldsymbol{\omega}} = \begin{bmatrix} \boldsymbol{\omega}^{\mathrm{T}} & \gamma_1 & \gamma_2 & \cdots & \gamma_b \end{bmatrix}^{\mathrm{T}}, \tag{5.138}$$

the optimization problem (5.136) can be represented by the linear program

$$\begin{aligned} &\min_{\hat{\boldsymbol{\omega}}} \begin{bmatrix} \boldsymbol{0}_{1\times a} & 1 & 1 & \cdots & 1 \end{bmatrix} \hat{\boldsymbol{\omega}} \\ \text{s.t.} & \begin{bmatrix} \boldsymbol{H} & -\boldsymbol{I}_b \\ -\boldsymbol{H} & -\boldsymbol{I}_b \end{bmatrix} \hat{\boldsymbol{\omega}} \leq \begin{bmatrix} -\boldsymbol{h} \\ \boldsymbol{h} \end{bmatrix}. \end{aligned} \tag{5.139}$$

Using the extended vector of optimization variables

$$\hat{\boldsymbol{\omega}}_i = \begin{bmatrix} \alpha_i & \boldsymbol{\gamma}_{1,i}^{\mathrm{T}} & \boldsymbol{\gamma}_{2,i}^{\mathrm{T}} & \cdots & \boldsymbol{\gamma}_{m,i}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \quad \text{with} \quad \boldsymbol{\gamma}_{j,i} \in \mathbb{R}^{\delta} \tag{5.140}$$

and performing the same steps for optimization problem (5.120) results in the linear program

$$\begin{gathered}\min_{\hat{\boldsymbol{\omega}}_i} \boldsymbol{c}^{\mathrm{T}} \hat{\boldsymbol{\omega}}_i \\ \text{s.t. } \boldsymbol{D}_i \hat{\boldsymbol{\omega}}_i \leq \boldsymbol{d}_i\end{gathered} \tag{5.141}$$

with

$$\boldsymbol{c}^{\mathrm{T}} = \begin{bmatrix} 0 & 1 & 1 & \cdots & 1 \end{bmatrix} \tag{5.142}$$

$$\boldsymbol{D}_i = \begin{bmatrix} 1 & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ -1 & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ -\underline{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,1} \tilde{f}_1 & -\boldsymbol{I}_\delta & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \underline{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,1} \tilde{f}_1 & -\boldsymbol{I}_\delta & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ -\underline{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,2} \tilde{f}_2 & \boldsymbol{0} & -\boldsymbol{I}_\delta & \cdots & \boldsymbol{0} \\ \underline{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,2} \tilde{f}_2 & \boldsymbol{0} & -\boldsymbol{I}_\delta & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\underline{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,m} \tilde{f}_m & \boldsymbol{0} & \boldsymbol{0} & \cdots & -\boldsymbol{I}_\delta \\ \underline{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,m} \tilde{f}_m & \boldsymbol{0} & \boldsymbol{0} & \cdots & -\boldsymbol{I}_\delta \end{bmatrix} \tag{5.143}$$

$$\begin{aligned} \boldsymbol{d}_i = [\bar{\alpha}_i \quad -\underline{\alpha}_i \quad -(\bar{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,1} \tilde{f}_1)^{\mathrm{T}} \quad (\bar{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,1} \tilde{f}_1)^{\mathrm{T}} \quad -(\bar{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,2} \tilde{f}_2)^{\mathrm{T}} \quad (\bar{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,2} \tilde{f}_2)^{\mathrm{T}} \\ \cdots \quad -(\bar{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,m} \tilde{f}_m)^{\mathrm{T}} \quad (\bar{\boldsymbol{h}}_i \hat{\boldsymbol{b}}_{f,m} \tilde{f}_m)^{\mathrm{T}}]^{\mathrm{T}} \end{aligned} \tag{5.144}$$

$$\underline{\boldsymbol{h}}_i = \begin{bmatrix} \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{A}}^{\delta-1} \\ \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{A}}^{\delta-2} \\ \vdots \\ \hat{\boldsymbol{m}}_i^{\mathrm{T}} \end{bmatrix} \qquad \bar{\boldsymbol{h}}_i = \begin{bmatrix} \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{A}}^{\delta} \\ \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{A}}^{\delta-1} \\ \vdots \\ \hat{\boldsymbol{m}}_i^{\mathrm{T}} \hat{\boldsymbol{A}} \end{bmatrix} \tag{5.145}$$

Remark 5.14. It would be possible to extend (5.112) even further by including the estimates $\bar{\boldsymbol{\sigma}}_{k+\delta-1}, \cdots, \bar{\boldsymbol{\sigma}}_{k+1}$ of $\boldsymbol{\sigma}_{k+\delta-1}, \cdots, \boldsymbol{\sigma}_{k+1}$. A general reaching law of this class is

$$\boldsymbol{\sigma}_{k+\delta+1} = \boldsymbol{\Gamma}_\delta \bar{\boldsymbol{\sigma}}_{k+\delta} + \boldsymbol{\Gamma}_{\delta-1} \bar{\boldsymbol{\sigma}}_{k+\delta-1} + \cdots + \boldsymbol{\Gamma}_0 \boldsymbol{\sigma}_k - \boldsymbol{S}\,\mathrm{sign}\,(\bar{\boldsymbol{\sigma}}_{k+\delta}) - \hat{\boldsymbol{p}} + \boldsymbol{p}_k \tag{5.146}$$

with appropriately chosen diagonal matrices $\boldsymbol{\Gamma}_\delta, \cdots, \boldsymbol{\Gamma}_0, \boldsymbol{S}$. The increased number of optimization variables will lead to an even smaller perturbation acting on $\boldsymbol{\sigma}_{k+\delta+1}$. This method, however, is not reasonable for two reasons. First, simulation studies have shown that, in some scenarios, the quasi-sliding mode band is not smaller even if the perturbation acting on $\boldsymbol{\sigma}_{k+\delta+1}$ was reduced and the elements in $\boldsymbol{S}$ therefore chosen smaller. Second, in the nominal case, i.e. $\boldsymbol{f}_k = \boldsymbol{0}$ and $\boldsymbol{S} = \boldsymbol{0}$, a reaching law should ensure that once the i^{th} element of the sliding variable approaches zero, all future values of this element are also zero. This can only be achieved by (5.146) with $\boldsymbol{\Gamma}_{\delta-1} = \boldsymbol{\Gamma}_{\delta-2} = \cdots = \boldsymbol{\Gamma}_0 = \boldsymbol{0}$, which results in (5.112).
The switching reaching law proposed in section 5.2.1 does not satisfy the second property, which is an additional drawback of the switching reaching law.

Example 5.15. The predictive switching based controller is now applied to the networked control system described in example 5.9.
Using (5.110) gives

$$\bar{\sigma}_{k+\delta} = \begin{bmatrix} 107.793 & 15.777 & 1 & 1.05 & 1.101 & 1.154 & 1.208 \end{bmatrix} \boldsymbol{\xi}_k + 5.514. \tag{5.147}$$

Solving the optimization problem (5.120) with $\underline{\alpha} = 0$ and $\bar{\alpha} = 0.95$ and evaluating (5.121) results in

$$\alpha = 0.95 \qquad \text{and} \qquad \rho > 3.079. \tag{5.148}$$

Consequently, a valid setting is

$$\rho = 3.082 \tag{5.149}$$

which yields the quasi-sliding mode band given by (5.122) as

$$\Delta = 33.273 \tag{5.150}$$

which is significantly smaller than the one ensured by the switching reaching law (see (5.75)) and just slightly larger than for the nonswitching reaching law (see (5.108)).

The simulation results of the sliding variable σ_k with initial condition (5.76) are shown in figure 5.9. As this figure illustrates, the sliding variable starts to converge monotonically after δ steps until it enters the quasi-sliding mode band Δ which it never leaves. Comparing the results in figure 5.9 and the ones obtained using the switching reaching law in figure 5.4 clearly shows the enhanced behavior in the reaching phase.

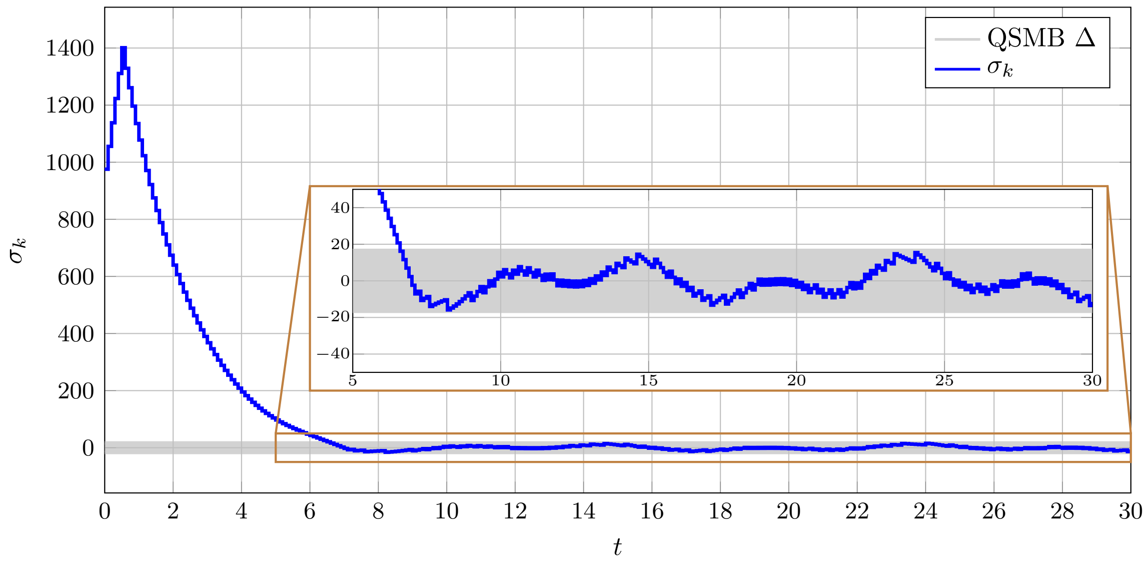


Figure 5.9: Example 5.15: Sliding variable σ_k using the nonswitching reaching law (5.112).

The simulation results for the plant states $\boldsymbol{x}_k$ and the actuating signal u_k are shown in figure 5.10. These results also show the enhanced behavior in the reaching phase. The control signal in particular does not show these extremely high values as visible in the simulation results of the control signal in figure 5.6.

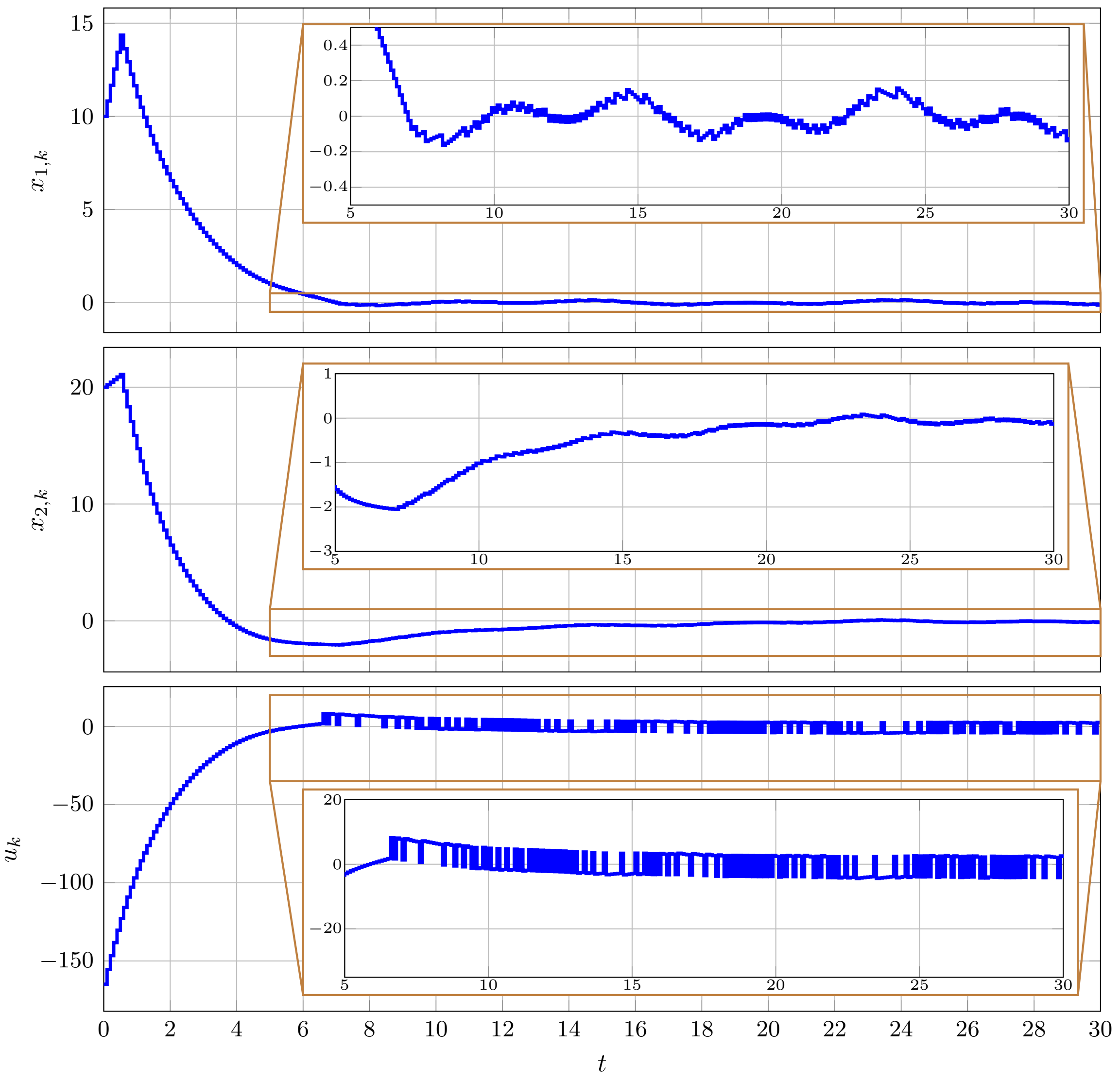


Figure 5.10: Example 5.15: Plant states $\boldsymbol{x}_k$ and input signal u_k using the nonswitching reaching law (5.112).

5.3 Comparison of the Reaching Laws

In sections 5.2.1–5.2.3 three reaching laws for buffered networked systems were proposed, their properties investigated and the parameter choices discussed. This section is dedicated to comparing the presented algorithms.

5.3.1 Comparison of the Simulation Results

For comparison reasons, the simulation results shown in examples 5.9, 5.11 and 5.15 are summarized in this section. Figure 5.11 shows the simulation results of the sliding variable for the networked control system described in example 5.9 with the three different reaching laws applied. To compare the achieved accuracy, a zoomed plot showing the behavior within the quasi-sliding mode band is included. The widths of the quasi-sliding mode bands are shown by the background in the same color as the reaching law ensuring the respective band. This figure shows that the highest accuracy is achieved by the nonswitching reaching law. However, the accuracy achieved with the predictive switching reaching law is just slightly worse. The largest quasi-sliding mode band and therefore the poorest accuracy are reached when the switching reaching law is applied. The simulation results for the states

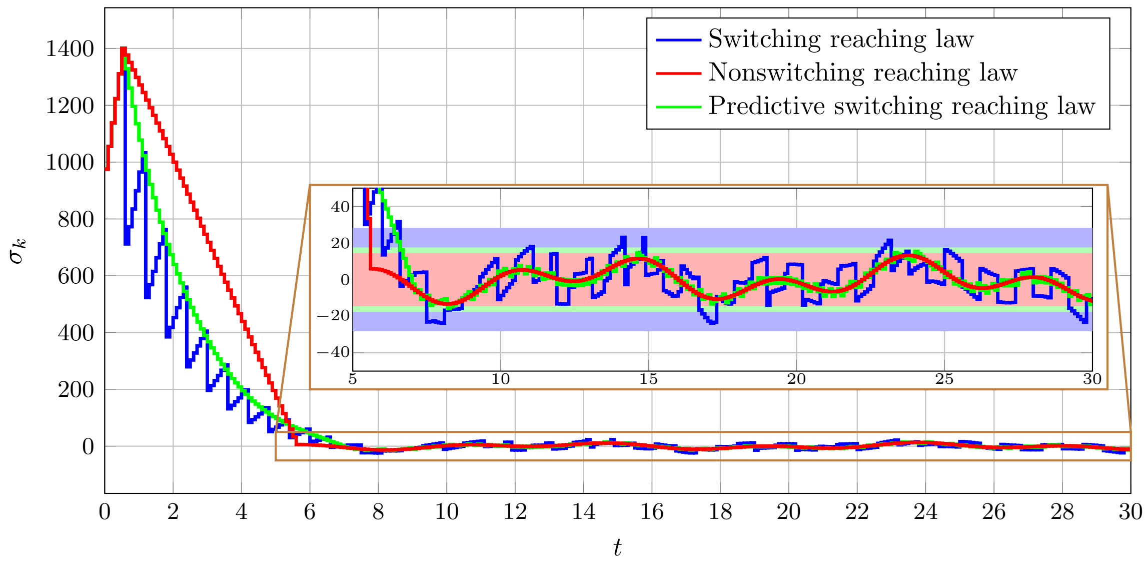


Figure 5.11: Comparison of sliding variables σ_k obtained with the three proposed reaching laws.

$\boldsymbol{x}_k$ and the control signal u_k obtained with the three proposed reaching laws are shown in figure 5.12. This figure also reflects the different achieved accuracies obtained with the proposed reaching laws.

In order to quantify the effectiveness of the proposed reaching laws for real world applications, the robustness of the reaching laws with respect to first order actuator dynamics is investigated by means of numerical simulations.

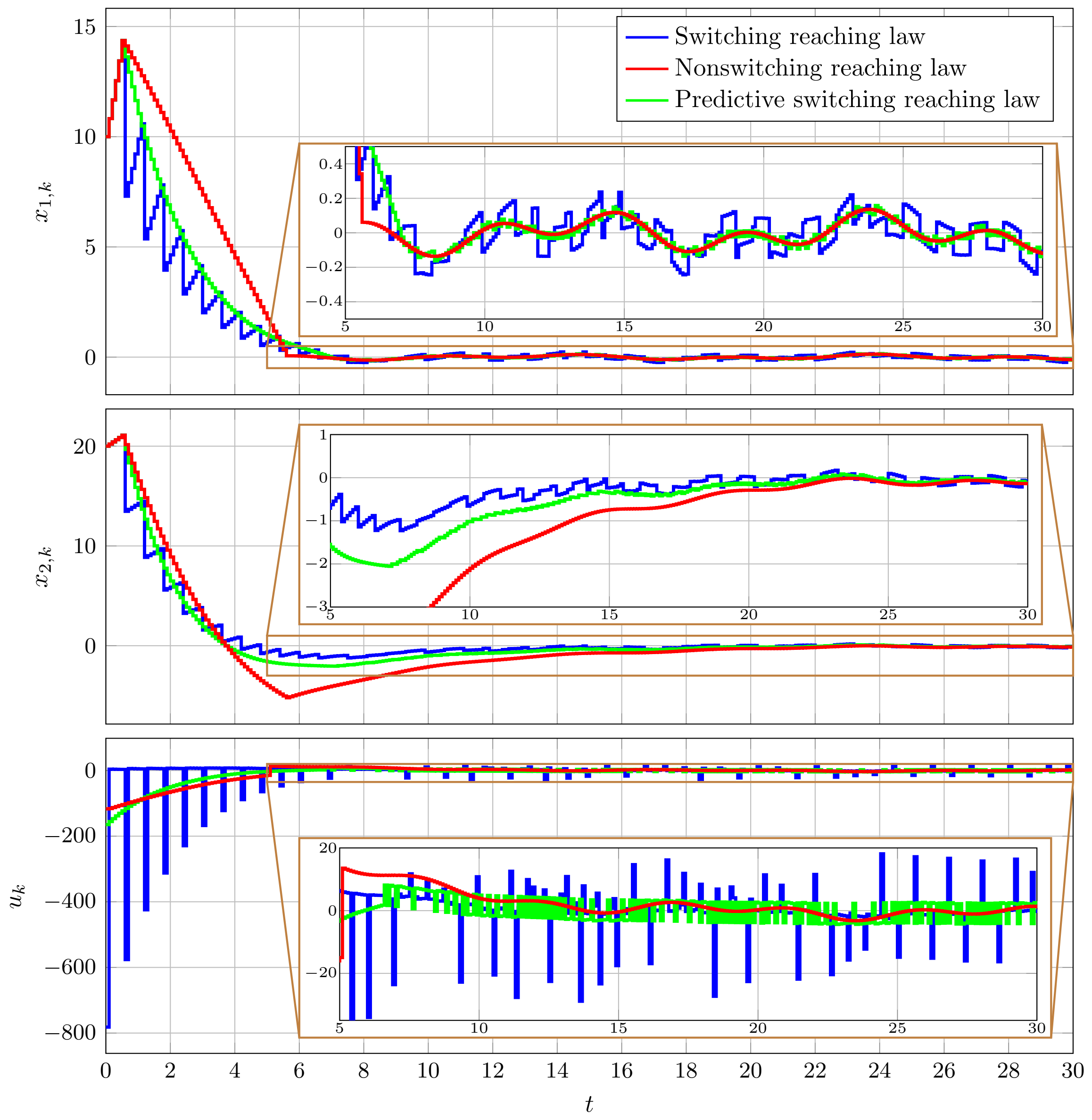


Figure 5.12: Comparison of the plant states $\boldsymbol{x}_k$ and the control signal u_k obtained with the three proposed reaching laws.

