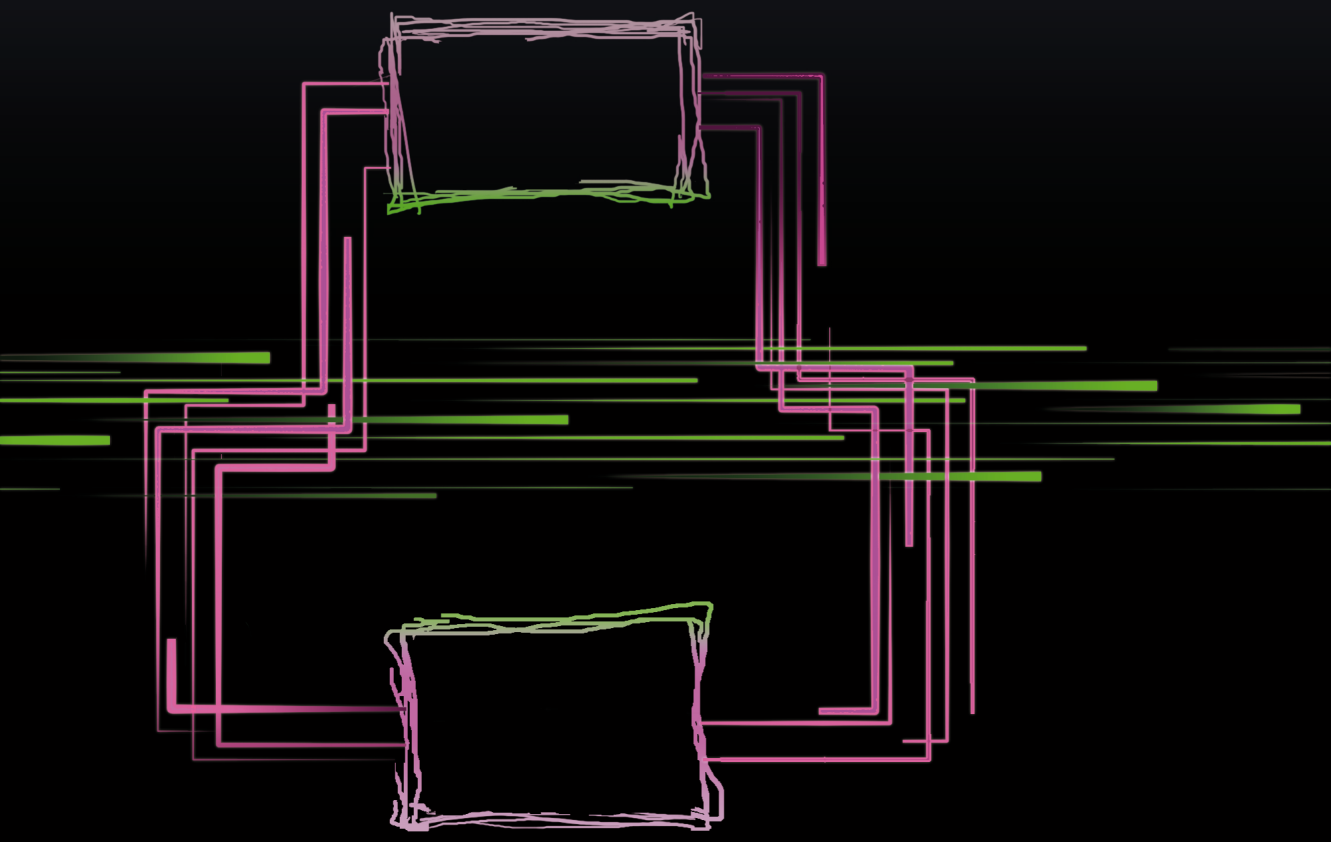
Networked Control Systems

From Theory to Experiments



Nick Bauer

Networked Control Systems
From Theory to Experiments

This work is supported by the Innovational Research Incentives Scheme under the VICI grant "Wireless control systems: A new frontier in automation" (no. 11382) awarded by NWO (Netherlands Organization for Scientific Research) and STW (Dutch Science Foundation), by the European 7th Framework Network of Excellence "Highly-complex and networked control systems (HYCON2)", under program number 257462, and by the European Community through the FP7-ICT-2007-2 thematic program under the WIDE-224168 project.