5.3.2 Robustness with respect to First Order Actuator Dynamics

In this section, a physically motivated simulation example is shown and the robustness of the presented algorithms with respect to first order actuator dynamics is analyzed. Consider the mechanical system depicted in figure 5.13. The setup consists of a mass m attached to a spring (with linear spring characteristics) through a pulley using a nylon cord. Friction in the pulley is assumed to be viscous (proportional to the velocity). The other side of the spring is connected to a wheel which can be actuated using a speed controlled electric motor. Defining the state vector

$$\boldsymbol{x} = \begin{bmatrix} y & \frac{\mathrm{d}y}{\mathrm{d}t} & z \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}^{\mathrm{T}} \tag{5.151}$$

results in the mathematical model

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \begin{bmatrix} 0 & 1 & 0 \\ -\frac{c}{m} & \frac{-v}{m} & \frac{c}{m} \\ 0 & 0 & 0 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ 0 \\ b \end{bmatrix} (u^* + f) = \boldsymbol{A}_c \boldsymbol{x} + \boldsymbol{b}_c (u^* + f) \tag{5.152}$$

with the known parameters mass $m = 0.18\,\mathrm{kg}$, spring constant $c = 3.840\,\mathrm{N\,m^{-1}}$, friction coefficient $v = 0.042\,\mathrm{kg\,s^{-1}}$ and input gain $b = 0.086\,\mathrm{m\,s^{-1}\,V^{-1}}$. The sampling time is chosen

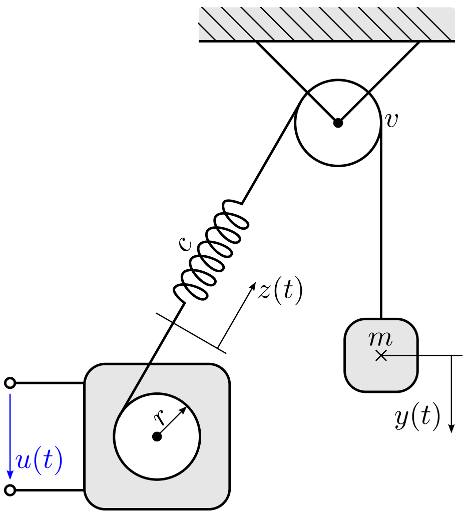


Figure 5.13: Mechanical scheme of the mass spring system.

as $T = 0.02s$ and the round trip time is bounded by

$$\tau_k < 0.2s = 10T. \tag{5.153}$$

The buffer is implemented in such a way that a constant round trip time of $10T$ is achieved, which leads to $\delta = 10$. Discretizing plant (5.152) gives

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} 0.996 & 0.02 & 0.004 \\ -0.425 & 0.991 & 0.425 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} 0 \\ 0 \\ 0.002 \end{bmatrix} (u_{k-10} + f_k). \tag{5.154}$$

Using the lifted state vector

$$\boldsymbol{\xi}_k = \begin{bmatrix} \boldsymbol{x}_k^{\mathrm{T}} & u_{k-1} & u_{k-2} & \cdots & u_{k-10} \end{bmatrix}^{\mathrm{T}} \tag{5.155}$$

results in the lifted model as in (3.26). Applying the methods described in section 4.2 and theorem 5.2, the sliding variable is designed by placing the two poles at $\lambda_1 = \lambda_2 = e^{-10T} = 0.819$, which results in

$$\hat{\boldsymbol{m}}^{\mathrm{T}} = \begin{bmatrix} 1763.556 & 445.886 & 484.664 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{5.156}$$

The perturbation f_k was defined as

$$f_k = \hat{f} + \frac{\tilde{f}}{2} \sin(kT + 5) + \frac{\tilde{f}}{2} \sin\left(\frac{1}{\pi}(kT + 5)\right) \tag{5.157}$$

with

$$\hat{f} = \frac{1}{30}, \qquad \tilde{f} = \frac{2}{30}. \tag{5.158}$$

Applying these bounds to (5.42) and (5.43), the mean $\hat{p}$ and magnitude $\tilde{p}$ of perturbation p_k are given by

$$\hat{p} = 1.079, \qquad \tilde{p} = 2.158. \tag{5.159}$$

The initial conditions of the plant states are set to

$$\boldsymbol{x}_0 = \begin{bmatrix} 0.1 & 0 & 0.1 \end{bmatrix}^{\mathrm{T}}. \tag{5.160}$$

These initial conditions are also used in the estimation process for the nonswitching reaching law. Remember that this estimation will be evaluated in the implementation of the simulation once the first measurement sample is received at the controller. Three control laws are designed based on the reaching laws proposed in sections 5.2.1–5.2.3. The boundaries of α for the predictive switching reaching law were set to

$$\underline{\alpha} = 0, \qquad \bar{\alpha} = 0.98. \tag{5.161}$$

In table 5.1, the chosen parameter values for the different reaching laws are summarized. Simulation results for the sliding variable σ_k obtained with the three reaching laws are shown in figure 5.14. These results show the already discussed properties of these reaching laws. The simulation results for the three states $\boldsymbol{x}_k$ and the control signal u_k are shown in figure 5.15.

In order to investigate the sensitivity of the resulting control algorithms with respect to first order actuator dynamics, consider the first order actuator dynamics

$$\frac{\mathrm{d}\zeta}{\mathrm{d}t} = -\omega\zeta + \omega u \tag{5.162}$$

Table 5.1: Parameter settings for the three reaching laws.

Switching	$\alpha = 0.95$	$\rho = 2.16$	$\Delta = 8.636$
Nonswitching	$\hat{\sigma}_0 = 221.751$	$k^* = 150$	$\Delta = 4.316$
Predictive switching	$\alpha = 0.98$	$\rho = 0.369$	$\Delta = 5.053$

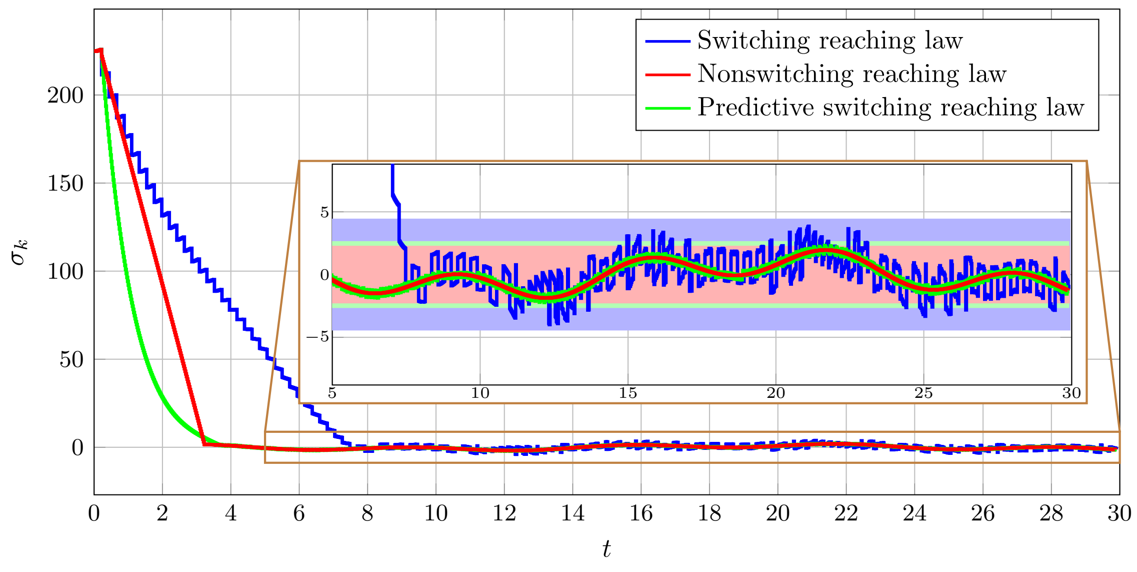


Figure 5.14: Sliding variable σ_k obtained with the three proposed reaching laws.

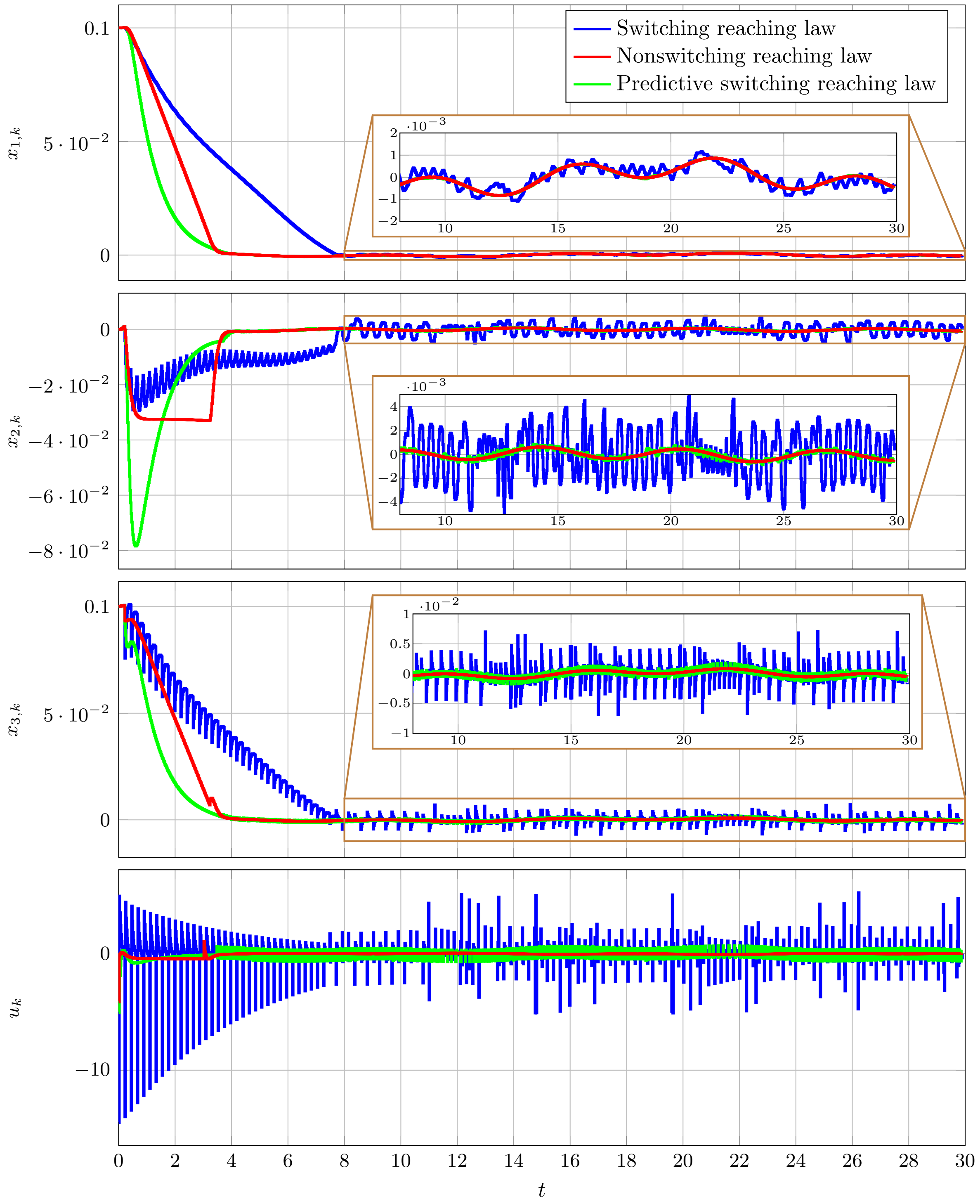


Figure 5.15: Plant states $\boldsymbol{x}_k$ and the control signal u_k obtained with the three proposed reaching laws.

with $\omega > 0$. Combining this actuator dynamics with the nominal plant, i.e. (5.152) with $f = 0$, gives

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\boldsymbol{x}\\ \zeta\end{bmatrix} = \begin{bmatrix}\boldsymbol{A}_c & \boldsymbol{b}_c\\ \boldsymbol{0} & -\omega\end{bmatrix}\begin{bmatrix}\boldsymbol{x}\\ \zeta\end{bmatrix} + \begin{bmatrix}\boldsymbol{0}\\ \omega\end{bmatrix}u^* = \tilde{\boldsymbol{A}}_c\begin{bmatrix}\boldsymbol{x}\\ \zeta\end{bmatrix} + \tilde{\boldsymbol{b}}_c u^*. \tag{5.163}$$

Using this extended system to construct the lifted model with the extended lifted states

$$\tilde{\boldsymbol{\xi}}_k = \begin{bmatrix}\boldsymbol{x}_k^{\mathrm{T}} & \zeta_k & u_{k-1} & u_{k-2} & \cdots & u_{k-10}\end{bmatrix}^{\mathrm{T}} \tag{5.164}$$

results in the extended lifted model

$$\tilde{\boldsymbol{\xi}}_{k+1} = \tilde{\boldsymbol{A}}\tilde{\boldsymbol{\xi}}_k + \tilde{\boldsymbol{b}}u_k \tag{5.165}$$

with

$$\tilde{\boldsymbol{A}} = \begin{bmatrix}\bar{\boldsymbol{A}} & \boldsymbol{0}_{4\times 9} & \bar{\boldsymbol{b}}\\ \boldsymbol{0}_{1\times 4} & \boldsymbol{0} & 0\\ \boldsymbol{0}_{9\times 4} & \boldsymbol{I}_9 & \boldsymbol{0}\end{bmatrix}, \quad \tilde{\boldsymbol{b}} = \begin{bmatrix}\boldsymbol{0}_{4\times 1}\\ 1\\ \boldsymbol{0}_{9\times 1}\end{bmatrix}, \quad \bar{\boldsymbol{A}} = e^{\tilde{\boldsymbol{A}}_c T}, \quad \bar{\boldsymbol{b}} = \int_0^T e^{\tilde{\boldsymbol{A}}_c s}\tilde{\boldsymbol{b}}_c ds. \tag{5.166}$$

Using the singular transformation

$$\boldsymbol{\xi}_k = \boldsymbol{R}\tilde{\boldsymbol{\xi}}_k \tag{5.167}$$

with

$$\boldsymbol{R} = \begin{bmatrix}\boldsymbol{I}_3 & \boldsymbol{0}_{3\times 1} & \boldsymbol{0}_{3\times 10}\\ \boldsymbol{0}_{10\times 3} & \boldsymbol{0}_{10\times 1} & \boldsymbol{I}_{10}\end{bmatrix}, \tag{5.168}$$

the control law in the nominal case (i.e. $f_k = 0$, $\forall k$) for the switching reaching law is given by

$$u_k = (\alpha\hat{\boldsymbol{m}}^{\mathrm{T}} - \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta+1})\boldsymbol{\xi}_k = (\alpha\hat{\boldsymbol{m}}^{\mathrm{T}} - \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta+1})\boldsymbol{R}\tilde{\boldsymbol{\xi}}_k, \tag{5.169}$$

for the nonswitching reaching law

$$u_k = -\hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta+1}\boldsymbol{\xi}_k = -\hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta+1}\boldsymbol{R}\tilde{\boldsymbol{\xi}}_k \tag{5.170}$$

and for the predictive switching law

$$u_k = (\alpha\hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta} - \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta+1})\boldsymbol{\xi}_k = (\alpha\hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta} - \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta+1})\boldsymbol{R}\tilde{\boldsymbol{\xi}}_k. \tag{5.171}$$

Applying (5.169)–(5.171) to (5.165) results in the following closed loop systems:

$$\tilde{\boldsymbol{\xi}}_{k+1} = \left[\tilde{\boldsymbol{A}} + \tilde{\boldsymbol{b}}(\alpha\hat{\boldsymbol{m}}^{\mathrm{T}} - \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta+1})\boldsymbol{R}\right]\tilde{\boldsymbol{\xi}}_k \quad \text{switching reaching law,} \tag{5.172}$$

$$\tilde{\boldsymbol{\xi}}_{k+1} = \left[\tilde{\boldsymbol{A}} - \tilde{\boldsymbol{b}}\hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta+1}\boldsymbol{R}\right]\tilde{\boldsymbol{\xi}}_k \quad \text{nonswitching reaching law,} \tag{5.173}$$

$$\tilde{\boldsymbol{\xi}}_{k+1} = \left[\tilde{\boldsymbol{A}} + \tilde{\boldsymbol{b}}(\alpha\hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta} - \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\delta+1})\boldsymbol{R}\right]\tilde{\boldsymbol{\xi}}_k \quad \text{predictive switching reaching law.} \tag{5.174}$$

The robustness with respect to first order actuator dynamics is now investigated by finding the lowest value of ω for each of these closed loop systems while ensuring that all eigenvalues of the closed loop system matrix have a magnitude smaller than one.
This gives the minimum admissible ω for each reaching law which guarantees asymptotic

Table 5.2: Minimum admissible ω for the proposed reaching laws.

Switching reaching law	10402.883
Nonswitching reaching law	637.104
Predictive switching reaching law	3.868

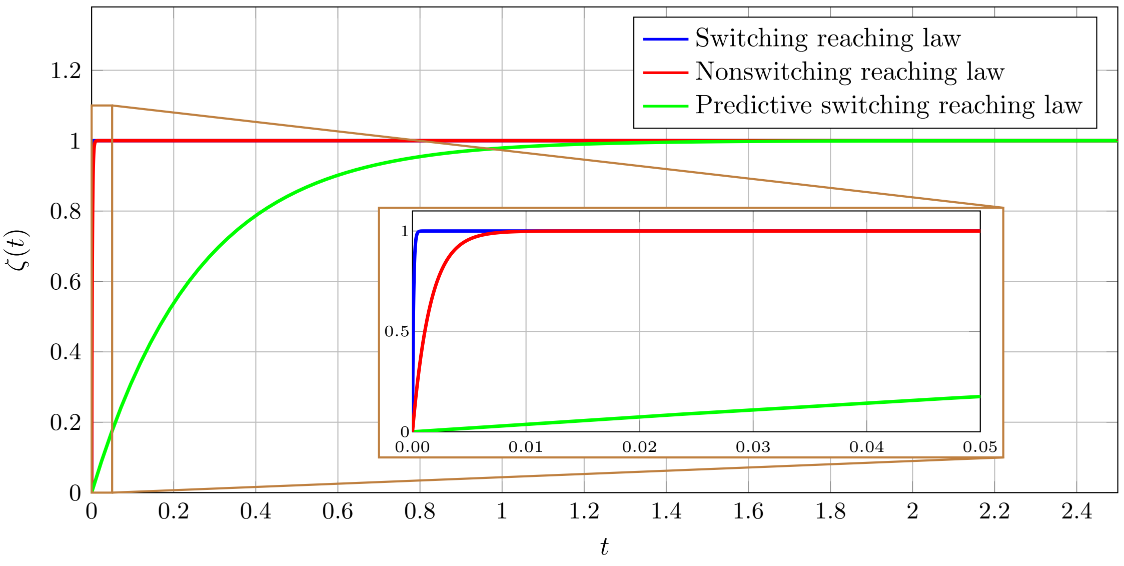


Figure 5.16: Step response of the actuator dynamics with the minimum admissible ω for each of the proposed reaching laws.

stability of the nominal closed loop system. The minimum admissible ω for the scenario and reaching laws considered in this section are summarized in table 5.2. Additionally, the step responses of the actuators with this minimum admissible ω are given in figure 5.16.

As this figure and the values in table 5.2 point out, the minimum admissible ω is extremely high for the switching reaching law and for the nonswitching reaching law. Comparing the step responses of these two actuators with the sampling time $T = 0.02$ clarifies that assuming this fast actuator response for real world scenarios is not realistic. The minimum admissible ω for the predictive switching reaching law is much lower and comparing the step response with the sampling time T shows that the actuator dynamics in a practical application will be faster. Otherwise, the actuator dynamics have to be considered in the controller design.

To show the effect of the first order actuator dynamics, the simulation was extended by a model of the actuator which saturates the input to

$$-10\,\mathrm{V} \leq u_k \leq 10\,\mathrm{V} \tag{5.175}$$

and feeds this saturated signal to the first order actuator dynamics (5.162) with $\omega = 300$. The value ω was chosen in such a way that three times the time constant equals half the sampling time. Therefore, the step response of the actuator reaches 95% of its final value in half the sampling time.

The saturation is also considered in the controller implementation to saturate the output signal and the corresponding elements in the lifted state vector $\boldsymbol{\xi}_k$. Simulation results for the sliding variable obtained with the three proposed reaching laws are shown in figure 5.17. This figure shows no unstable behavior of the switching and the nonswitching reaching law. This is achieved by the saturation of the actuator. As the plant is asymptotically stable and the input is bounded, it is clear that the states and therefore the sliding variable remain bounded. However, the plots for the switching and nonswitching reaching law show a significant loss of accuracy. This is caused by the nominal closed loop system being unstable without saturation. Only the plot of the predictive switching reaching law shows just a slight negative influence by the actuator dynamics.

The simulation results of the states $\boldsymbol{x}_k$ and u_k depicted in figure 5.18 demonstrate the loss of accuracy for the switching and the nonswitching reaching law more profoundly. The control signal in particular shows high-frequency switching between the limits of the control signal. However, the results obtained with the predictive switching reaching law show a desirable behavior. The state variables as well as the control signal are just slightly influenced by the actuator dynamics.

5.4 Conclusions

In this chapter, the matching condition for buffered networked systems was analyzed with the result that it is satisfied for the discrete time model but not for the lifted model. Therefore, a disturbance insensitive motion is only possible in the subspace $\boldsymbol{x}_k$ of the lifted

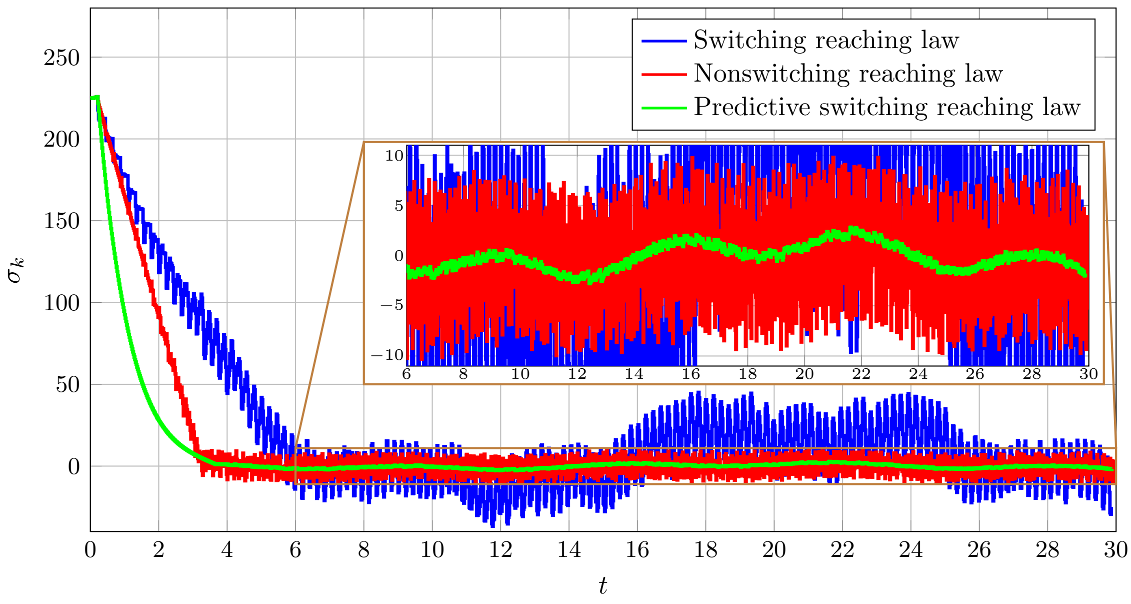


Figure 5.17: Sliding variables σ_k obtained with the three proposed reaching laws and the plant extended by a model of the actuator.

states $\boldsymbol{\xi}_k$.

As a result, the sliding variable has to be designed as a linear combination of the plant states $\boldsymbol{x}_k$ only and not the full lifted states $\boldsymbol{\xi}_k$. Due to this specific structure, the sliding variable is of relative degree δ. Since reaching laws for higher relative degree in literature typically depend on future values of the sliding variable, which are not available for buffered networked systems, it is necessary to design reaching laws specifically for this case.

In this chapter, three reaching laws (switching reaching law, nonswitching reaching law and predictive switching reaching law) are proposed. The switching reaching law is characterized by an independent convergence of $\delta + 1$ consecutive elements of the sliding variable, possibly leading to high-frequency oscillations in the reaching phase, which is typically undesired in practical applications. The nonswitching reaching law makes use of a desired trajectory for the sliding variable in such a way that the actual sliding variable converges monotonically to the quasi-sliding mode band. With the predictive switching reaching law, the disadvantage of high-frequency oscillations in the reaching phase of the switching reaching law is eliminated. Additionally, the parameters of the predictive switching reaching law are chosen by optimizing the width of the quasi-sliding mode band.

Nevertheless, the smallest quasi-sliding mode band is achieved by the nonswitching reaching law followed by the predictive switching law and the poorest accuracy is achieved by the switching reaching law. However, a physically motivated simulation shows that the switching and the nonswitching reaching laws can only be applied if the parasitic actuator dynamics are extremely fast. In the shown physically motivated simulation example, the required actuator dynamics are considered to be unachievable in real world applications. In contrast to that, the predictive switching reaching law can be applied even if the parasitic actuator

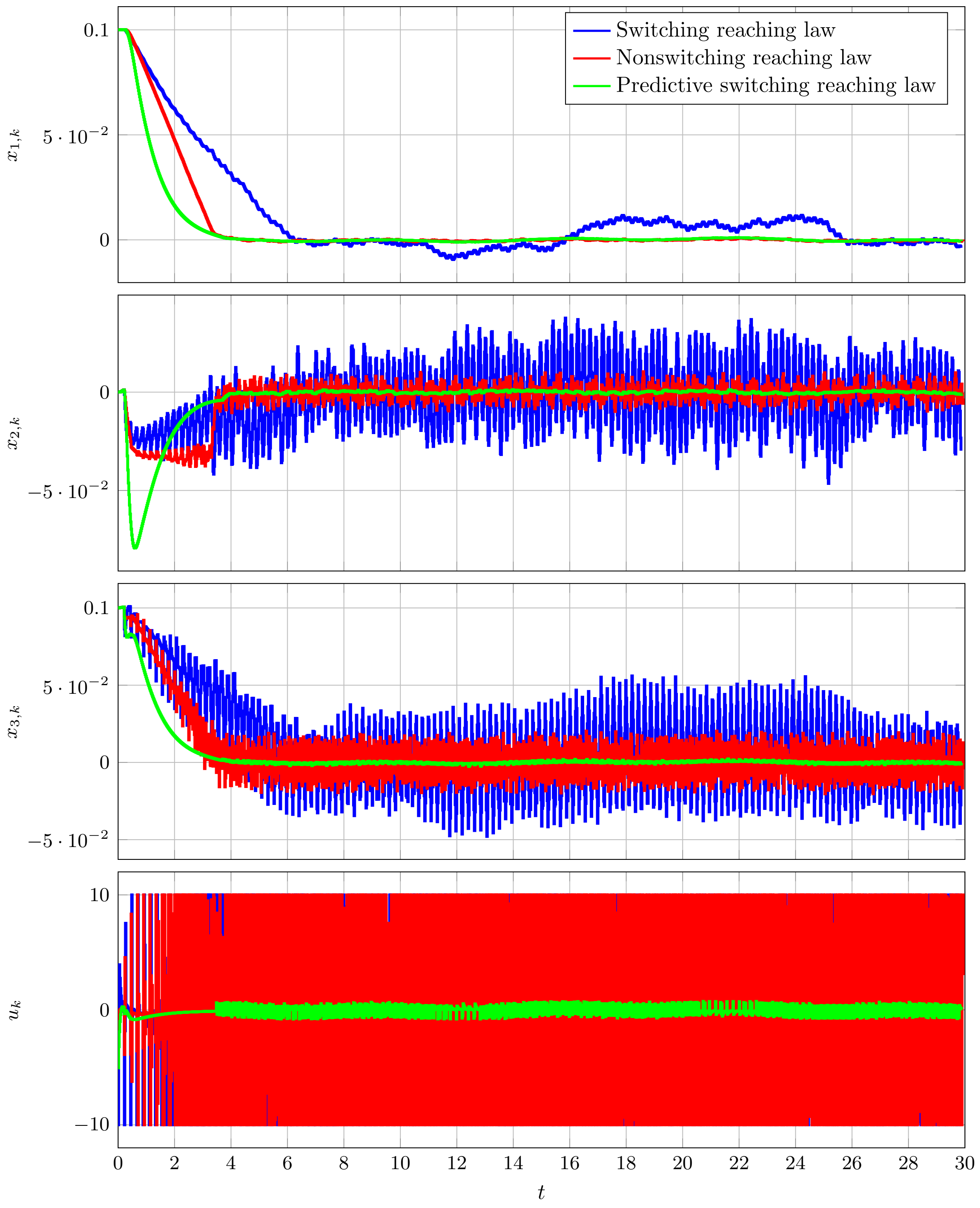


Figure 5.18: Plant states $\boldsymbol{x}_k$ and the control signal u_k obtained with the three proposed reaching laws and the plant extended by a model of the actuator.

dynamics are very slow. This property is especially important for real world applications as parasitic dynamics can never be avoided in such scenarios.

It was also shown that sliding mode control laws based on sliding variables with the proposed structure are suitable for a centralized network topology but not for the spatially distributed network topology. Control laws for spatially distributed networked systems are proposed in the next chapter.

Chapter 6

Spatially Distributed Sliding Mode Control for Buffered Networked Systems

Contents

In chapter 5, three control algorithms for centralized buffered networked systems based on the reaching law approach are proposed. However, some applications trigger the need to develop spatially distributed control algorithms. In these algorithms, the control law for each input channel is implemented in a distinct controller node. This architecture introduces the additional difficulties that the upper limit of the round trip times could be different for each feedback channel and that the controller nodes do not communicate with each other.

In this chapter, a control architecture based on the results published in [20, 21] is proposed which can be implemented in a spatially distributed setup. The proposed approach is shown for single-input systems first and then extended to the multi-input case.

6.1 Integral Sliding Mode Control for Single-Input Networked Systems

In this section, a control scheme for single-input buffered networked control systems as depicted in figure 6.1 with continuous time plant

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{A}_c\boldsymbol{x} + \boldsymbol{b}_c(u^* + f) \tag{6.1}$$

based on integral sliding mode control is presented. The proposed approach makes it possible to cast the problem into a form in which discrete time sliding mode control algorithms designed for systems with relative degree one can be applied. For the single-input case,

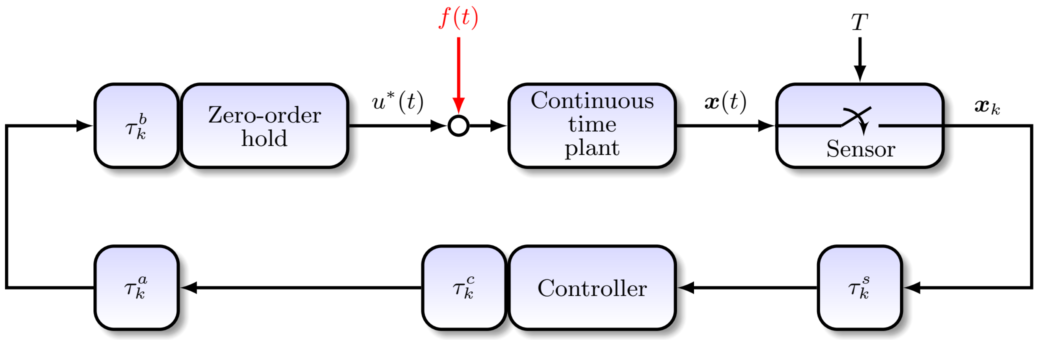


Figure 6.1: Single-input buffered networked control system.

consider the lifted model given in (3.26) with $m = 1$, which results in

$$\boldsymbol{\xi}_{k+1} = \hat{\boldsymbol{A}}\boldsymbol{\xi}_k + \hat{\boldsymbol{b}}u_k + \hat{\boldsymbol{b}}_f f_k \tag{6.2}$$

where

$$\hat{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{0}_{n\times\delta-1} & \boldsymbol{b} \\ \boldsymbol{0}_{1\times n} & \boldsymbol{0} & 0 \\ \boldsymbol{0}_{\delta-1\times n} & \boldsymbol{I}_{\delta-1} & \boldsymbol{0} \end{bmatrix}, \qquad \hat{\boldsymbol{b}} = \begin{bmatrix} \boldsymbol{0}_{n\times 1} \\ 1 \\ \boldsymbol{0}_{\delta-1\times 1} \end{bmatrix}, \qquad \hat{\boldsymbol{b}}_f = \begin{bmatrix} \boldsymbol{b} \\ 0 \\ \boldsymbol{0} \end{bmatrix}. \tag{6.3}$$

The basic concept of integral sliding mode control is to robustify a control loop designed for the nominal case, i.e. $f_k = 0,\ \forall k$. This is achieved by defining the control signal as

$$u_k = u_k^N + u_k^S \tag{6.4}$$

where u_k^N denotes a nominal control signal designed for the nominal case and u_k^S represents the sliding mode part of the control law. The control component u_k^S is designed to compensate for the matched perturbation in order to keep the desired properties achieved by the nominal control law also in the perturbed case.

The nominal control law u_k^N is designed for the nominal lifted model

$$\hat{\boldsymbol{\xi}}_{k+1} = \hat{\boldsymbol{A}}\hat{\boldsymbol{\xi}}_k + \hat{\boldsymbol{b}}u_k^N \tag{6.5}$$

with the nominal lifted state vector

$$\hat{\boldsymbol{\xi}}_k = \begin{bmatrix}\boldsymbol{x}_k^{\mathrm{T}} & u_{k-1}^N & u_{k-2}^N & \cdots & u_{k-\delta}^N\end{bmatrix}^{\mathrm{T}} \tag{6.6}$$

which results from the lifted model (6.2) with (6.4) for $f_k = u_k^S = 0,\ \forall k$. In the single-input case, no specific structure of the nominal control law has to be considered. Hence, classical approaches such as assigning $n+\delta$ eigenvalues of the closed loop system matrix $\hat{\boldsymbol{A}} - \hat{\boldsymbol{b}}\hat{\boldsymbol{k}}^{\mathrm{T}}$ using a linear state control law

$$u_k^N = -\hat{\boldsymbol{k}}^{\mathrm{T}}\hat{\boldsymbol{\xi}}_k \qquad \text{with} \qquad \hat{\boldsymbol{k}} \in \mathbb{R}^{n+\delta} \tag{6.7}$$

can be applied. The eigenvalues have to be chosen in a way to achieve stability and desired performance of the unperturbed networked control system.

To design the sliding mode part of the control law, the discrete time integral sliding variable

$$\sigma_k = \hat{\boldsymbol{m}}^{\mathrm{T}}\boldsymbol{\xi}_k + w_k \tag{6.8}$$

is defined. Considering the forward increment of (6.8) using (6.2) and (6.4) results in

$$\begin{aligned}
\sigma_{k+1} &= \hat{\boldsymbol{m}}^{\mathrm{T}}\boldsymbol{\xi}_{k+1} + w_{k+1} &(6.9)\\
&= \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{A}}\boldsymbol{\xi}_k + \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{b}}u_k + \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{b}}f_k + w_{k+1} &(6.10)\\
&= \hat{\boldsymbol{m}}^{\mathrm{T}}(\hat{\boldsymbol{A}}\boldsymbol{\xi}_k + \hat{\boldsymbol{b}}u_k^N) + \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{b}}u_k^S + \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{b}}_f f_k + w_{k+1}. &(6.11)
\end{aligned}$$

Defining

$$w_{k+1} = -\hat{\boldsymbol{m}}^{\mathrm{T}}(\hat{\boldsymbol{A}}\boldsymbol{\xi}_k + \hat{\boldsymbol{b}}u_k^N) \tag{6.12}$$

with

$$w_0 = -\hat{\boldsymbol{m}}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{x}_0\\ \boldsymbol{0}\end{bmatrix}, \tag{6.13}$$

which contains known quantities only, and applying it to (6.11) yields

$$\sigma_{k+1} = \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{b}}u_k^S + \hat{\boldsymbol{m}}^{\mathrm{T}}\hat{\boldsymbol{b}}_f f_k. \tag{6.14}$$

In addition, the initial condition (6.13) ensures $\sigma_0 = 0$ since the past input samples are assumed to be zero, i.e.

$$u_{-1} = u_{-2} = \cdots = u_{-\delta} = 0. \tag{6.15}$$

The vector $\hat{\boldsymbol{m}}^{\mathrm{T}}$ can be freely chosen and a reasonable choice is

$$\hat{\boldsymbol{m}}^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{m}^{\mathrm{T}} & 1 & \boldsymbol{0} \end{bmatrix} \qquad \text{with} \qquad \boldsymbol{m}^{\mathrm{T}}\boldsymbol{b} = 1 \quad \boldsymbol{m} \in \mathbb{R}^n \tag{6.16}$$

which applied to (6.14) yields

$$\sigma_{k+1} = u_k^S + f_k. \tag{6.17}$$

Remark 6.1. Assume a sliding mode based control law u_k^S for system (6.17) which is capable of forcing σ_{k+1} to zero, i.e.

$$\sigma_{k+1} = 0 \qquad \Leftrightarrow \qquad u_k^S = -f_k. \tag{6.18}$$

Considering (6.2) in state coordinates, i.e.

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{b}(u_{k-\delta} + f_k), \tag{6.19}$$

together with (6.18) shows that even if $\sigma_{k+1} = 0$, $\forall k$ were achieved, the perturbation

$$f_k - f_{k-\delta} \tag{6.20}$$

would act on (6.19). As a consequence, it is reasonable to not consider perturbations with bounded amplitude but with bounded change rate, i.e.

$$\sup \left| \frac{f_{k+1} - f_k}{T} \right| = L_f. \tag{6.21}$$

In this case, (6.20) is bounded by

$$\sup |f_k - f_{k-\delta}| = \delta T L_f. \tag{6.22}$$

Theorem 6.2. Consider the buffered networked control system depicted in figure 6.1 with plant (6.1) and sampling time T. Also, assume that assumptions 1.2, 1.4, 2.2 and 2.4 are satisfied and the change rate of perturbation f_k is bounded by (6.21). The sliding variable is given by (6.8) with w_{k+1} given by (6.12) and $\hat{\boldsymbol{m}}^{\mathrm{T}}$ given by (6.16). Let the discrete time sliding mode control law be designed as

$$u_k^S = \sigma_k + T\tilde{u}_k^S \tag{6.23}$$

with

$$\begin{aligned} \tilde{u}_k^S &= \frac{1}{T}(\tilde{\ell}_k - 1)\sigma_k + \nu_k \\ \nu_{k+1} &= \nu_k + \ell_k \sigma_k \end{aligned} \tag{6.24}$$

$$\begin{aligned} \tilde{\ell}_k &= q_k^{(1)} + q_k^{(2)} - 1 \\ \ell_k &= \frac{1}{T}\left(\tilde{\ell}_k - q_k^{(1)} q_k^{(2)}\right). \end{aligned} \tag{6.25}$$

The elements $q_k^{(1)}$ and $q_k^{(2)}$ depend on the desired discretization method of the super twisting algorithm and are defined as

$$\begin{aligned} q_k^{(1)} &= 1 + Ts^{(1)}(\sigma_k) \\ q_k^{(2)} &= 1 + Ts^{(2)}(\sigma_k) \end{aligned} \qquad \text{explicit Euler,} \tag{6.26}$$

$$\begin{aligned} q_k^{(1)} &= \begin{cases} e^{s^{(1)}(\sigma_k)T} & \sigma_k \neq 0 \\ 0 & \sigma_k = 0 \end{cases} \\ q_k^{(2)} &= \begin{cases} e^{s^{(2)}(\sigma_k)T} & \sigma_k \neq 0 \\ & \sigma_k = 0 \end{cases} \end{aligned} \qquad \text{matching approach} \tag{6.27}$$

where

$$s^{(1)}(\sigma_k) = p^{(1)} \left|\sigma_k\right|^{-\frac{1}{2}} \qquad s^{(2)}(\sigma_k) = p^{(2)} \left|\sigma_k\right|^{-\frac{1}{2}} . \tag{6.28}$$

If

$$p^{(1)} = \left(-0.75 - \frac{\sqrt{2.15}}{2} i\right) \sqrt{\frac{L_f}{T}} \qquad p^{(2)} = \left(-0.75 + \frac{\sqrt{2.15}}{2} i\right) \sqrt{\frac{L_f}{T}}, \tag{6.29}$$

then the states of the plant are ultimately bounded.

Proof. The forward increment of the sliding variable (6.8) with (6.12) and (6.16) yields (6.17). This is equivalent to the discretized perturbed integrator

$$\sigma_{k+1} = \sigma_k + T\tilde{u}_k^S + T\varphi_k \tag{6.30}$$

$$\varphi_{k+1} = \varphi_k + Tv_k \tag{6.31}$$

for $\varphi_k = \frac{1}{T} f_k$ and using (6.23). The two discrete time super twisting algorithms can then be applied as described in section 4.5. The parameters α and β are chosen using the well-established parameter setting

$$\alpha = 1.5\sqrt{L_\varphi}, \qquad \beta = 1.1 L_\varphi \tag{6.32}$$

with

$$\sup \left| \frac{\varphi_{k+1} - \varphi_k}{T} \right| = L_\varphi . \tag{6.33}$$

The change rate (6.33) can easily be computed from (6.21) using

$$\sup \left| \frac{\varphi_{k+1} - \varphi_k}{T} \right| = \sup \left| \frac{f_{k+1} - f_k}{T^2} \right| = L_\varphi = \frac{L_f}{T} . \tag{6.34}$$

Applying (6.34) to (6.32) and using relations (4.91) yields

$$\alpha = -\left(p^{(1)} + p^{(2)}\right) = 1.5\sqrt{\frac{L_f}{T}}, \qquad \beta = p^{(1)}p^{(2)} = 1.1\frac{L_f}{T} . \tag{6.35}$$

Solving this set of equations for $p^{(1)}$ and $p^{(2)}$ yields the parameter setting (6.29). □

Example 6.3. This integral sliding mode based controller for single-input networked control systems is now applied to the networked control system described in example 5.9. Choosing the eigenvalues of the nominal lifted closed loop dynamic matrix as

$$\boldsymbol{p} = \begin{bmatrix} 0.905 & 0.905 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}} \tag{6.36}$$

leads to the nominal control law (6.7) with

$$\hat{\boldsymbol{k}}^{\mathrm{T}} = \begin{bmatrix} 35.863 & -6.479 & 0.221 & 0.236 & 0.252 & 0.268 & 0.285 \end{bmatrix}. \tag{6.37}$$

One possible choice for $\hat{\boldsymbol{m}}^{\mathrm{T}}$ satisfying (6.16) is

$$\hat{\boldsymbol{m}}^{\mathrm{T}} = \begin{bmatrix} 49.741 & 48.76 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.38}$$

where $\boldsymbol{m}^{\mathrm{T}} = (\boldsymbol{b}^{\mathrm{T}}\boldsymbol{b})^{-1}\boldsymbol{b}^{\mathrm{T}}$ was chosen by computing the Moore-Penrose pseudoinverse of $\boldsymbol{b}$. In a first simulation, only the nominal control law is implemented and the sliding mode part of the control law is set to zero, i.e. $u_k = u_k^N$.
The simulation results of the sliding variable σ_k and the applied perturbation f_k are shown in figure 6.2. Since σ_{k+1} equals f_k for $u_k^S = 0$, the sliding variable σ_k equals the perturbation delayed by one step, i.e. f_{k-1}.

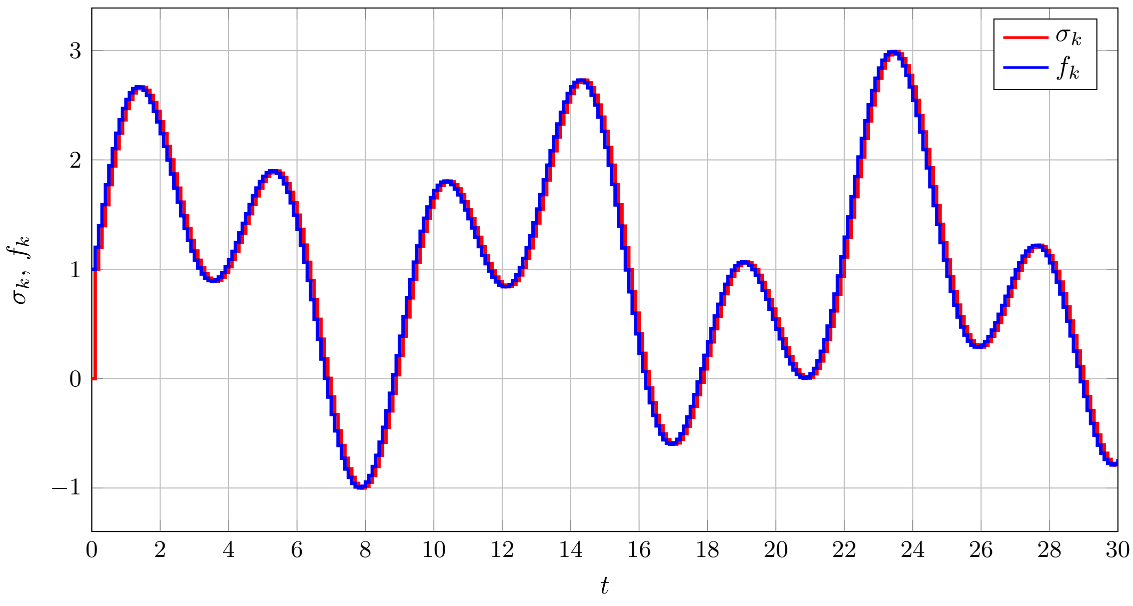


Figure 6.2: Example 6.3: Sliding variable σ_k and perturbation f_k for $u_k^S = 0$.

The design of the sliding mode part of the control law relies on the change rate of the perturbation. As the perturbation is given by

$$f_k = \sin\left(\sqrt{2}kT\right) + \sin\left(\frac{3kT}{5}\right) + 1, \tag{6.39}$$

in this simulation, the exact limit of the change rate is given by

$$L_f = 2.014. \tag{6.40}$$

In practical applications, this value is typically not known exactly. However, an upper bound $\Lambda > \frac{L_f}{T}$ can usually be estimated. This upper bound is then used in (6.29) to obtain the parameters

$$p^{(1)} = \left(-0.75 - \frac{\sqrt{2.15}}{2}i\right)\sqrt{\Lambda}, \qquad p^{(2)} = \left(-0.75 + \frac{\sqrt{2.15}}{2}i\right)\sqrt{\Lambda}. \tag{6.41}$$

In order to compare the accuracies with respect to the sliding variable achieved with the two sliding mode controllers, the value of Λ which leads to the highest asymptotic accuracy of the sliding variable was used for each controller. In order to determine these values, control laws are designed using $\frac{L_f}{10T} \leq \Lambda \leq \frac{10L_f}{T}$ which is obtained by applying this procedure in simulation. Therefore, simulations using Λ in the range of $\frac{L_f}{10T}$ and $\frac{10L_f}{T}$ are performed and the achieved asymptotic accuracy on the sliding variable σ_k is evaluated. Figure 6.3 shows the asymptotic accuracies of the sliding variable σ_k achieved using the two algorithms for increasing values of Λ. Each of the two zoomed plots show the values of Λ leading to the highest accuracy obtained with the corresponding algorithm. The resulting parameters are given in table 6.1.