A catalog record is available from the Eindhoven University of Technology Library.
ISBN: 978-90-386-3322-0

Typeset by the author using LaTeX 2_ε

Reproduction: Ipskamp Drukkers, Enschede, the Netherlands.
Cover Design: N.W. Bauer

Networked Control Systems
From Theory to Experiments

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Technische Universiteit Eindhoven,
op gezag van de rector magnificus, prof.dr.ir. C.J. van Duijn,
voor een commissie aangewezen door het College voor Promoties
in het openbaar te verdedigen
op donderdag 7 februari 2013 om 16.00 uur

door

Nicolas William Bauer

geboren te Californië, Verenigde Staten van Amerika

Dit proefschrift is goedgekeurd door de promotor:

prof.dr.ir. W.P.M.H. Heemels

Copromotor:
dr.ir. N. van de Wouw

Contents

Chapter 1

Introduction

Many current challenges in the context of real-time feedback control require that the communication channels between sensors, controllers and actuators must be shared with other (unknown) applications. Sharing a communication channel naturally implies that network behavior related to properties that are crucial to real-time feedback control, such as the data rate, will be uncertain. The control community has realized that the sharing of communication resources and the uncertainty of the network undermines basic assumptions on which (traditional) feedback controllers rely. Therefore, the field of networked control systems (NCSs) is currently (further) developing theory and novel control algorithms which can guarantee control properties, such as stability and performance, in the presence of uncertain communication.

The sharing of communication channels is most common when the communication distance is large due to the high installation costs of each channel. Systems which span across such large distances are often desired to be controlled in a decentralized manner. Therefore, the combination of NCS theory and decentralized control theory emerges as a highly relevant research field and, forms the general topic that has been identified in this thesis. Before elaborating on the specific contributions of the thesis, in the next section we will provide a brief overview of how telecommunications advances have already enabled a variety of control relevant applications to solve modern-day challenges, and, highlight new control challenges for which current telecommunication advances do not offer a solution.

1.1 Motivation

The dawn of the smart phone has caused data demand to skyrocket, thereby, driving the telecommunications field to grow by leaps and bounds in an effort

to meet this demand. It is possible today to have a faster internet connection almost anywhere in the world via a hand-held smart phone with a 3G connection (14 Mbit/s) than just 20 years ago via a large PC with a dial-up modem connection (56 Kbit/s). These high-speed networks can easily be connected to and can, therefore, be used to create communication channels which can be, and have already been, exploited by a variety of control related applications such as wireless sensor networks (WSNs) and motion control.

WSNs have already been successful in exploiting *wireless* telecommunication advances in a variety of monitoring applications. These applications include forest fire detection [39, 70], air pollution monitoring [81], patient health monitoring [27] and structural monitoring [124], just to mention a few. WSNs are allowed to operate in the unlicensed 2.4GHz band, in which there are currently three major standards available: IEEE 802.11 for wireless local area networks (WLAN/WiFi), IEEE 802.15.1 for wireless personal area networks (WPAN/Bluetooth) and IEEE 802.15.4 for low-rate wireless personal area networks (LR-WPAN). Due to its low data rate, low power consumption and low cost, IEEE 802.15.4 is a very suitable candidate for battery-powered WSNs. Specifically, the IEEE 802.15.4 standard specifies the physical layer and the media access control (MAC) layer for LR-WPANs. The ZigBee, ISA100.11a, WirelessHART, and MiWi specifications are all different architectures (commonly referred to as 'stacks') built upon the IEEE 802.15.4 standard in an effort to create a general networking architecture flexible enough to handle a variety of potential applications. Due to the wide variety of applications, open-source operating systems such as TinyOS [75] and the WaspMote API have been developed by computer science communities to enable developers to easily interact with these devices and to contribute to the constantly growing software library.

Although motion control has not yet been able to (fully) exploit *wireless* technology, it has been able to successfully exploit *wired* telecommunication advances. A controller area network (CAN) bus [74], introduced in 1986, is a dedicated wired network architecture designed originally for automotive systems to cope with the increasingly large number of electrical systems within automobiles which must communicate in order to operate properly. Currently, like IEEE 802.15.4, the CAN standard has many custom architectures built upon it to serve many different application areas other than automotive automation, including industrial automation (DeviceNet, CANopen, SynqNet), aircraft and aerospace automation (CANaerospace, ARINC 825), building automation (VSCP) and many more. The reason for the popularity of the CAN standard is that, besides the benefit of huge reductions in wiring, it is highly reliable. CAN has the ability to guarantee a very high quality of service (QoS) level for control applications, in the sense that it guarantees time-critical control related data will have priority over other systems sharing the bus and that the latency is negligible.