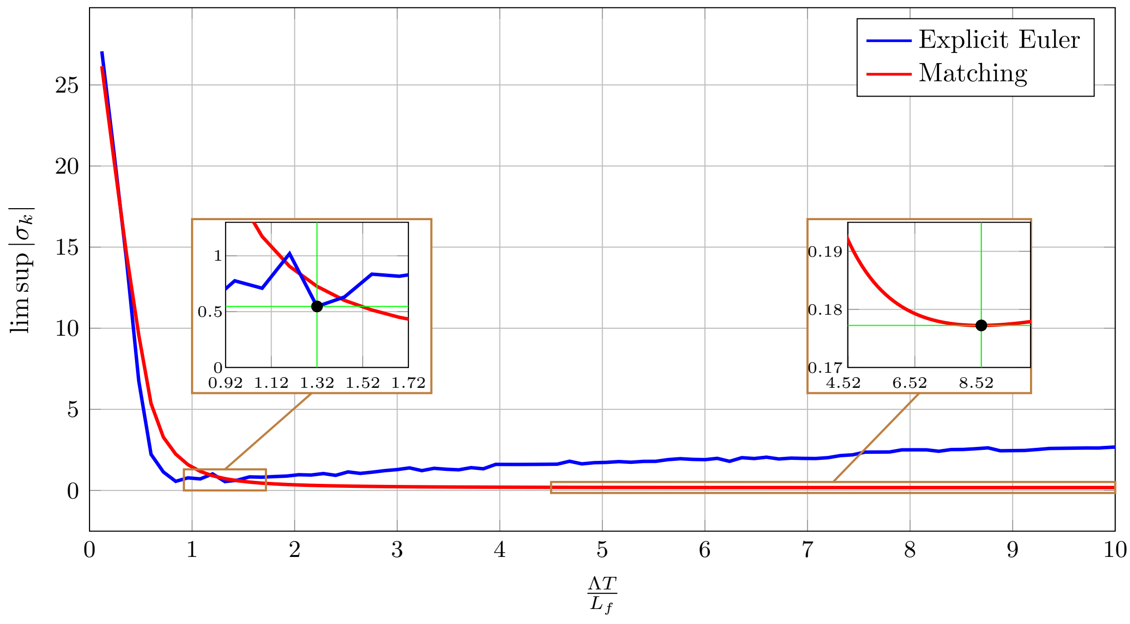


Figure 6.3: Example 6.3: Asymptotic accuracies of the sliding variable using the explicit Euler and the matching discretization method for increasing values of Λ obtained in simulation. The zoomed plots show the values of Λ which lead to the highest accuracy for each of the two algorithms.

Table 6.1: Parameter values used for the comparison of the control laws.

	Explicit Euler	Matching
Λ	26.588	171.611
$p^{(1)}$	$-3.867 - 3.780i$	$-9.825 - 9.604i$
$p^{(2)}$	$-3.867 + 3.780i$	$-9.825 + 9.604i$

The simulation results for the sliding variable σ_k using the two sliding mode controllers with the settings as given in table 6.1 are shown in figure 6.4. As expected, the results obtained with the matching algorithm do not show discretization chattering. In addition, the accuracy achieved with the matching algorithm is slightly better than achieved using the explicit Euler discretization. However, the increase of accuracy is almost not visible in the plant states $\boldsymbol{x}_k$, which are depicted in figure 6.5, but the increase in accuracy achieved by adding the sliding mode part of the control law is clearly visible.

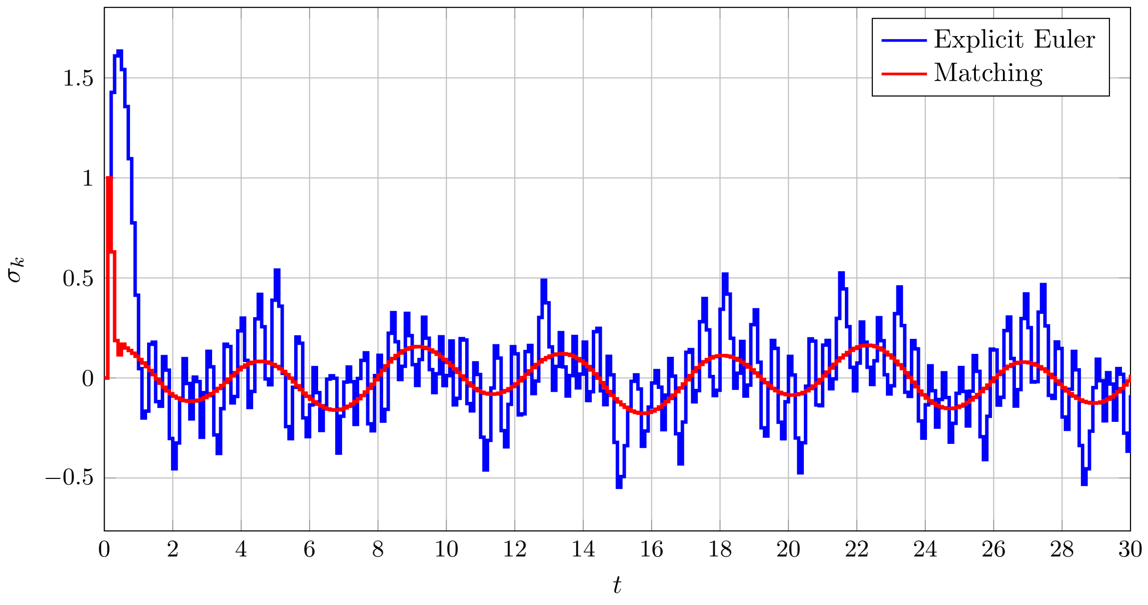


Figure 6.4: Example 6.3: Sliding variable σ_k using the integral sliding mode approach with two different sliding mode controllers.

6.2 Spatially Distributed Super Twisting Control for Multi-Input Networked Control Systems

The previously developed method for single-input systems is extended to the multi-input case in this section. In order to be applicable in a wide range of networked control systems, the method proposed in this section not only aims to robustly stabilize the origin of the plant

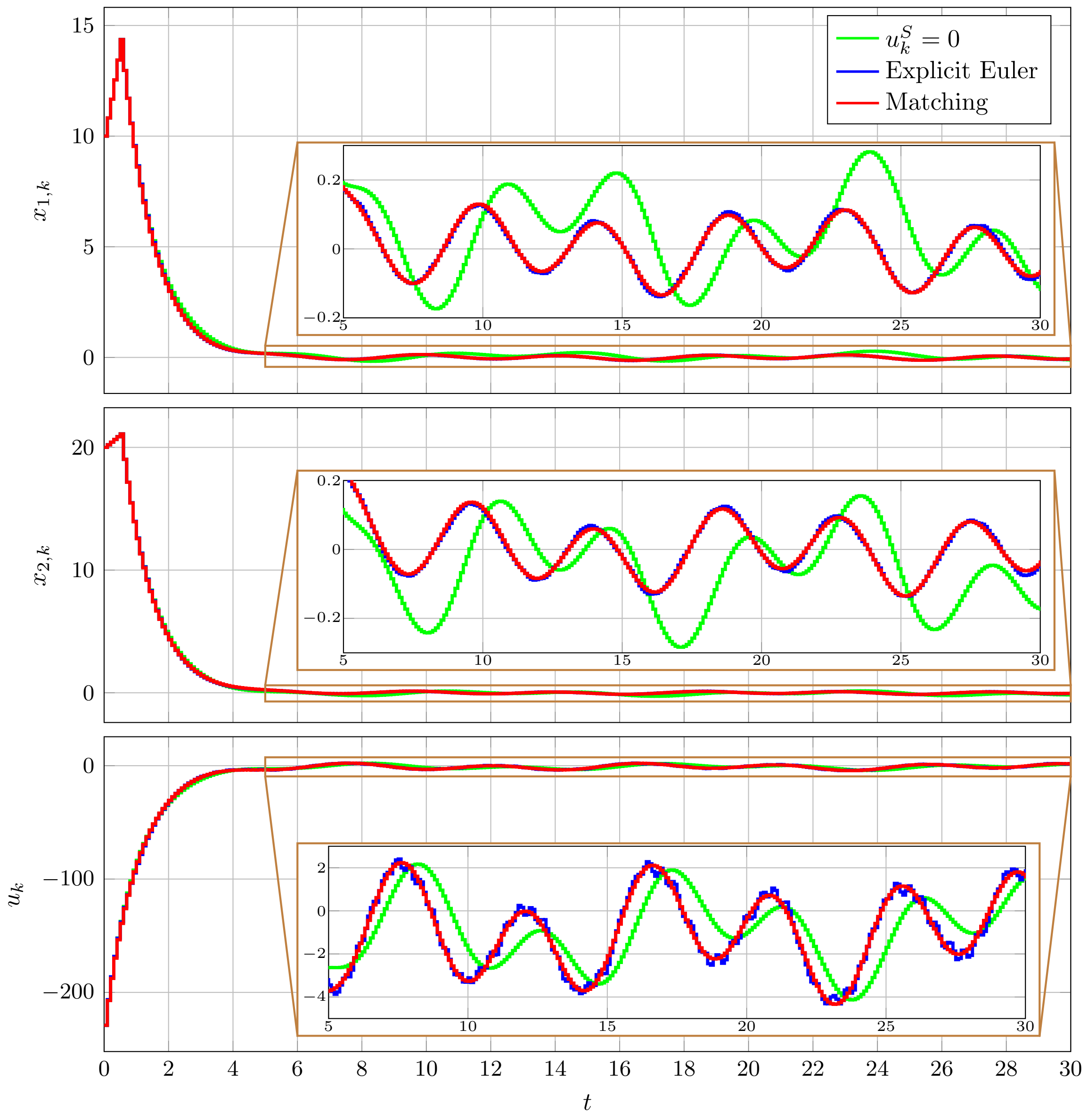


Figure 6.5: Example 6.3: Plant states $\boldsymbol{x}_k$ and the input signal u_k using the integral sliding mode approach with two different sliding mode controllers.

but also to allow the control laws for each input channel to be implemented without the need to share information among the controller nodes. Consider the buffered networked control system for multi-input systems described in chapter 3. The lifted model for multi-input systems is given by

$$\boldsymbol{\xi}_{k+1} = \hat{\boldsymbol{A}}\boldsymbol{\xi}_k + \hat{\boldsymbol{B}}\boldsymbol{u}_k + \hat{\boldsymbol{B}}_f\boldsymbol{f}_k \tag{6.42}$$

with $\hat{\boldsymbol{A}}$, $\hat{\boldsymbol{B}}$ and $\hat{\boldsymbol{B}}_f$ respectively given by (3.27)–(3.29). The control law for the multi-input case again consists of the nominal part $\boldsymbol{u}_k^N$ and the sliding mode part $\boldsymbol{u}_k^S$, i.e.

$$\boldsymbol{u}_k = \boldsymbol{u}_k^N + \boldsymbol{u}_k^S. \tag{6.43}$$

In order to comply with the problem statement, the i$^{\text{th}}$ controller node cannot use the whole state vector $\boldsymbol{\xi}_k$ since this vector includes the controller outputs of all other controller nodes and their history which is not available.

6.2.1 Nominal Control Law

The nominal lifted model

$$\hat{\boldsymbol{\xi}}_{k+1} = \hat{\boldsymbol{A}}\hat{\boldsymbol{\xi}}_k + \hat{\boldsymbol{B}}u_k^N \tag{6.44}$$

with nominal lifted state vector

$$\hat{\boldsymbol{\xi}}_k = \begin{bmatrix}\boldsymbol{x}_k^{\mathrm{T}} & u_{1,k-1}^N & \cdots & u_{1,k-\delta_1}^N & \cdots & u_{m,k-1}^N & u_{m,k-2}^N & \cdots & u_{m,k-\delta_m}^N\end{bmatrix}^{\mathrm{T}} \tag{6.45}$$

results from (6.42) and (6.43) for $\boldsymbol{f}_k = \boldsymbol{u}_k^S = \boldsymbol{0}\ \forall k$. In the following theorem, the nominal control law is designed based on theorem 5 in [22] in such a way that the global asymptotic stability of the nominal closed loop system is ensured. In addition, the control law for each controller node relies on the measurements $\boldsymbol{x}_k$ and locally available information only.

Theorem 6.4. Consider the buffered networked control system depicted in figure 3.5 with plant (1.5) and sampling time T. Also, assume that assumptions 1.2, 1.4 and 2.6 are satisfied. Let the controller be given by (6.43) with the nominal control law

$$\boldsymbol{u}_k^N = -\hat{\boldsymbol{K}}\hat{\boldsymbol{\xi}}_k = -\begin{bmatrix}\boldsymbol{K}_x & \boldsymbol{K}_u\end{bmatrix}\hat{\boldsymbol{\xi}}_k \tag{6.46}$$

with

$$\boldsymbol{K}_x \in \mathbb{R}^{m\times n}, \qquad \boldsymbol{K}_u = \begin{bmatrix}\boldsymbol{k}_1^{\mathrm{T}} & \boldsymbol{0} & \cdots & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{k}_2^{\mathrm{T}} & \cdots & \boldsymbol{0}\\ \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{k}_m^{\mathrm{T}}\end{bmatrix}, \qquad \begin{array}{l}\boldsymbol{k}_i^{\mathrm{T}} \in \mathbb{R}^{1\times\delta_i}\\ i = 1,\dots,m.\end{array} \tag{6.47}$$

If a symmetric positive definite matrix $\boldsymbol{Y} \in \mathbb{R}^{n+\mu\times n+\mu}$ with

$$\mu = \sum_{i=1}^{m}\delta_i, \tag{6.48}$$

matrices

$$\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{Z}_x & \boldsymbol{Z}_u \end{bmatrix}, \tag{6.49}$$

$$\boldsymbol{Z}_x \in \mathbb{R}^{m\times n}, \qquad \boldsymbol{Z}_u = \begin{bmatrix} \boldsymbol{z}_1^{\mathrm{T}} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{z}_2^{\mathrm{T}} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{z}_m^{\mathrm{T}} \end{bmatrix}, \qquad \begin{matrix} \boldsymbol{z}_i^{\mathrm{T}} \in \mathbb{R}^{1\times\delta_i} \\ i = 1,\dots,m \end{matrix} \tag{6.50}$$

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{0} \\ \boldsymbol{X}_2 & \boldsymbol{X}_3 \end{bmatrix}, \tag{6.51}$$

$$\begin{matrix} \boldsymbol{X}_1 \in \mathbb{R}^{n\times n} \\ \boldsymbol{X}_2 \in \mathbb{R}^{\mu\times n} \end{matrix}, \qquad \boldsymbol{X}_3 = \begin{bmatrix} \bar{\boldsymbol{X}}_{3,1} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \bar{\boldsymbol{X}}_{3,2} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \bar{\boldsymbol{X}}_{3,m} \end{bmatrix}, \qquad \begin{matrix} \bar{\boldsymbol{X}}_{3,i} \in \mathbb{R}^{\delta_i\times\delta_i} \\ i = 1,\dots,m. \end{matrix} \tag{6.52}$$

and a scalar

$$0 \le \gamma < 1 \tag{6.53}$$

that satisfy

$$\begin{bmatrix} \boldsymbol{X} + \boldsymbol{X}^{\mathrm{T}} - \boldsymbol{Y} & \boldsymbol{X}^{\mathrm{T}}\hat{\boldsymbol{A}}^{\mathrm{T}} - \boldsymbol{Z}^{\mathrm{T}}\hat{\boldsymbol{B}}^{\mathrm{T}} \\ \hat{\boldsymbol{A}}\boldsymbol{X} - \hat{\boldsymbol{B}}\boldsymbol{Z} & (1-\gamma)\boldsymbol{Y} \end{bmatrix} \succ 0, \tag{6.54}$$

exist then the nominal lifted closed loop system is globally asymptotically stable with

$$\boldsymbol{K}_u = \boldsymbol{Z}_u\boldsymbol{X}_3^{-1} \qquad \text{and} \qquad \boldsymbol{K}_x = (\boldsymbol{Z}_x - \boldsymbol{K}_u\boldsymbol{X}_2)\boldsymbol{X}_1^{-1}. \tag{6.55}$$

In addition, only measurements $\boldsymbol{x}_k$ and local information, i.e. the history of the own control signals $u_{i,k}$, are used in the i$^{\text{th}}$ controller node.

Proof. Consider the product

$$\hat{\boldsymbol{K}}\boldsymbol{X} = \begin{bmatrix} \boldsymbol{K}_x\boldsymbol{X}_1 + \boldsymbol{K}_u\boldsymbol{X}_2 & \boldsymbol{K}_u\boldsymbol{X}_3 \end{bmatrix} \tag{6.56}$$

which results from (6.46) and (6.52). Applying (6.55) gives

$$\hat{\boldsymbol{K}}\boldsymbol{X} = \boldsymbol{Z}. \tag{6.57}$$

Using (6.57) in (6.54) results in

$$\begin{bmatrix} \boldsymbol{X} + \boldsymbol{X}^{\mathrm{T}} - \boldsymbol{Y} & \boldsymbol{X}^{\mathrm{T}}(\hat{\boldsymbol{A}} - \hat{\boldsymbol{B}}\hat{\boldsymbol{K}})^{\mathrm{T}} \\ (\hat{\boldsymbol{A}} - \hat{\boldsymbol{B}}\hat{\boldsymbol{K}})\boldsymbol{X} & (1-\gamma)\boldsymbol{Y} \end{bmatrix} \succ 0. \tag{6.58}$$

Applying theorem 3 from [49] to (6.58) with $N = 1$, $G_1 = \boldsymbol{X}$, $S_1 = \boldsymbol{Y}$ and $A_1 = (\hat{\boldsymbol{A}} - \hat{\boldsymbol{B}}\hat{\boldsymbol{K}})$ proofs the global asymptotic stability of the nominal closed loop system. Furthermore, the specific structure of $\boldsymbol{K}_u$ ensures that the i$^{\text{th}}$ controller node uses locally available information only. □

Remark 6.5. Some applications trigger the need to design the nominal control law in such a way that it depends exclusively on the measurements $\boldsymbol{x}_k$ and not on the history of the actuating signals. The solution to this problem represents a special case of theorem 6.4. Setting $\boldsymbol{Z}_u = \boldsymbol{0}$ and using a dense matrix $\boldsymbol{X}_3 \in \mathbb{R}^{\mu\times\mu}$ results in a nominal control law (6.46) with $\boldsymbol{K}_u = \boldsymbol{0}$.

6.2.2 Sliding Mode Control Law

In the following theorem, the sliding mode part of the control law is designed such that the input channels are decoupled and no communication between the controller nodes is necessary.

Theorem 6.6. Consider the buffered networked control system depicted in figure 3.5 with plant (1.5) and sampling time T. Also, assume that assumptions 1.2, 1.4, 2.4 and 2.6 are satisfied and the change rate of the ith perturbation $f_{i,k}$ is bounded by

$$\sup\left|\frac{f_{i,k+1} - f_{i,k}}{T}\right| = L_{f_i} < \infty, \qquad i = 1, 2, \ldots, m. \tag{6.59}$$

The discrete time integral sliding variables are defined as

$$\boldsymbol{\sigma}_k = \begin{bmatrix}\sigma_{1,k} & \cdots & \sigma_{m,k}\end{bmatrix}^{\mathrm{T}} = \hat{\boldsymbol{M}}\boldsymbol{\xi}_k + \boldsymbol{w}_k \tag{6.60}$$

with

$$\hat{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{m}_1^{\mathrm{T}} & 1 & \boldsymbol{0}_{1\times\delta_1-1} & 0 & \boldsymbol{0}_{1\times\delta_2-1} & \cdots & 0 & \boldsymbol{0}_{1\times\delta_m-1} \\ \boldsymbol{m}_2^{\mathrm{T}} & 0 & \boldsymbol{0}_{1\times\delta_1-1} & 1 & \boldsymbol{0}_{1\times\delta_2-1} & \cdots & 0 & \boldsymbol{0}_{1\times\delta_m-1} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \cdots & \vdots & \vdots \\ \boldsymbol{m}_m^{\mathrm{T}} & 0 & \boldsymbol{0}_{1\times\delta_1-1} & 0 & \boldsymbol{0}_{1\times\delta_2-1} & \cdots & 1 & \boldsymbol{0}_{1\times\delta_m-1} \end{bmatrix} \tag{6.61}$$

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{m}_1^{\mathrm{T}} \\ \boldsymbol{m}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{m}_m^{\mathrm{T}} \end{bmatrix} = \boldsymbol{B}^+ \tag{6.62}$$

where $\boldsymbol{B}^+ = (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B})^{-1}\boldsymbol{B}$ denotes the left inverse of $\boldsymbol{B}$. Let the nominal control law in (6.43) be given by (6.46) and

$$\boldsymbol{w}_{k+1} = \begin{bmatrix} w_{1,k+1} & w_{2,k+1} & \cdots & w_{m,k+1}\end{bmatrix}^{\mathrm{T}} = -\hat{\boldsymbol{M}}(\hat{\boldsymbol{A}}\boldsymbol{\xi}_k + \hat{\boldsymbol{B}}\boldsymbol{u}_k^N) \tag{6.63}$$

with

$$\boldsymbol{w}_0 = -\boldsymbol{M}\boldsymbol{x}_0. \tag{6.64}$$

The ith element of the sliding mode part $u_{i,k}^S$ in (6.43) is given by

$$u_{i,k}^S = \sigma_{i,k} + T\tilde{u}_{i,k}^S \tag{6.65}$$

with

$$\begin{aligned}\tilde{u}^S_{i,k} &= \frac{1}{T}(\tilde{\ell}_{i,k}-1)\sigma_{i,k}+\nu_{i,k}\\ \nu_{i,k+1} &= \nu_{i,k}+\ell_{i,k}\sigma_{i,k}\\ \tilde{\ell}_{i,k} &= q^{(1)}_{i,k}+q^{(2)}_{i,k}-1\\ \ell_{i,k} &= \frac{1}{T}\left(\tilde{\ell}_{i,k}-q^{(1)}_{i,k}q^{(2)}_{i,k}\right).\end{aligned} \tag{6.66}$$

For the explicit Euler discretized super twisting algorithm use

$$q^{(1)}_{i,k} = 1+Ts^{(1)}_i(\sigma_{i,k}), \qquad q^{(2)}_{i,k} = 1+Ts^{(2)}_i(\sigma_{i,k}) \tag{6.67}$$

and for the matching approach use

$$q^{(1)}_{i,k} = \begin{cases} e^{s^{(1)}_i(\sigma_{i,k})T} & \sigma_{i,k}\neq 0\\ 0 & \sigma_{i,k}=0\end{cases}, \qquad q^{(2)}_{i,k} = \begin{cases} e^{s^{(2)}_i(\sigma_{i,k})T} & \sigma_{i,k}\neq 0\\ 0 & \sigma_{i,k}=0\end{cases} \tag{6.68}$$

where

$$s^{(1)}_i(\sigma_{i,k}) = p^{(1)}_i\,|\sigma_{i,k}|^{-\frac{1}{2}}, \qquad s^{(2)}_i(\sigma_{i,k}) = p^{(2)}_i\,|\sigma_{i,k}|^{-\frac{1}{2}}. \tag{6.69}$$

If the parameter settings

$$p^{(1)}_i = \left(-0.75-\frac{\sqrt{2.15}}{2}i\right)\sqrt{\frac{L_{f_i}}{T}}, \qquad p^{(2)}_i = \left(-0.75+\frac{\sqrt{2.15}}{2}i\right)\sqrt{\frac{L_{f_i}}{T}} \tag{6.70}$$

are used, then

1. the states $\boldsymbol{x}_k$ of the plant are ultimately bounded,
2. only measurements $\boldsymbol{x}_k$ and local information, i.e. the history of the own control signals $u_{i,k}$, are used in each controller node.

Proof. The proof consists of two parts. In the first part, it will be shown that a sliding mode controller based on theorem 6.6 results in the ultimate boundedness of the states $\boldsymbol{x}_k$. In the second part, it will be shown that the resulting control law uses the locally available information only. The initial condition (6.64) ensures that the initial value of the sliding variable $\boldsymbol{\sigma}_0=\boldsymbol{0}$ since the previous values of the actuating signals are assumed to be zero, i.e.

$$u_{1,-1}=u_{1,-2}=\cdots=u_{1,-\delta_1}=\cdots=u_{m,-1}=\cdots=u_{m,-\delta_m}=0. \tag{6.71}$$

Using (6.42) and (6.43) to compute the forward increment of (6.60) results in

$$\boldsymbol{\sigma}_{k+1} = \hat{\boldsymbol{M}}\hat{\boldsymbol{A}}\boldsymbol{\xi}_k+\hat{\boldsymbol{M}}\hat{\boldsymbol{B}}(\boldsymbol{u}^N_k+\boldsymbol{u}^S_k)+\hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f\boldsymbol{f}_k+\boldsymbol{w}_{k+1}. \tag{6.72}$$

Applying (6.63) yields

$$\boldsymbol{\sigma}_{k+1} = \hat{\boldsymbol{M}}\hat{\boldsymbol{B}}\boldsymbol{u}^S_k+\hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f\boldsymbol{f}_k. \tag{6.73}$$

Due to assumption 1.2, $\boldsymbol{B}_c$ has full column rank and the sampling time is not equal to a pathological sampling time. Thus, the left inverse $\boldsymbol{M} = \boldsymbol{B}^+$ exists. As a consequence,

$$\hat{\boldsymbol{M}}\hat{\boldsymbol{B}} = \hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f = \boldsymbol{I}_m \tag{6.74}$$

is satisfied using (6.61) and (6.62). This simplifies (6.73) even further to

$$\boldsymbol{\sigma}_{k+1} = \boldsymbol{u}_k^S + \boldsymbol{f}_k. \tag{6.75}$$

From (6.75) it is clear that the i^th^ sliding variable $\sigma_{i,k}$ is only affected by the corresponding control signal $u_{i,k}^S$ and perturbation $f_{i,k}$. Due to this property, m discrete time sliding mode controllers can independently be designed using the same methods as proposed in theorem 6.2. This results in the sliding mode part of the control law as given in (6.66)–(6.69). The parameter settings are given in (6.70).

To show that the i^th^ controller node only uses $\boldsymbol{x}_k$ and the history of $u_{i,k}$, variables $\boldsymbol{\sigma}_k$ and $\boldsymbol{w}_{k+1}$ have to be analyzed because they depend on the entire lifted state vector in theorem 6.6. Using (2.29) and (6.61) to evaluate the i^th^ component of (6.60) results in

$$\sigma_{i,k} = \boldsymbol{m}_i^{\mathrm{T}}\boldsymbol{x}_k + u_{i,k-1} + w_{i,k}. \tag{6.76}$$

Using (3.25) and (3.27) to compute $\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}\boldsymbol{\xi}_k$ results in

$$\hat{\boldsymbol{M}}\hat{\boldsymbol{A}}\boldsymbol{\xi}_k = \boldsymbol{M}\boldsymbol{A}\boldsymbol{x}_k + \underbrace{\boldsymbol{M}\begin{bmatrix}\boldsymbol{b}_1 & \boldsymbol{b}_2 & \cdots & \boldsymbol{b}_m\end{bmatrix}}_{\boldsymbol{I}_m}\begin{bmatrix} u_{1,k-\delta_1} \\ u_{2,k-\delta_2} \\ \vdots \\ u_{m,k-\delta_m}\end{bmatrix} \tag{6.77}$$

which simplifies the i^th^ component of (6.63) to

$$w_{i,k+1} = -\boldsymbol{m}_i^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}_k - u_{i,k}^N - u_{i,k-\delta_i} \tag{6.78}$$

using (6.43) and (6.61). From (6.76) and (6.78), it is clear that only local information is used in each controller node in addition to the measurements $\boldsymbol{x}_k$. □

Example 6.7. The effectiveness of the approach proposed in this section is shown by means of a numerical simulation. The plant is given by the unstable continuous time system

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \begin{bmatrix} -3 & -3 & -2 & 1 \\ 2 & -3 & -2 & 2 \\ 1 & 2 & -3 & 1 \\ -3 & -3 & 3 & 0 \end{bmatrix}\boldsymbol{x} + \begin{bmatrix} -1 & -3 & 0 \\ 1 & 3 & 1 \\ -3 & 3 & -1 \\ -2 & -3 & 2 \end{bmatrix}(\boldsymbol{u}^* + \boldsymbol{f}) \tag{6.79}$$

and the sampling time was chosen as $T = 0.1$s. The constant round trip times ensured by the buffers are known to be

$$\boldsymbol{\delta}T = \begin{bmatrix} 3 & 7 & 6 \end{bmatrix} T. \tag{6.80}$$

Constructing the lifted model (2.13) and solving the LMI given in theorem 6.4 for $\gamma = 0.02$ results in the nominal control law (6.46) with

$$\begin{aligned}
\boldsymbol{K}_x &= \begin{bmatrix} 0.248 & -0.021 & -0.484 & -0.505 \\ -0.002 & 0.001 & 0.003 & 0.004 \\ -0.005 & 0.002 & 0.006 & 0.007 \end{bmatrix} \\
\boldsymbol{k}_1^{\mathrm{T}} &= \begin{bmatrix} 0.453 & 0.357 & 0.21 \end{bmatrix} \\
\boldsymbol{k}_2^{\mathrm{T}} &= \begin{bmatrix} 79.634 & 57.195 & 31.486 & 16.267 & 9.209 & 4.907 & 2.22 \end{bmatrix} \cdot 10^{-3} \\
\boldsymbol{k}_3^{\mathrm{T}} &= \begin{bmatrix} 46.736 & 32.501 & 20.702 & 12.428 & 6.643 & 2.932 \end{bmatrix} \cdot 10^{-3}.
\end{aligned}$$

The perturbation is chosen as

$$\boldsymbol{f}_k = \boldsymbol{\kappa}_1 \sin(\boldsymbol{\omega}_1 kT) + \boldsymbol{\kappa}_2 \sin(\boldsymbol{\omega}_2 kT) + \boldsymbol{\kappa}_3 \tag{6.81}$$

with

$$\boldsymbol{\kappa}_1 = \boldsymbol{\kappa}_3 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \qquad \boldsymbol{\omega}_1 = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.3 \end{bmatrix}, \qquad \boldsymbol{\kappa}_2 = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}, \qquad \boldsymbol{\omega}_2 = \boldsymbol{\omega}_1 \pi. \tag{6.82}$$

Computing the exact change rates (6.59) of (6.81) results in

$$\boldsymbol{L_f} = \begin{bmatrix} L_{f_1} & L_{f_2} & L_{f_3} \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 1.042 & 1.657 & 1.842 \end{bmatrix}^{\mathrm{T}}. \tag{6.83}$$

Based on theorem 6.6, the sliding variable $\boldsymbol{\sigma}_k$ is given by (6.60) with

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{m}_1^{\mathrm{T}} \\ \boldsymbol{m}_2^{\mathrm{T}} \\ \boldsymbol{m}_3^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} -0.384 & 0.661 & -2.244 & -1.314 \\ -1.383 & 1.143 & 1.066 & -0.302 \\ -1.285 & 3.551 & -1.261 & 3.048 \end{bmatrix} \tag{6.84}$$

$$\hat{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{m}_1^{\mathrm{T}} & 1 & \boldsymbol{0}_{1\times 2} & 0 & \boldsymbol{0}_{1\times 6} & 0 & \boldsymbol{0}_{1\times 5} \\ \boldsymbol{m}_2^{\mathrm{T}} & 0 & \boldsymbol{0}_{1\times 2} & 1 & \boldsymbol{0}_{1\times 6} & 0 & \boldsymbol{0}_{1\times 5} \\ \boldsymbol{m}_3^{\mathrm{T}} & 0 & \boldsymbol{0}_{1\times 2} & 0 & \boldsymbol{0}_{1\times 6} & 1 & \boldsymbol{0}_{1\times 5} \end{bmatrix} \tag{6.85}$$

and

$$\boldsymbol{w}_{k+1} = \begin{bmatrix} \boldsymbol{w}_1^{\mathrm{T}} & \boldsymbol{0} & -1 & \boldsymbol{0} & 0 & \boldsymbol{0} & 0 \\ \boldsymbol{w}_2^{\mathrm{T}} & \boldsymbol{0} & 0 & \boldsymbol{0} & -1 & \boldsymbol{0} & 0 \\ \boldsymbol{w}_3^{\mathrm{T}} & \boldsymbol{0} & 0 & \boldsymbol{0} & 0 & \boldsymbol{0} & -1 \end{bmatrix} \boldsymbol{\xi}_k - \boldsymbol{u}_k^N \tag{6.86}$$

$$\begin{bmatrix} \boldsymbol{w}_1^{\mathrm{T}} \\ \boldsymbol{w}_2^{\mathrm{T}} \\ \boldsymbol{w}_3^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} 0.012 & -0.522 & 2.09 & 1.43 \\ 0.684 & -1.301 & -0.697 & 0.063 \\ 1.395 & -1.997 & 0.33 & -3.38 \end{bmatrix}. \tag{6.87}$$

One possible choice of parameters (6.70) considering (6.83) is given in table 6.2 where Λ_i is an upper bound of $\frac{L_{f_i}}{T}$.

Simulation results with initial condition $\boldsymbol{x}_0 = \begin{bmatrix} -7 & 7 & 3 & -3 \end{bmatrix}^{\mathrm{T}}$ for the sliding variables $\boldsymbol{\sigma}_k$ using the proposed approach are depicted in figure 6.6. This figure shows that the

Table 6.2: Example 6.7: Parameter settings for sliding mode controllers.

	Explicit Euler	Matching
Λ_1	11.467	87.568
$p_1^{(1)}$	$-2.540 - 2.483i$	$-7.018 - 6.861i$
$p_1^{(2)}$	$-2.540 + 2.483i$	$-7.018 + 6.861i$
Λ_2	18.223	139.158
$p_2^{(1)}$	$-3.202 - 3.130i$	$-8.847 - 8.649i$
$p_2^{(2)}$	$-3.202 + 3.130i$	$-8.847 + 8.649i$
Λ_3	20.267	154.768
$p_3^{(1)}$	$-3.376 - 3.301i$	$-9.330 - 9.121i$
$p_3^{(2)}$	$-3.376 + 3.301i$	$-9.330 + 9.121i$

sliding variable is ultimately bounded and that the sliding variables start at $\boldsymbol{\sigma}_0 = \mathbf{0}$, as was expected. The red and blue lines in figure 6.7 show the plant states $\boldsymbol{x}_k$ obtained with the proposed approach and the corresponding discrete time sliding mode controller. To demonstrate the massively increased accuracy achieved by using the sliding mode control technique, a simulation was performed without using the sliding mode part of the control law, i.e. $\boldsymbol{u}_k^S = \mathbf{0}$, and the results are shown by the green lines in figure 6.7. Comparing the green curves with the red and blue ones reveals the significant difference in accuracy. The simulation results for the control signals are depicted in figure 6.8.

6.3 Laboratory Experiment

In this section, the effectiveness of the previously proposed algorithm is shown by means of a laboratory experiment. Reconsider the mechanical system introduced in section 5.3.2. A photo of the laboratory setup and the mechanical scheme of this system are shown in figure 6.9. Again, the sampling time

$$T = 0.02s \tag{6.88}$$

is chosen. To show the performance of the perturbed networked control system, the perturbation

$$f_k = \frac{1}{3}\left[\sin(kT+5) + \sin\left(\frac{1}{\pi}(kT+5)\right) + 1\right] \tag{6.89}$$

is applied which consists of two sinusoidal functions with different frequencies and a constant. The exact change rate can be derived as

$$L_f = 0.439. \tag{6.90}$$

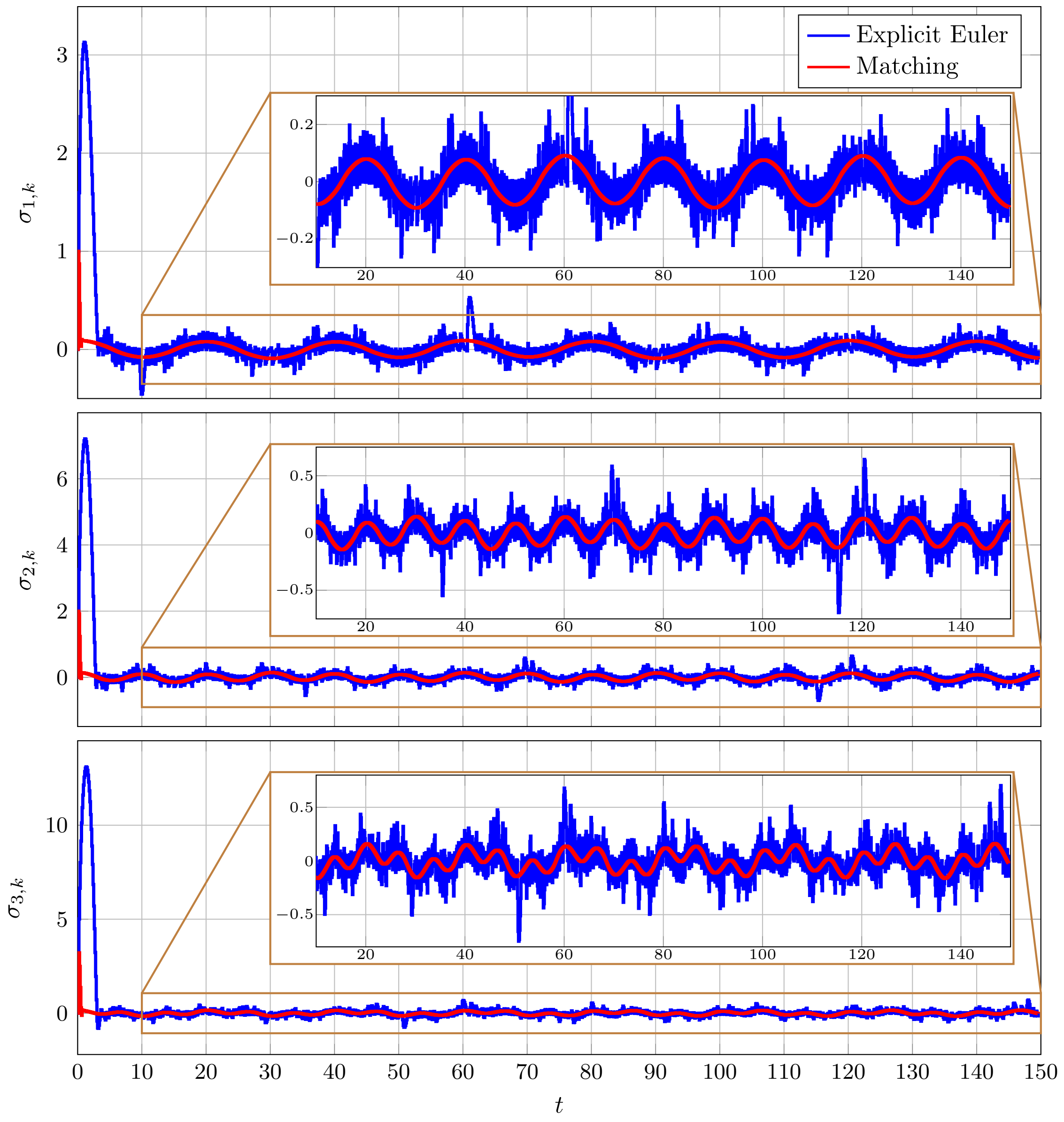


Figure 6.6: Example 6.7: Sliding variable $\boldsymbol{\sigma}_k$ using the spatially distributed control law with two different sliding mode controllers.

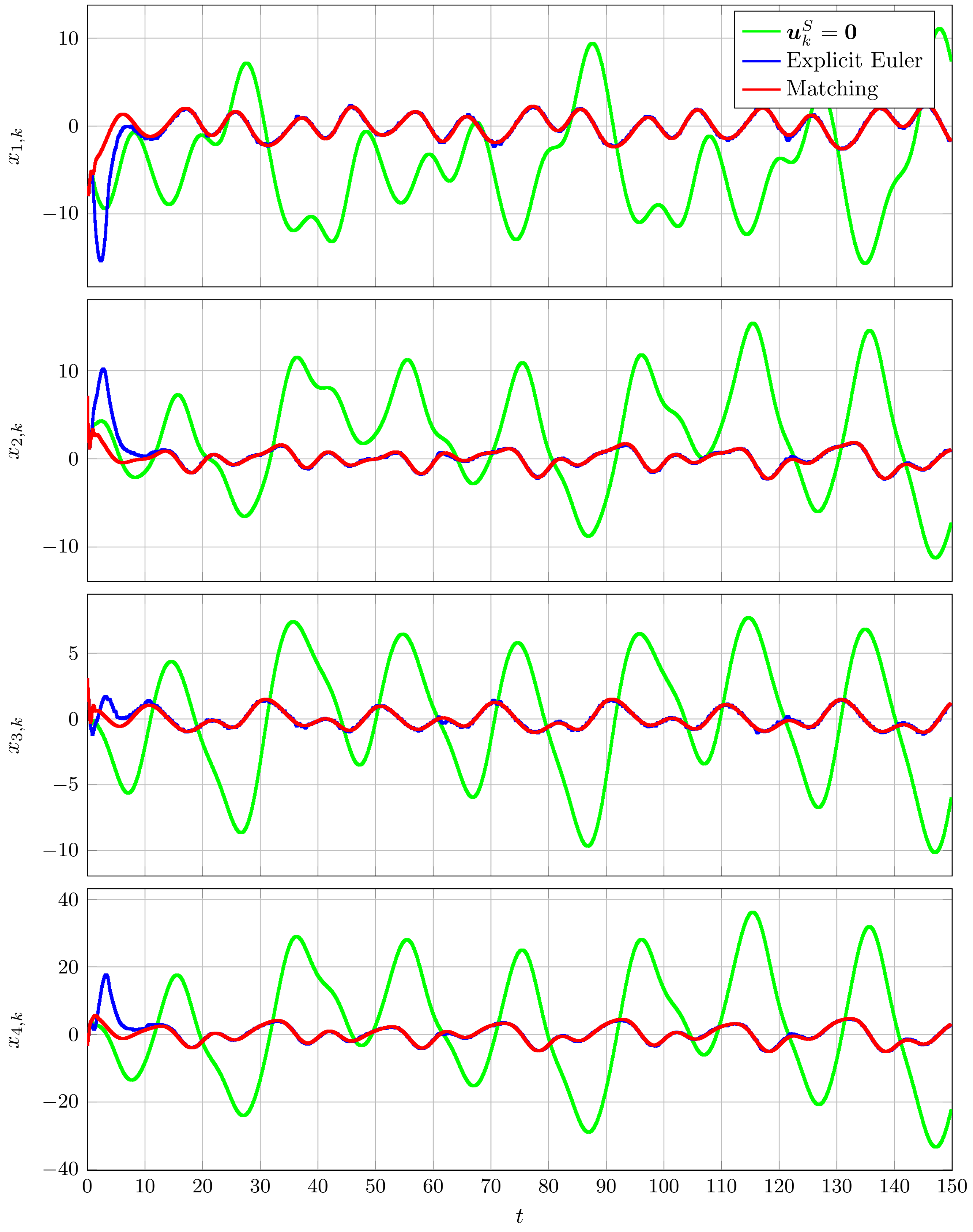


Figure 6.7: Example 6.7: Plant states $\boldsymbol{x}_k$ using the spatially distributed control law with two different sliding mode controllers.

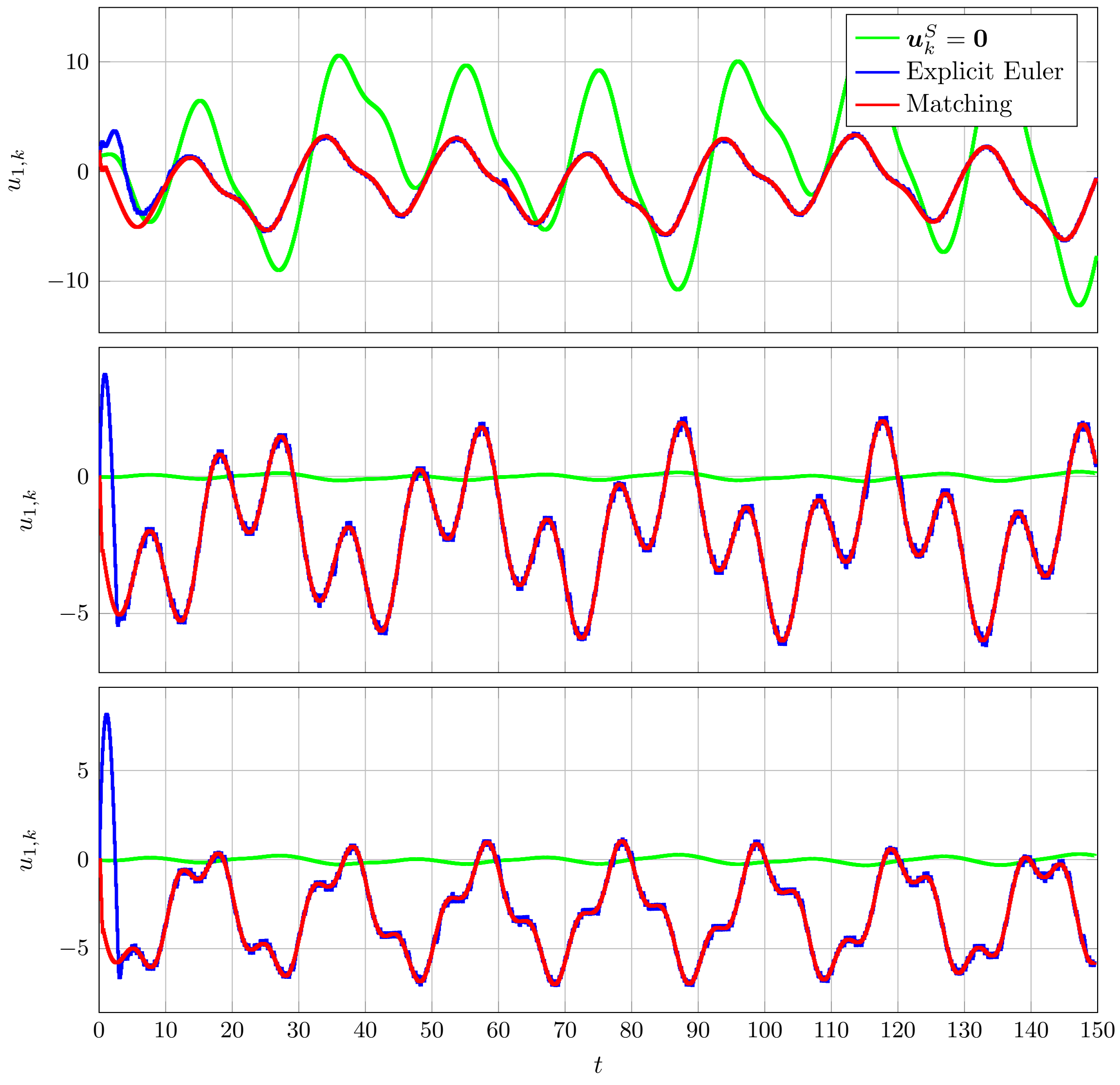


Figure 6.8: Example 6.7: Control signals u_k using the spatially distributed control law with two different sliding mode controllers.

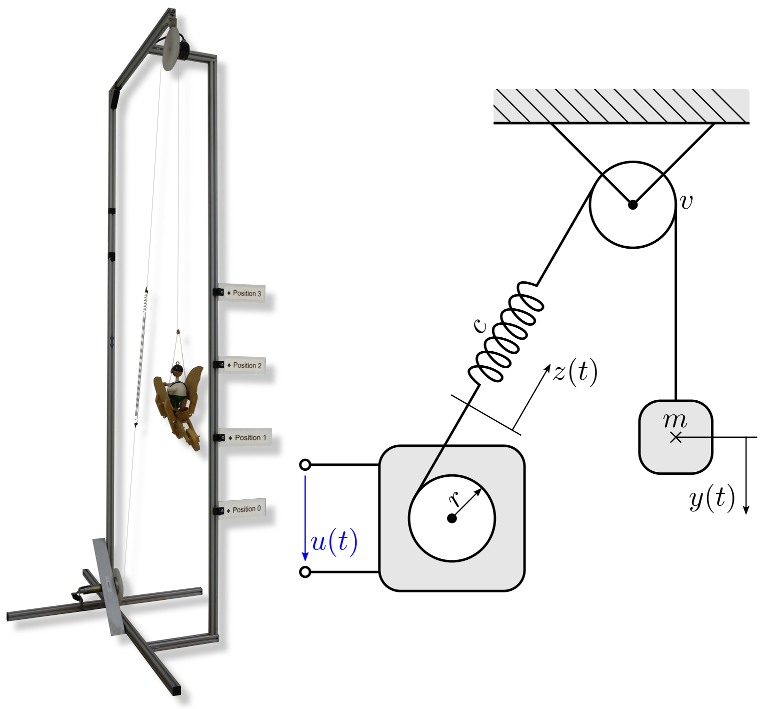


Figure 6.9: Spring mass laboratory experiment (left) and mechanical scheme (right).

In the given setup, encoders are attached to the pulley and the wheel at the motor. Thus, the states $x_{1,k} = y$ and $x_{3,k} = z$ are measurable. The state $x_{2,k}$ is obtained using a differentiating filter. Imperfections of sensors and the differentiating filter are neglected in the present investigations. The networked induced delay is implemented using two Simulink delay blocks, one to delay the sensor measurements and the other one to delay the control signal. These blocks are configured such that the worst case round trip time

$$\tau_k \leq 10T \tag{6.91}$$

is achieved, which results in $\delta = 10$. The nominal control law (6.7) was designed by placing the $n + \delta = 13$ poles at $z_i = e^{-10T}$, $i = 1, 2, \ldots, 13$ using Ackermann's formula. Vector $\hat{\boldsymbol{m}}^{\mathrm{T}}$ is designed according to (6.16). One possible choice for $\hat{\boldsymbol{m}}^{\mathrm{T}}$ is

$$\hat{\boldsymbol{m}}^{\mathrm{T}} = \begin{bmatrix} 1.143 & -9.157 & 584.02 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{6.92}$$

In theory, the choice of $\boldsymbol{m}^{\mathrm{T}}$ is only restricted by (6.16) but due to measurement noise and unmodeled dynamics in real world applications, it is necessary to find suitable values of those parameters. To estimate the quality of the current setting, choose $u_k^S = 0\ \forall k$ and evaluate the sliding variable σ_k. According to (6.17) the sliding variable in this case should equal the perturbation delayed by one step.

The result after this tuning process is depicted in figure 6.10. This figure illustrates the sliding variable σ_k derived from the measurements in red and the applied perturbation

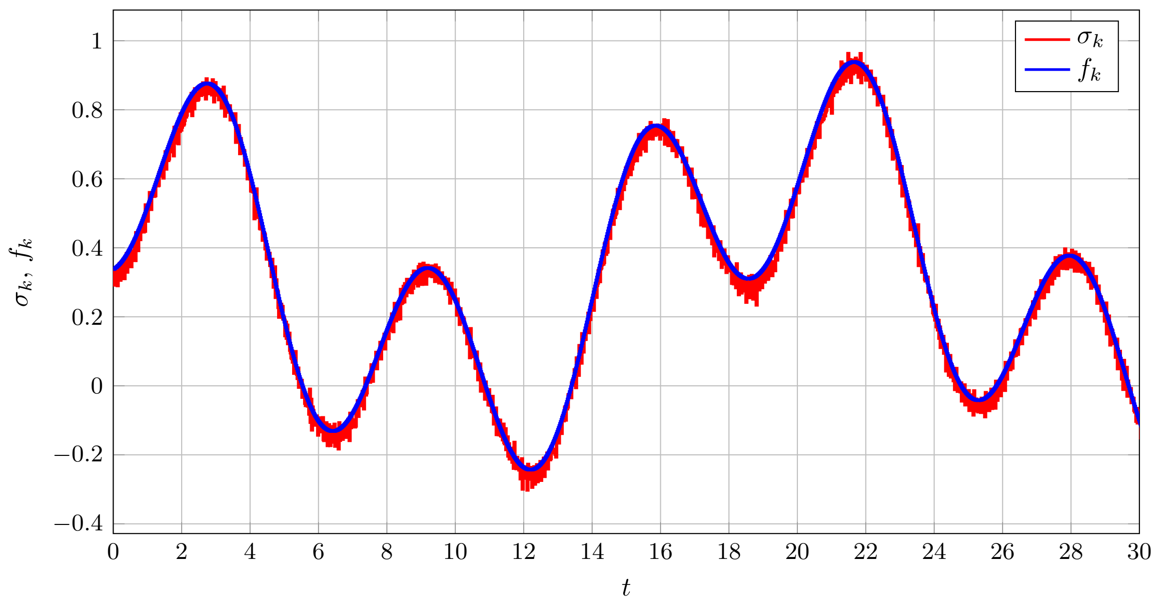


Figure 6.10: Laboratory experiment: Sliding variable σ_k for $u_k^S = 0$ and perturbation f_k.

in blue. Comparing the signals reveals that using the vector $\hat{\boldsymbol{m}}^{\mathrm{T}}$ as given in (6.92) is a reasonable choice since the sliding variable follows the perturbation very well.

To design the parameters $p^{(1)}$ and $p^{(2)}$ of the super twisting algorithm to ensure (6.29) is satisfied, the change rate L_f of the perturbation has to be known or estimated. In real world applications, this change rate is usually not exactly known and must therefore be estimated. As a too small choice could lead to instabilities, the value typically is significantly overestimated. In this laboratory experiment it is assumed that the estimated change rate $\Lambda = 132.615$ is about six times larger than the actual value $\frac{L_f}{T} = 21.972$. This choice leads to

$$p^{(1)} = -8.637 - 8.443i, \qquad p^{(2)} = -8.637 + 8.443i. \tag{6.93}$$

Figure 6.11 shows the evolution of the plant states $\boldsymbol{x}_k$ during the experiment using the proposed approach with the explicit Euler algorithm in blue and using the matching algorithm in red. In order to point out the effectiveness of the sliding mode based part of the control law, the results obtained with the nominal control law only are shown as green lines. Comparing the signals shows a significant increase of accuracy. This increased accuracy is also represented in the sliding variable since this variable acts as an accuracy measure. Figure 6.12 shows the values of the sliding variable σ_k obtained with the matching algorithm in red and with the explicit Euler algorithm in blue. Comparing figures 6.10 and 6.12 exemplifies that the magnitude of the sliding variable using the proposed approach is about ten times smaller compared to only using the nominal control law. In addition, figure 6.12 illustrates that the magnitude of σ_k is about three times smaller using the matching algorithm compared to using the explicit Euler variant. This is due to the rapidly increasing amplitude of discretization chattering for increasing values of the estimated change rate Λ.

This phenomenon is also depicted in figure 6.13 in which the magnitude of the sliding variable σ_k is shown for both algorithms and increasing values of the estimated change rate Λ. One can clearly see that the curve representing the explicit Euler discretized scheme ascents much faster than the one representing the matching algorithm. Hence, the negative influence of a high value of Λ on accuracy is small using the matching algorithm. This is a significant advantage for practical applications as the exact value L_f might be unknown.

6.4 Conclusions

In this chapter, an integral sliding mode approach based control strategy for spatially distributed networked control systems is proposed. In the first section of this chapter, the design procedure is described for single-input systems. No specific structure of the nominal control law has to be considered in the single-input case. However, a specific structure of the sliding variable is needed. It is shown that even in ideal quasi-sliding mode, i.e. the sliding variable equals zero, no exact compensation of the perturbation is ensured. In fact, past elements of the perturbation are compensated. Hence, a limited change rate of the perturbation was assumed and therefore the resulting problem allows the super twisting algorithm to be applied. Specifically, two discrete time versions of the super twisting algorithm are applied, namely the Euler discretized super twisting algorithm and the super twisting algorithm resulting from discretization using the matching approach. A numerical simulation shows the significant increase of accuracy achieved by using the sliding mode part of the control law. In addition, the properties of the discrete time super twisting algorithms are verified in simulation.

In the second section, the approach for single-input systems is extended to multi-input systems. Since no communication channel between the controller nodes is available, the nominal control law has to have a specific structure. A LMI based design procedure was developed to design linear state controllers with the desired structure that ensures asymptotic stability of the nominal closed loop system. Using a specific structure of the sliding variable, a decoupled design of the sliding mode part is possible. Thus, the sliding mode controller for each channel is designed individually. The design process and the achieved increase of accuracy is again shown by a numerical simulation.

The applicability of the proposed approach to real world scenarios is demonstrated by a single-input mass spring laboratory experiment. The results of this experiment reflect the expected properties of the proposed algorithms.

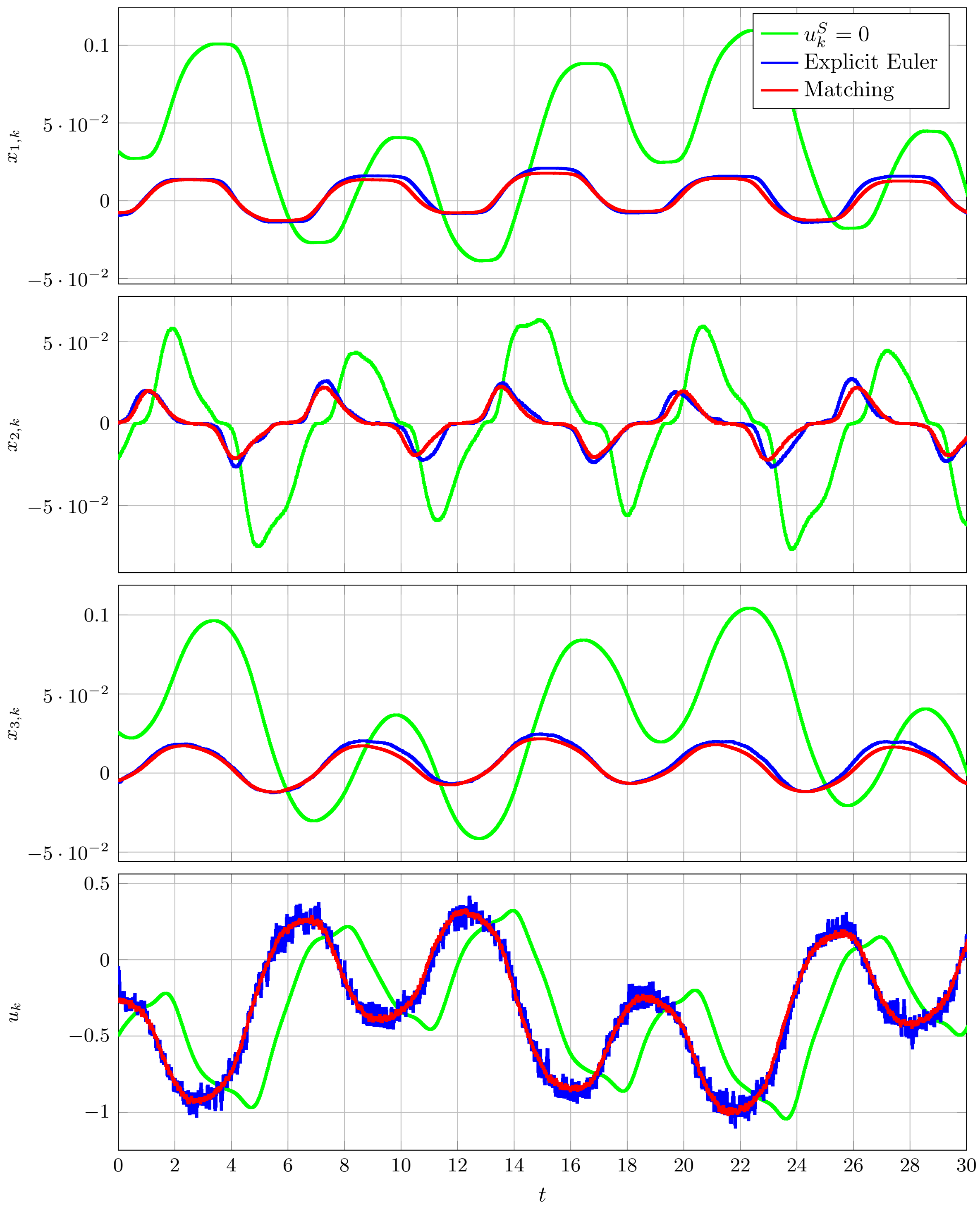


Figure 6.11: Laboratory experiment: States $\boldsymbol{x}_k$ and control signal u_k for $u_k^S = 0$ in green, using the proposed approach with the explicit Euler algorithm in blue and using the matching algorithm in red.

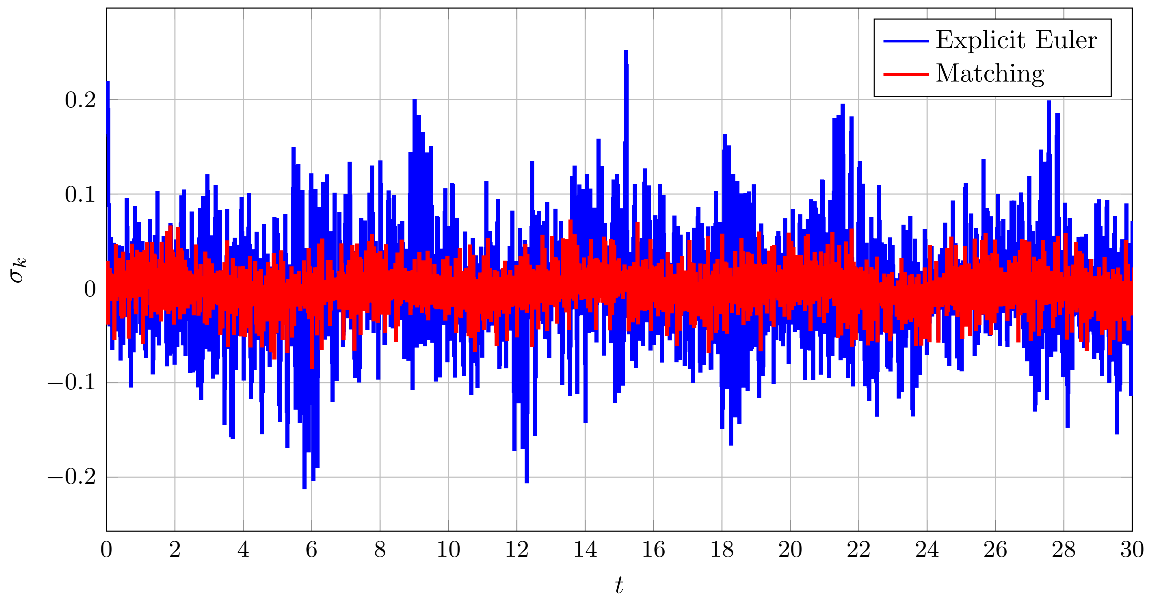


Figure 6.12: Laboratory experiment: Comparison of the sliding variable σ_k obtained with the matching algorithm in red and explicit Euler algorithm in blue.

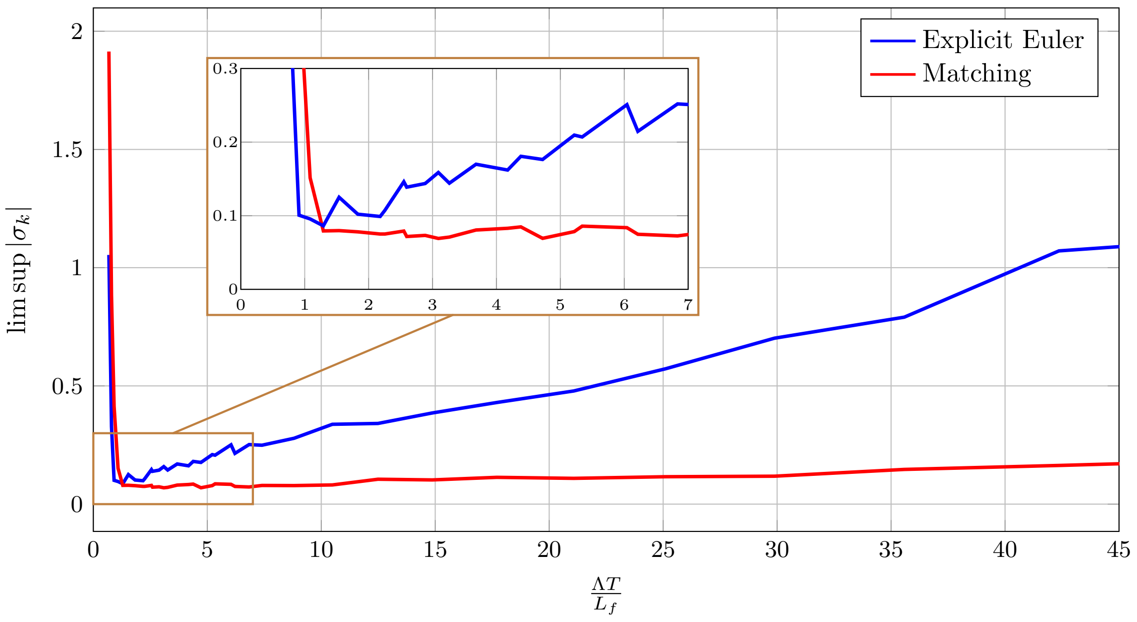


Figure 6.13: Laboratory experiment: Magnitude of sliding variable σ_k with increasing values of the estimated change rate Λ.

Chapter 7

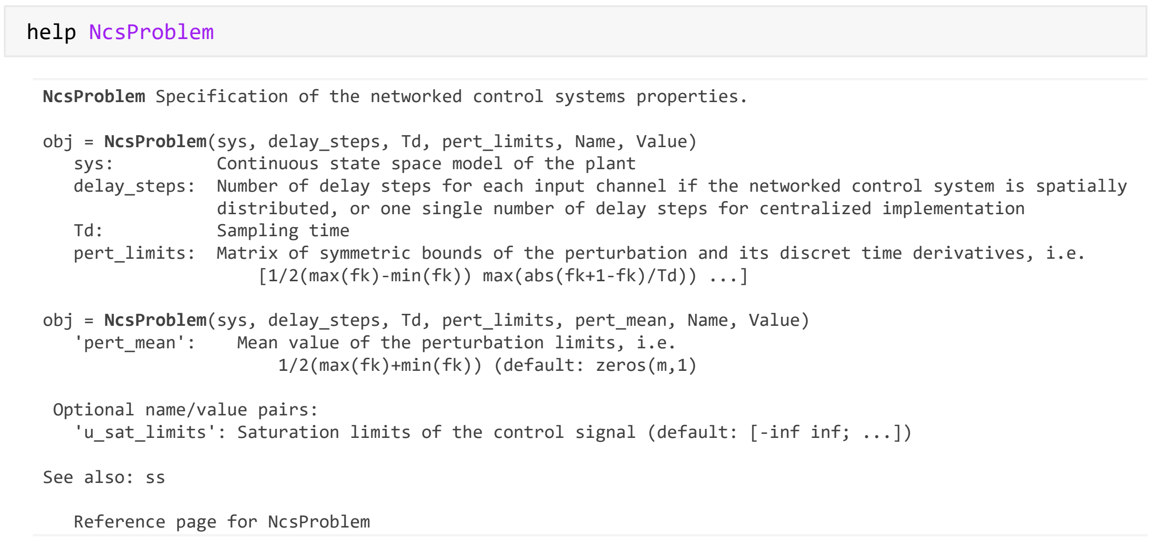
The Simulation Toolbox

Contents

In the previous chapters numerous simulation results are shown. In this chapter, the developed simulation environment for sliding mode based networked control systems is described. The simulation environment is based on MATLAB and SIMULINK in combination with the TRUETIME simulation software described in [23]. The simulation environment was designed such that it can be easily customized, e.g. new control strategies can be integrated with very little effort.

7.1 User Guide

Before the controllers can be designed or a simulation can be started, all relevant information on the considered networked control system has to be specified. All this information is collected in an “NcsProblem” object. The output of the help command in MATLAB results in the lines shown in figure 7.1. The constructor of the “NcsProblem” class is called with a



```
help NcsProblem
```

```
NcsProblem Specification of the networked control systems properties.

obj = NcsProblem(sys, delay_steps, Td, pert_limits, Name, Value)
   sys:          Continuous state space model of the plant
   delay_steps:  Number of delay steps for each input channel if the networked control system is spatially
                 distributed, or one single number of delay steps for centralized implementation
   Td:           Sampling time
   pert_limits:  Matrix of symmetric bounds of the perturbation and its discret time derivatives, i.e.
                     [1/2(max(fk)-min(fk)) max(abs(fk+1-fk)/Td)) ...]

obj = NcsProblem(sys, delay_steps, Td, pert_limits, pert_mean, Name, Value)
   'pert_mean':    Mean value of the perturbation limits, i.e.
                       1/2(max(fk)+min(fk)) (default: zeros(m,1)

 Optional name/value pairs:
   'u_sat_limits': Saturation limits of the control signal (default: [-inf inf; ...])

See also: ss

   Reference page for NcsProblem
```

Figure 7.1: Help output for the “NcsProblem” class.

state space model of the continuous time plant (1.5) followed by the vector of maximum delay steps $\boldsymbol{\delta}$ for a spatially distributed networked system or one single number of delay steps δ for a centralized networked control system. These integer values are used in the implementation of the buffers to ensure constant round trip times. As a consequence, these values have to satisfy (1.11). The variable “Td” represents the sampling time T and has to be non-pathological. The last required variable, “pert_limits”, represents the limits of the perturbation and its discrete time derivatives. For the control laws considered in this dissertation, information on the amplitude $\tilde{\boldsymbol{f}}$ of the perturbation $\boldsymbol{f}_k$ or its change rate $\boldsymbol{L_f}$ is necessary, i.e.

$$\text{pert\_limits} = \begin{bmatrix} \tilde{\boldsymbol{f}} & \boldsymbol{L_f} \end{bmatrix}. \tag{7.1}$$

The mean value $\hat{\boldsymbol{f}}$ of perturbation $\boldsymbol{f}_k$ and the saturation limits of the control signal can optionally be provided, see figure 7.1. The next two subsections describe how to use the simulation toolbox to design and simulate centralized as well as spatially distributed networked control systems.

7.1.1 Centralized Networked Control Systems

A minimal working example to design and simulate a centralized networked control system is given in listing 7.1. The corresponding SIMULINK block diagram is given in figure 7.2.

```
1  sys = ss(1,1,1,0);                                    %Continuous plant
2  Td = 0.1;                                             %Samplingtime
3  ampert = 1;                                           %Perturbation amplitude
4  meanpert = 1;                                         %Perturbation mean
5  maxdelstep = 3;                                       %Maximum delay steps
6  fk = @(k) 0.4*sin(0.8*k*Td)+0.6*sin(pi/2*k*Td)+1;     %Perturbation
7  ncs_prob = NcsProblem(sys,maxdelstep,Td,amppert,meanpert);
8  reachinglaw = {PredSW_Reachinglaw(1.01,[0 0.9])};     %Select reaching law
9  NCS = NCS_Centralized(ncs_prob, reachinglaw, [], 'tsim',50);
10 x0 = 1;                                               %Initial condition
11 sim('Centralized_sim')                                %Perform simulation
12
13 results = NCS.Results;                                %Get simulation results
14 plot(results.sigmak)                                  %Plot sliding variable
```

Listing 7.1: Minimal working example to simulate a centralized networked control system.

In this example, the plant is given by the first order system

$$\frac{\mathrm{d}x}{\mathrm{d}t} = x + u^* + f, \tag{7.2}$$

the sampling time by $T = 0.1\,\mathrm{s}$, the round trip time is bounded by

$$\tau_k \leq 3T \tag{7.3}$$

and the applied perturbation is

$$f_k = 0.4\sin(0.8kT) + 0.6\sin\left(\frac{\pi}{2}kT\right) + 1. \tag{7.4}$$

In line 7 of listing 7.1, a “NcsProblem” object is instantiated as previously described. In line 8, a reaching law object which implements the predictive switching reaching law is created and placed in a cell array. For multi-input systems, this cell array has to include a reaching law object for each input channel. It is not necessary to choose the same type of reaching law for each input channel. The three reaching laws proposed in chapter 5 are available and can be constructed as explained in the output of the help command (see figures 7.3–7.5).

In line 9, an “NCS_Centralized” object is instantiated. Creating this object automatically creates all objects necessary for the simulation, i.e. sensor node, controller node, time delays and buffer. Figure 7.6 shows the output of the help command for this “NCS_Centralized” class. The first argument passed to the constructor of this class is the “NcsProblem” object.

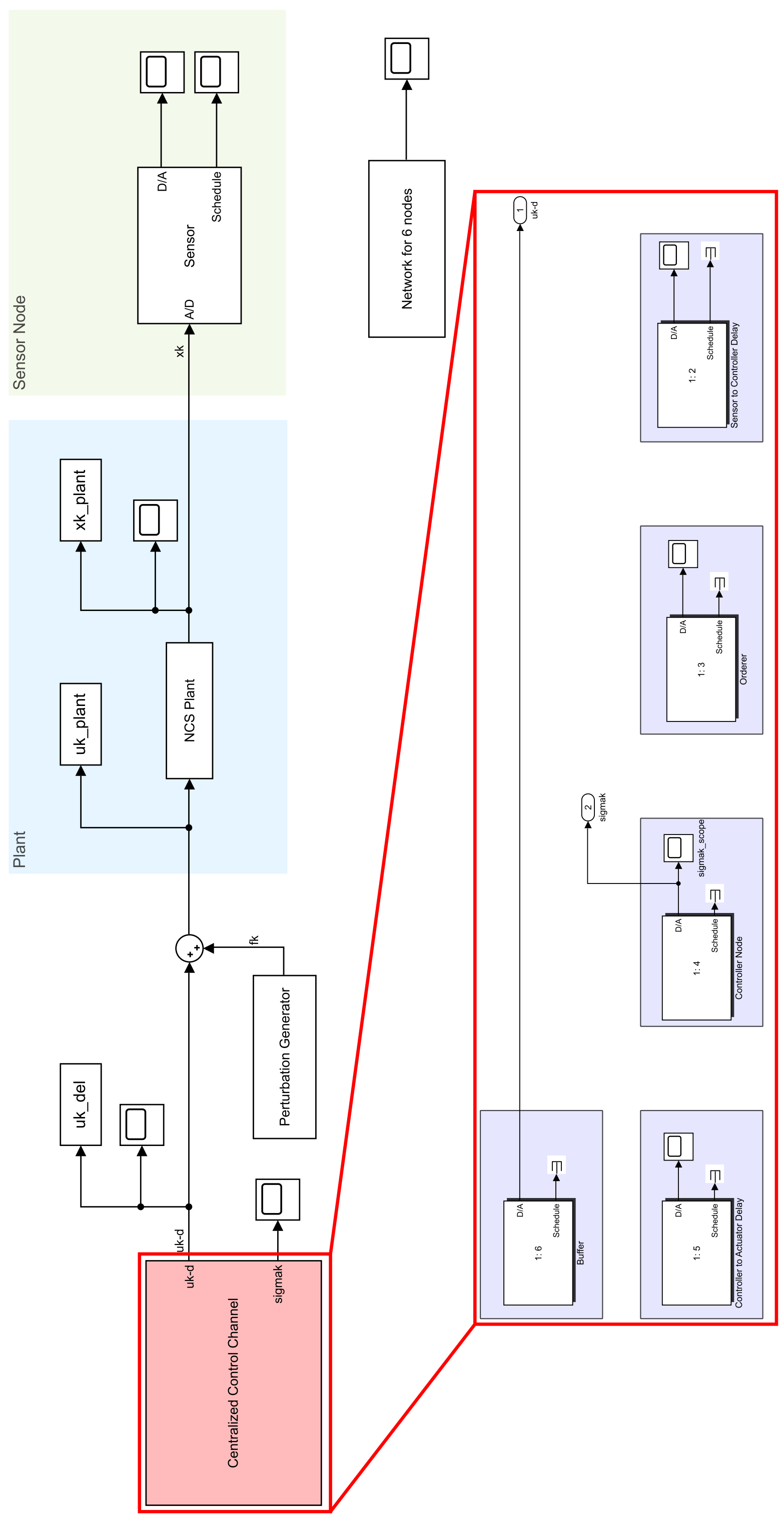


Figure 7.2: SIMULINK block diagram to simulate centralized networked control systems.

```
help SW_Reachinglaw
```

```
SW_Reachinglaw: Implementation of the switching reaching law

 obj = SW_Reachinglaw(rho_factor, alpha)
   rho_factor: The value of rho is given by rho_factor times the minimal possible value (rho_factor > 1)
   alpha:      Value of the linear part of the reaching law (0 <= alpha < 1)

See also: ReachingLaw

   Reference page for SW_Reachinglaw
```

Figure 7.3: Help output for the "SW_Reachinglaw" class.

```
help NonSW_Reachinglaw
```

```
NonSW_Reachinglaw: Implementation of the nonswitching reaching law

 obj = NonSW_Reachinglaw(kstar_factor, s0_factor)
   kstar_factor: The value of kstar is given by kstar_factor times the maximal possible value
                 (kstar_factor < 1)
   s0_factor:    The value of s0 is given by s0_factor times the maximal possible value (s0_factor < 1)

See also: ReachingLaw

   Reference page for NonSW_Reachinglaw
```

Figure 7.4: Help output for the "NonSW_Reachinglaw" class.

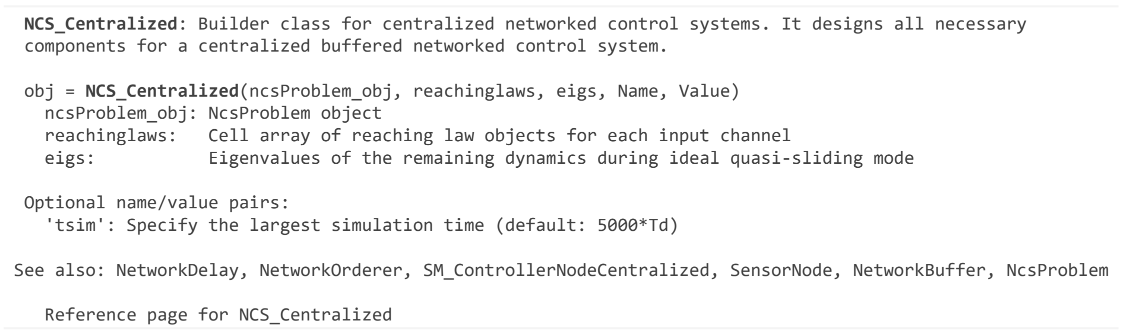
```
help PredSW_Reachinglaw
```

```
PredSW_Reachinglaw: Implementation of the predictive switching reaching law

 obj = PredSW_Reachinglaw(rho_factor, Name, Value)
   rho_factor: The value of rho is given by rho_factor times the minimal possible value (rho_factor > 1)

 Optional name/value pairs:
   alpha_limits: Lower and upper bound of the linear part of the reaching law (default: [0 0.98])

See also: ReachingLaw

   Reference page for PredSW_Reachinglaw
```

Figure 7.5: Help output for the "PredSW_Reachinglaw" class.

The second argument, “reachinglaws”, is the cell array containing a reaching law object for each input channel. Eigenvalues of the remaining dynamics during ideal quasi-sliding mode are specified through the third argument. In the present example, the plant is of order one. Hence, no dynamics remain during ideal quasi-sliding mode. Therefore, no eigenvalues have to be specified. Using the optional parameter “tsim”, the maximum simulation time of 50 seconds is set.

```
help NCS_Centralized
```



```
NCS_Centralized: Builder class for centralized networked control systems. It designs all necessary
components for a centralized buffered networked control system.

obj = NCS_Centralized(ncsProblem_obj, reachinglaws, eigs, Name, Value)
  ncsProblem_obj: NcsProblem object
  reachinglaws:   Cell array of reaching law objects for each input channel
  eigs:           Eigenvalues of the remaining dynamics during ideal quasi-sliding mode

Optional name/value pairs:
  'tsim': Specify the largest simulation time (default: 5000*Td)

See also: NetworkDelay, NetworkOrderer, SM_ControllerNodeCentralized, SensorNode, NetworkBuffer, NcsProblem

  Reference page for NCS_Centralized
```

Figure 7.6: Help output for the “NCS_Centralized” class.

The command in line 11 starts the simulation of the block diagram shown in figure 7.2. All blocks are included in a SIMULINK block library and are configured by passing a “NCS_Centralized” object. The “Sensor” block samples the states of the plant periodically with sampling time T and sends the results to the “Sensor to Controller Delay” block. This block forwards the packets delayed by a random delay to the “Orderer” block which sequences the packets to ensure that the “Controller Node” receives the packets in the correct order. This is necessary since the dynamic control law can only be evaluated after all previous measurement samples have been received. The controller node evaluates the control law and transmits the results to the “Controller to Actuator Delay” block that forwards the packets delayed by a random delay to the “Buffer” block. This block implements the buffer to guarantee the constant round trip time. The random delays are generated in a way that the worst case round trip time δT defined in the “NcsProblem” object is never exceeded. However, the buffer checks in each step weather the current round trip time is smaller and prints an error message if this is violated. The matched perturbation $\boldsymbol{f}_k$ is generated in the “Perturbation Generator”. This block is configured by passing a function handle that defines the perturbation $\boldsymbol{f}_k$. The passed function is evaluated in each sampling step with the integer time index k. The input to the plant is then given by the sum of the delayed controller output $\boldsymbol{u}_{k-\delta}$ and the perturbation $\boldsymbol{f}_k$.

While the simulation is running, a progress bar indicates the remaining simulation time. After the simulation has finished, the member variable “Results” of the “NCS_Centralized” object contains a structure with the simulation results (see line 14).

7.1.2 Spatially Distributed Networked Control Systems

A minimal working example to design and simulate a spatially distributed networked control system is given in listing 7.2. The corresponding SIMULINK block diagram is shown in figure 7.7.

```
1  sys = ss(diag([-1 -2]),eye(2),eye(2),zeros(2));%Continuous plant
2  Td = 0.1;                                       %Sampling time
3  amppert = [1 1.3; 2 2.3];                       %Perturbation amplitude
4                                                  %and change rate
5  meanpert = [1;2];                               %Perturbation mean
6  maxdelstep = [3 4];                             %Maximum delay steps
7                                                  %for each channel
8  fk = @(k) [0.4*sin(0.8*k*Td)+0.6*sin(pi/2*k*Td)+1; ...
9       1.2*cos(0.8*k*Td)+0.8*cos(pi/2*k*Td)+2];   %Perturbation
10 ncs_prob = NcsProblem(sys,maxdelstep,Td,amppert,meanpert);
11 NCS = NCS_ISMC(ncs_prob,'ExplicitSTA', 'tsim', 30);
12 x0 = [1 2];                                     %Initial condition
13 sim('Spat_Dist_sim')                            %Perform simulation
14
15 results = NCS.Results;                          %Get simulation results
16 plot(results.sigmak(1))                         %Plot first sliding varible
```

Listing 7.2: Minimal working example to simulate a spatially distributed networked control system.

In this example, the plant is given by the second order system

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \begin{bmatrix} -1 & 0 \\ 0 & -2 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} (\boldsymbol{u}^* + \boldsymbol{f}), \tag{7.5}$$

the sampling time by $T = 0.1\,\mathrm{s}$, the round trip times are bounded by

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} \leq \boldsymbol{\tau}_k \leq \begin{bmatrix} 3 \\ 4 \end{bmatrix} T \tag{7.6}$$

and the applied perturbations are

$$\boldsymbol{f}_k = \begin{bmatrix} 0.4\sin(0.8kT) + 0.6\sin\left(\frac{\pi}{2}kT\right) + 1 \\ 1.2\cos(0.8kT) + 0.8\cos\left(\frac{\pi}{2}kT\right) + 2 \end{bmatrix}. \tag{7.7}$$

The “NcsProblem” object is again created as previously described. In line 11 of listing 7.2, an “NCS_ISMC” object is created by passing the “NcsProblem” object, the name of the desired sliding mode control law and the maximum simulation time. The output of the help command for this class is shown in figure 7.8. As this output indicates, both discrete time versions of the super twisting algorithm used in this dissertation are available. The eigenvalues of the nominal closed loop system matrix can only be provided for single-input systems since multi-input systems require a specific structure of the nominal control law. Therefore, the LMI approach is used for multi-input systems which can be tuned with the parameter γ. The matrix $\hat{\boldsymbol{M}}$ can explicitly be provided with the keyword “Mlift”. However, it has to satisfy

$$\hat{\boldsymbol{M}}\hat{\boldsymbol{B}}_f = \hat{\boldsymbol{M}}\hat{\boldsymbol{B}} = \boldsymbol{I}_m. \tag{7.8}$$

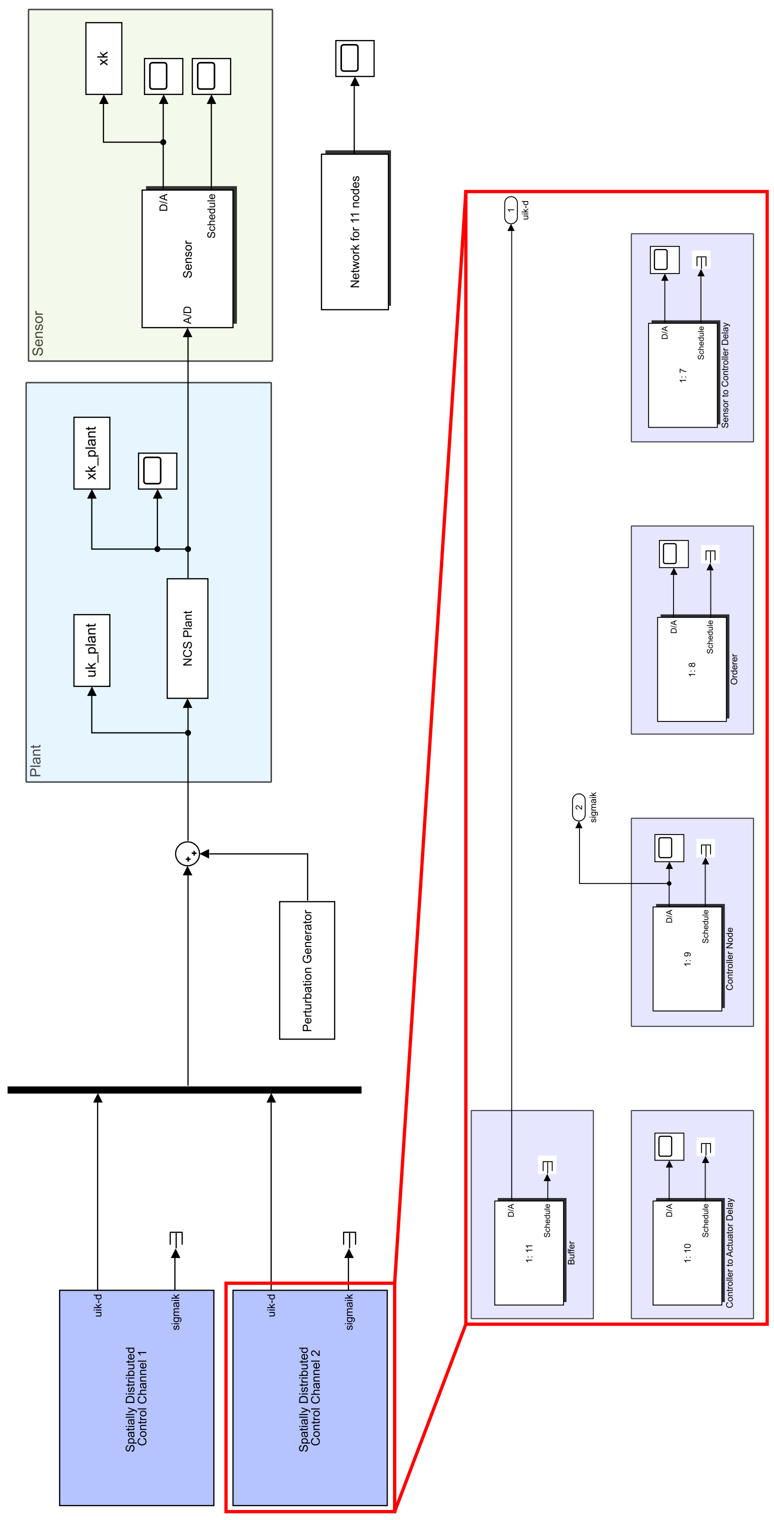
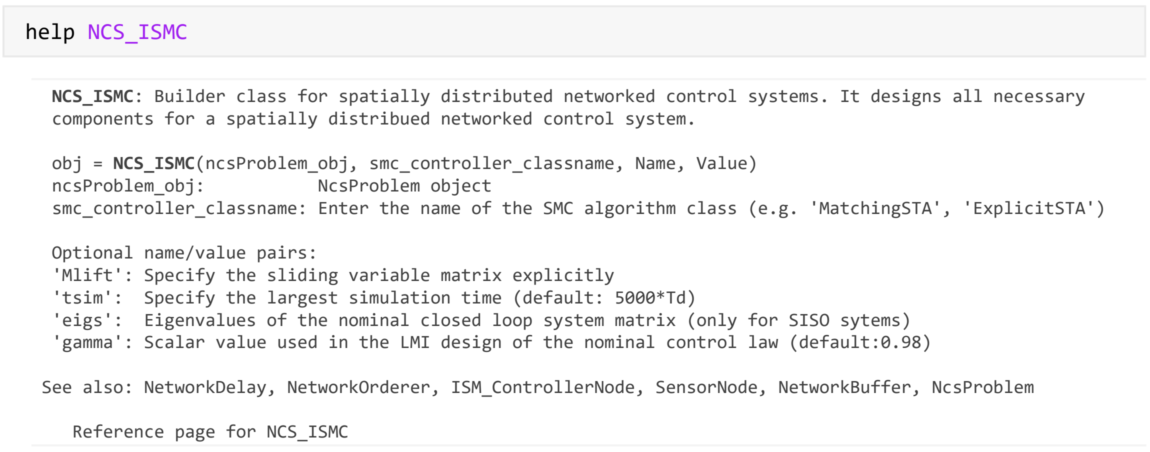


Figure 7.7: SIMULINK block diagram to simulate a spatially distributed networked control systems with two feedback channels.



```
help NCS_ISMC
```

```
  NCS_ISMC: Builder class for spatially distributed networked control systems. It designs all necessary
  components for a spatially distribued networked control system.

  obj = NCS_ISMC(ncsProblem_obj, smc_controller_classname, Name, Value)
  ncsProblem_obj:          NcsProblem object
  smc_controller_classname: Enter the name of the SMC algorithm class (e.g. 'MatchingSTA', 'ExplicitSTA')

  Optional name/value pairs:
  'Mlift': Specify the sliding variable matrix explicitly
  'tsim':  Specify the largest simulation time (default: 5000*Td)
  'eigs':  Eigenvalues of the nominal closed loop system matrix (only for SISO sytems)
  'gamma': Scalar value used in the LMI design of the nominal control law (default:0.98)

 See also: NetworkDelay, NetworkOrderer, ISM_ControllerNode, SensorNode, NetworkBuffer, NcsProblem

    Reference page for NCS_ISMC
```

Figure 7.8: Help output for the “NCS_ISMC” class.

Similar to the example shown in section 7.1.1, the simulation is performed by calling the command in line 13 and the results are obtained in line 15. Finally, in line 16, the simulation results of the first sliding varible are shown in a Matlab plot.

7.2 Integrate Custom Control Strategies

As previously mentioned, the simulation toolbox was designed to allow for easy integration of custom control strategies. The provided interfaces to implement custom control strategies are described in this section.

7.2.1 Centralized Networked Control Systems

The controller design for centralized networked systems proposed in chapter 5 uses sliding mode control laws based on the reaching law approach. As a consequence, a reaching law interface is provided by the abstract class “ReachingLaw”. A custom reaching law has to be derived from this abstract class and must therefore implement the abstract methods and properties. A minimal working example which implements the reaching law

$$\sigma_{i,k+\delta} = 0 \tag{7.9}$$

is given in listing 7.3.

The abstract method “reachinglaw(·)” implemented in line 9 of listing 7.3 is called for every received measurement packet. The reaching law is evaluated in line 12 and the width of the quasi-sliding mode band is set in line 11. If the reaching law implementation uses internal states, the abstract method “init_reachinglaw()” is used to initialize them. Reaching law specific simulation results, which should be available after the simulation has finished, can be returned in the abstract method “getSimResults()”. This method is called when

```
classdef Dummy_Reachinglaw < ReachingLaw %Derive from ReachingLaw
    properties
       qsmb %Property that provides the width of the quasi-sliding mode band
    end
    methods
        function obj = Dummy_Reachinglaw() %Empty constructor
        end

        function result = reachinglaw(obj,sigma_hist,k, xik)
            %Reaching law method is called for every received measurement packet
            obj.qsmb = 2*obj.amplpert; %Set the quasi-sliding mode band
            result = 0; %Evaluate the reaching law
        end

        function init_reachinglaw(obj)
        %This method is called before the simulation starts
        end

        function results = getSimResults(obj)
        %Method to get reaching law specific simulation results
        %This method is called when the simulation results are retrieved
            results = [];
        end
    end
end
```

Listing 7.3: Minimal working example to implement a custom reaching law for centralized networked control systems.

the simulation results are retrieved from the “NCS_Centralized” object which holds this reaching law object. To simulate a centralized networked control system using this custom reaching law, line 8 in listing 7.1 has to be replaced by

```
        reachinglaw = {Dummy_Reachinglaw()};
```

The custom reaching law will then automatically be used in the simulation.

7.2.2 Spatially Distributed Networked Control Systems

The controller design for spatially distributed networked systems proposed in chapter 6 is based on integral sliding mode control. The control law consists of a nominal part, which is designed by solving linear matrix inequalities, and a sliding mode part. In order to create a flexible simulation environment, an interface to implement custom sliding mode based control laws is available. This interface is given by the abstract class “SmcControlLaw”. A minimal working example which defines the custom sliding mode part of the control law

$$u_{i,k}^{S} = 0 \tag{7.10}$$

is given in listing 7.4. The abstract method “evaluate($\cdot$)” which is implemented in line 6 of listing 7.4 is called for every received measurement packet and returns $u_{i,k}^{S}$. To initialize internal states at the beginning of the simulation, the “init()” method is implemented. In this case, the control law does not have any internal states and therefore this method is

```
classdef ZeroSMC < SmcControlLaw %Derive from SmcControlLaw
    methods
        function obj = ZeroSMC() %Empty constructor
        end

        function uSk = evaluate(obj,sigmak)
            %Sliding mode part of the control law
            uSk = 0;
        end

        function init(obj)
            %This function is called before the simulation starts
        end
    end
end
```

Listing 7.4: Minimal working example to implement a custom sliding mode control law for spatially distributed networked control systems.

empty. In order to simulate a spatially distributed networked control system using this custom control law, line 11 in listing 7.2 has to be replaced by

```
        NCS = NCS_ISMC(ncs_prob, 'ZeroSMC', 'tsim',30);
```

The custom control law will then automatically be used in the simulation.

7.3 The Classes and their Interaction

A central part of the simulation toolbox is the abstract class “NetworkNode”. This class is used to define a standard interface for implementing nodes that are connected to the network and used in SIMULINK “TRUETIME Kernels”. In the abstract method “init()”, TRUETIME tasks can be created and scheduled. Using a wrapper function which takes a function handle, it is possible to create and schedule TRUETIME tasks that call methods from the calling class instance. Figure 7.9 shows a class diagram that contains all classes derived from “NetworkNode” and their dependencies.

The class “SensorNode” creates a periodic task which calls the method “sample_states()” in every sampling step. In this method, a “NetworkMsg” object is created and sent to the receiver nodes. The “NetworkMsg” holds measurement samples of all states, the corresponding time stamp and a sequence number. In the simulations used in this dissertation, the sensor node sends the data to a “NetworkDelay” block (see figures 7.2 and 7.7).

The “NetworkDelay” and “NetworkBuffer” classes share the parent “VariableDelay” since both blocks introduce variable time delays until they forward the input packets to their receivers. The two classes differ in how they set the time delay. In the “NetworkDelay” class, the time delay is set randomly. In the “NetworkBuffer” class it is computed from the time stamp of the current packet to achieve the constant round trip time.

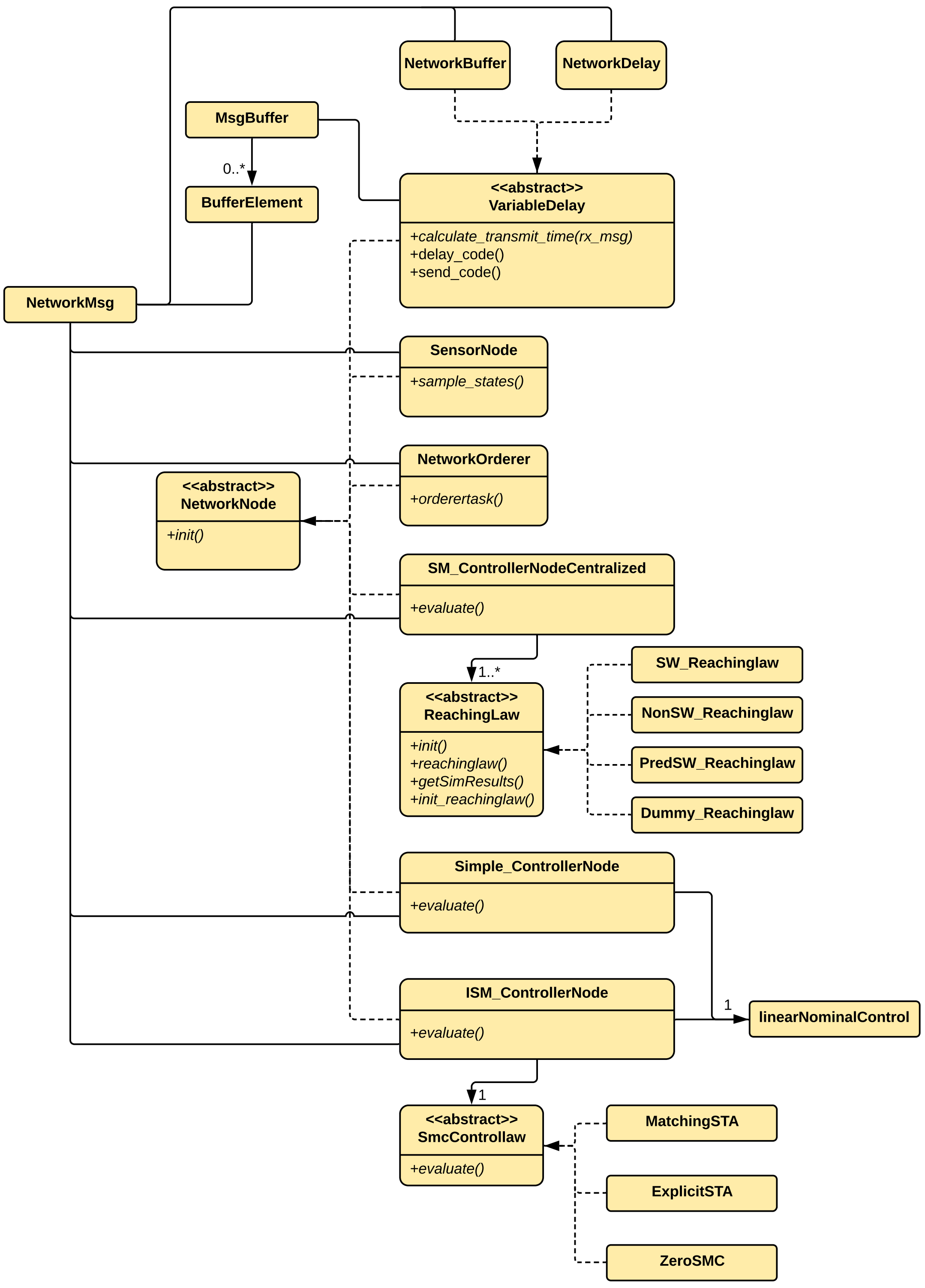


Figure 7.9: Class diagram for the node classes.

The class “NetworkOrderer” receives “NetworkMsg” objects from a “NetworkDelay” node and forwards these messages in the correct sequence. In detail, a message is only forwarded if all preceding messages have already been received.

The “Simple_ControllerNode” class implements a control law for spatially distributed networked systems that consists of the nominal part only, i.e. the sliding mode part is set to zero.

Sliding mode control laws for centralized networked systems based on the approaches proposed in chapter 5 are implemented in the “SM_ControllerNodeCentralized” class. An instance of this class holds an implementation of the abstract class “ReachingLaw” for each element of the sliding variable.

Integral sliding mode based control laws for spatially distributed networked systems based on the approaches proposed in chapter 6 are implemented in the “ISM_ControllerNode” class. Since the architecture is spatially distributed, an instance of this class holds only one implementation of the abstract class “SmcControllaw”.

Depending on the network structure (i.e. centralized or spatially distributed), different quantities of “NetworkNode” objects are necessary. The class diagram shown in figure 7.10 illustrates the dependency of the node classes on the network classes “NCS_Centralized” for centralized networked control systems and “NCS_ISMC” for spatially distributed networked control systems.

The centralized networked control systems need exactly one sensor node, two network delay nodes (one from the sensor node to the orderer node and one from the controller node to the buffer node), one orderer node, one controller node and one buffer node. These node quantities do not depend on the number of feedback channels.

For spatially distributed networked control systems, the required number of nodes depends on the number of feedback channels m. To be specific, m orderer and controller nodes, $2m$ delay nodes and one sensor node are necessary.

In the constructor of the classes “NCS_Centralized” and “NCS_ISMC”, these nodes are instantiated and configured accordingly. The SIMULINK blocks are configured by simply passing one of these objects.

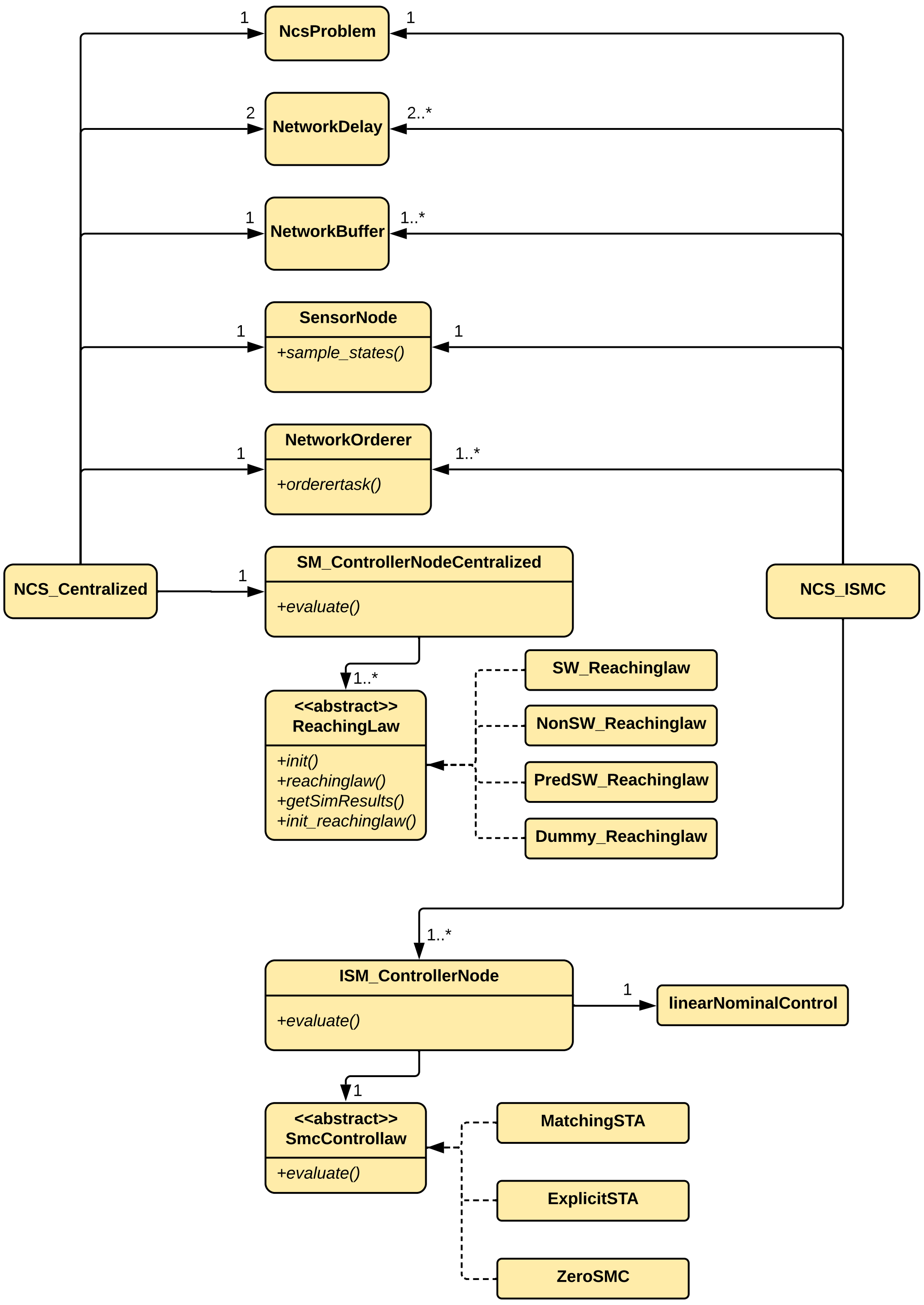


Figure 7.10: Class diagram for the network classes.

Chapter 8

Summary, Conclusions and Outlook

In the first part of this dissertation, a discrete time mathematical model for networked control systems is proposed. This model explicitly takes into account the time varying delays induced by the networked feedback. In order to cover spatially distributed networked control systems, different time delays for each feedback channel are considered. A spatially distributed network control system describes a system in which a dedicated controller node evolutes the control law for each feedback channel. Thus, the controller outputs are transmitted through several network channels. In contrast to that, centralized networked control systems are systems in which the control law is evaluated in one single controller node. As a consequence, the time delays for each feedback channel are the same since only one network channel is involved. In terms of modeling networked control systems, centralized networked control systems form a special case of spatially distributed networked control systems. Therefore, the proposed model covers both networked control system topologies.

Introducing a buffering mechanism which ensures constant round trip times from the sensor to the actuators results in the buffered networked control system. Due to the constant round trip times, a time-invariant mathematical model of the buffered networked control system can be used.

Using this time-invariant mathematical model, sliding mode based control laws for centralized buffered networked control systems are proposed. Due to the networked control setting, the sliding variable has to be designed following a specific structure. Due to this structure, classical sliding mode based reaching laws are not directly applicable. Hence, three reaching laws for centralized networked control systems are proposed. The parameter choices and achieved accuracies are discussed. The highest accuracy is achievable with the nonswitching reaching law followed by the predictive switching reaching law. The switching reaching law results in the poorest accuracy. Evaluations using a mechanical plant and considering first order parasitic actuator dynamics show that the predictive switching reaching law outperforms the nonswitching reaching law for practical applications even though the theoretically achievable accuracy is better with the nonswitching reaching law.

The sliding mode based control laws for spatially distributed networked control systems proposed in this dissertation also refer to the time-invariant mathematical model of the buffered networked control system. To design a control law for spatially distributed networked control systems, additional challenges have to be overcome. These are the different time delays for each feedback channel and the limited information as only local information is accessible (apart from the measurement data) in each controller node. Local information in this context means that each controller node only has access to the history of the own control signal since no communication between the controller nodes is available. These challenges are overcome using integral sliding mode control with a specific choice of the sliding variable and the nominal control law. The design process for the nominal control law involves solving linear matrix inequalities to fulfill the requirements. In addition, the specific choice of the sliding variable makes it possible to cast the problem in a form where classical sliding mode approaches are applicable. In this dissertation, two discrete time versions of the super twisting algorithm are considered, namely the explicit Euler discretized super twisting algorithm and the super twisting algorithm resulting from applying the matching approach. Also, the matching approach leads to better results in the networked setting since no discretization chattering appears when using this algorithm. Furthermore, the low sensitivity of this algorithm when choosing too large parameter values is verified for the networked control scenario in simulation and using a laboratory experiment.

In the last part of this dissertation, the developed simulation environment for buffered networked systems using the proposed algorithms is described. This simulation environment is developed with special focus on flexibility in terms of the ability to integrate custom control strategies with very little effort. Therefore, an interface for each of the two topologies, centralized and spatially distributed, are provided. As a result, this simulation environment is a good option to integrate and show the effectiveness of newly developed reaching laws or sliding mode based control laws in the networked control environment.

Since this work exclusively focuses on the networked induced time delays and neglects the influence of data loss, the perfect focus for further research would be to extend the proposed approaches to also deal with this type of network imperfection. In addition, time synchronization jitters between sensor and buffers or sampling time jitters are not considered in this work. Hence, these topics would be interesting to study in future research.

Bibliography

[1] J. Postel, “Transmission control protocol,” Internet Engineering Task Force, Request for Comments 793, Sep. 1981.

[2] J. Postel, “User datagram protocol,” Internet Engineering Task Force, Request for Comments 768, Aug. 1980.

[3] B. Azimi-Sadjadi, “Stability of networked control systems in the presence of packet losses,” in *42nd IEEE International Conference on Decision and Control*, Dec. 2003, pp. 676–681.

[4] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. S. Sastry, “Foundations of control and estimation over lossy networks,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 163–187, Jan. 2007.

[5] J. Xiong and J. Lam, “Stabilization of linear systems over networks with bounded packet loss,” *Automatica*, vol. 43, no. 1, pp. 80–87, Jan. 2007.

[6] K. Liu and E. Fridman, “Networked-based stabilization via discontinuous lyapunov functionals,” *International Journal of Robust and Nonlinear Control*, vol. 22, no. 4, pp. 420–436, Mar. 2012.

[7] K. Liu, E. Fridman, and L. Hetel, “Network-based control via a novel analysis of hybrid systems with time-varying delays,” in *51st IEEE Conference on Decision and Control*, Dec. 2012, pp. 3886–3891.

[8] G. P. Liu, “Predictive controller design of networked systems with communication delays and data loss,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 57, no. 6, pp. 481–485, Jun. 2010.

[9] W. Hu, G. Liu, and D. Rees, “Event-driven networked predictive control,” *IEEE Transactions on Industrial Electronics*, vol. 54, no. 3, pp. 1603–1613, Jun. 2007.

[10] M. C. F. Donkers and W. P. M. H. Heemels, “Output-based event-triggered control with guaranteed $\mathcal{L}_\infty$-gain and improved and decentralized event-triggering,” *IEEE Transactions on Automatic Control*, vol. 57, no. 6, pp. 1362–1376, Jun. 2012.

[11] J. Lunze and D. Lehmann, “A state-feedback approach to event-based control,” *Automatica*, vol. 46, no. 1, pp. 211–215, Jan. 2010.

[12] X. Zhang, Q. Han, and X. Yu, "Survey on recent advances in networked control systems," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 5, pp. 1740–1752, Oct. 2016.

[13] W. P. M. H. Heemels and N. van de Wouw, *Stability and Stabilization of Networked Control Systems.* Springer London, 2010, ch. 7, pp. 203–253.

[14] A. K. Behera and B. Bandyopadhyay, "Event-triggered sliding mode control for a class of nonlinear systems," *International Journal of Control*, vol. 89, no. 9, pp. 1916–1931, Jan. 2016.

[15] G. P. Incremona, A. Ferrara, and L. Magni, "Asynchronous networked MPC with ISM for uncertain nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 9, pp. 4305–4317, Sep. 2017.

[16] G. F. Franklin, M. L. Workman, and D. Powell, *Digital Control of Dynamic Systems*, 3rd ed. USA: Addison-Wesley Longman Publishing Co., Inc., 1997.

[17] R. Kalman, B. L. Ho, and N. Narendra, "Controllability of linear dynamical systems," *Contributions to Differential Equations*, vol. 1, pp. 198–213, Feb. 1963.

[18] J. Ludwiger, M. Steinberger, M. Rotulo, M. Horn, A. Luppi, G. Kubin, and A. Ferrara, "Towards networked sliding mode control," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, Dec. 2017, pp. 6021–6026.

[19] J. Ludwiger, M. Steinberger, M. Horn, G. Kubin, and A. Ferrara, "Discrete time sliding mode control strategies for buffered networked systems," in *2018 IEEE Conference on Decision and Control (CDC)*, Dec. 2018, pp. 6735–6740.

[20] J. Ludwiger, M. Reichhartinger, M. Steinberger, and M. Horn, "Discrete-time super twisting controller for networked control systems," *IFAC-PapersOnLine*, vol. 52, no. 16, pp. 789–794, Sep. 2019, 11th IFAC Symposium on Nonlinear Control Systems NOLCOS 2019.

[21] J. Ludwiger, M. Steinberger, and M. Horn, "Spatially distributed networked sliding mode control," *IEEE Control Systems Letters*, vol. 3, no. 4, pp. 972–977, Oct. 2019.

[22] M. B. G. Cloosterman, L. Hetel, N. van de Wouw, W. P. M. H. Heemels, J. Daafouz, and H. Nijmeijer, "Controller synthesis for networked control systems," *Automatica*, vol. 46, no. 10, pp. 1584–1594, Oct. 2010.

[23] A. Cervin, D. Henriksson, B. Lincoln, J. Eker, and K. Arzen, "How does control timing affect performance? Analysis and simulation of timing using Jitterbug and TrueTime," *IEEE Control Systems Magazine*, vol. 23, no. 3, pp. 16–30, Jun. 2003.

[24] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi, "The flooding time synchronization protocol," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, ser. SenSys '04. ACM, 2004, pp. 39–49.

[25] J. van Greunen and J. Rabaey, “Lightweight time synchronization for sensor networks,” in *Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications*, ser. WSNA ’03. ACM, 2003, pp. 11–19.

[26] U. Itkis, *Control systems of variable structure*, ser. A Halsted Press book. John Wiley & Sons, Incorporated, 1976.

[27] V. I. Utkin, *Sliding Modes in Control and Optimization.* Springer Berlin Heidelberg, 1992.

[28] V. Utkin, J. Guldner, and J. Shi, *Sliding Mode Control in Electro-Mechanical Systems, Second Edition.* CRC Press, May 2009.

[29] C. Edwards and S. Spurgeon, *Sliding mode control: theory and applications.* CRC Press, Aug. 1998.

[30] Y. Shtessel, C. Edwards, L. Fridman, and A. Levant, *Sliding Mode Control and Observation.* Springer New York, 2014.

[31] W. Gao, Y. Wang, and A. Homaifa, “Discrete-time variable structure control systems,” *IEEE Transactions on Industrial Electronics*, vol. 42, no. 2, pp. 117–122, Apr. 1995.

[32] J. Ackermann, “Der Entwurf linearer Regelungssysteme im Zustandsraum,” *Regelungstechnik*, vol. 20, pp. 297–300, 1972.

[33] J. Ackermann and V. Utkin, “Sliding mode control design based on Ackermann’s formula,” *IEEE Transactions on Automatic Control*, vol. 43, no. 2, pp. 234–237, Feb. 1998.

[34] R. A. Horn and C. R. Johnson, *Matrix Analysis*, 2nd ed. Cambridge University Press, 2012.

[35] J. Kautsky, N. K. Nichols, and P. van Dooren, “Robust pole assignment in linear state feedback,” *International Journal of Control*, vol. 41, no. 5, pp. 1129–1155, 1985.

[36] B. Drazenović, “The invariance conditions in variable structure systems,” *Automatica*, vol. 5, no. 3, pp. 287–295, May 1969.

[37] K. Abidi, J.-X. Xu, and Y. Xinghuo, “On the discrete-time integral sliding-mode control,” *IEEE Transactions on Automatic Control*, vol. 52, no. 4, pp. 709–715, Apr. 2007.

[38] A. Bartoszewicz, “Remarks on discrete-time variable structure control systems,” *IEEE Transactions on Industrial Electronics*, vol. 43, no. 1, pp. 235–238, Feb. 1996.

[39] A. Bartoszewicz, “Discrete-time quasi-sliding-mode control strategies,” *IEEE Transactions on Industrial Electronics*, vol. 45, no. 4, pp. 633–637, Aug. 1998.

[40] A. Levant, “Sliding order and sliding accuracy in sliding mode control,” *International Journal of Control*, vol. 58, no. 6, pp. 1247–1263, Dec. 1993.

[41] A. Levant, “Robust exact differentiation via sliding mode technique,” *Automatica*, vol. 34, no. 3, pp. 379–384, Mar. 1998.

[42] R. Seeber and M. Horn, “Stability proof for a well-established super-twisting parameter setting,” *Automatica*, vol. 84, pp. 241–243, Oct. 2017.

[43] M. Reichhartinger and S. Spurgeon, “An arbitrary-order differentiator design paradigm with adaptive gains,” *International Journal of Control*, vol. 91, no. 9, pp. 2028–2042, Feb. 2018.

[44] S. Koch and M. Reichhartinger, “Discrete-time equivalents of the super-twisting algorithm,” *Automatica*, vol. 107, pp. 190–199, Sep. 2019.

[45] M. Livne and A. Levant, “Proper discretization of homogeneous differentiators,” *Automatica*, vol. 50, no. 8, pp. 2007–2014, Aug. 2014.

[46] S. Koch, “Analysis and synthesis of discrete-time sliding mode controllers and observers,” Ph.D. dissertation, Graz University of Technology, 2019.

[47] S. Chakrabarty and A. Bartoszewicz, “Improved robustness and performance of discrete time sliding mode control systems,” *ISA Transactions*, vol. 65, pp. 143–149, Nov. 2016.

[48] A. Bartoszewicz and P. Latosinski, “Generalization of Gao’s reaching law for higher relative degree sliding variables,” *IEEE Transactions on Automatic Control*, vol. 63, no. 9, pp. 3173–3179, Sep. 2018.

[49] J. Daafouz and J. Bernussou, “Parameter dependent lyapunov functions for discrete time systems with time varying parametric uncertainties,” *Systems & Control Letters*, vol. 43, no. 5, pp. 355–359, Aug. 2001.