However, in several new (motion) control challenges such a high QoS is not available, e.g. especially in wireless environments. One of these new control chal-

lenges regards the development of cooperative adaptive cruise control (CACC) for vehicles. The research community focusing on this approach studies the consequences of having vehicles wirelessly communicate in an effort to reduce traffic congestion by autonomously regulating the spacing between vehicles that are following each other [96, 97, 103]. It is obvious that communication between vehicles can only be implemented over a wireless network; therefore, the uncertainty inherent in wireless communication is unavoidable. This networked uncertainty forms an interesting challenge for such a real-time feedback control application since it may produce unsatisfactory performance or could even cause vehicles to collide if not dealt with appropriately. Another situation, where a high QoS of the network cannot be guaranteed, which is problematic for control applications, occurs when communication is via the internet. The uncertainty of internet traffic induces a significant variation in the data rate which limits traditionally designed controllers from achieving desired levels of performance [2, 66]. So, although in the past, the telecommunication advances were able to provide a high QoS for control applications (e.g. via a dedicated CAN bus), current challenges requiring more recent telecommunication advances (e.g. such as wireless networks and the internet) impose a necessity to (further) develop control theory in order to reliably use them in a real-time control context.

The ability of WSNs to still operate satisfactorily in monitoring applications can be attributed to the dynamics being slow in general where as the dynamics of control systems are typically faster (which is especially true for motion control systems). Hence, monitoring applications do not suffer as much from data-rate related issues as real-time control applications do. As experienced by every cell phone user, there are times when the data-rate varies and even completely drops out when using a wireless telecommunication network. In order for (motion) control applications to take advantage of the (globally) available high-speed data network provided by telecommunication advances, NCS theory must be (further) developed so that a fundamental understanding of control properties, such as stability and performance, in relation to properties of this (uncertain) communication medium can be obtained.

Other classes of new (motion) control challenges where a high QoS is not available involve the control of large-scale systems. Arguably the most significant advantage of current telecommunication networks is the ability to communicate across large (even interplanetary) distances. Installing dedicated wires across even hundreds of meters for a particular application becomes expensive and reduces the flexibility of the network layout. Moreover, implementing a centralized controller that must collect all control-related data at one central location in order to regulate systems which span across hundreds of meters is often impractical due to the long communication distances required. As an example, the EU-project WIDE, which supported the work in this thesis, focused on regulating a city-wide water distribution system that spans anywhere from a few hundred meters to tens of kilometers. Such an application benefits from

decentralized control implemented over communication networks. On the one hand, using existing telecommunication networks offers a technological solution for the expensive implementation (of the control strategies) based on installing dedicated wires between all control devices. On the other hand, decentralized regulation and coordination of this system is desirable for maintenance purposes since occasionally the city is forced to shut down one section of the system and it is desirable not to disrupt the operation of the rest of the system. Therefore, the combination of NCS theory and decentralized control theory emerges as a highly relevant research field, for which, a fundamental understanding is needed of the joint effect of the decentralization of controllers and the networking communication on system performance.

Summarizing, although the advances in telecommunication are steadily expanding the diameter and distance of the data pipe, the uncertainty of networked traffic worries and deters control engineers from using this communication medium for real-time feedback control applications. Trusty and reliable, wired point-to-point communication is what control engineers conventionally rely on. However, breaking free of the dependence on dedicated wiring for communication will solve many current challenges as well as open up many interesting possibilities, and, therefore forms a highly relevant research field. This thesis addresses four main challenges in this context:

(i.) the robust stability analysis of NCSs,

(ii.) robust decentralized controller synthesis for NCSs,

(iii.) software development aiming at making the tools for stability analysis and controller synthesis for NCSs accessible to both control engineers and theorists,

(iv.) experimental validation of network models and existing NCS stability analysis techniques.

In the remainder of this chapter, we will provide a general introduction to these four topics, specific details regarding the related research objectives and obtained contributions, and, an outline of the thesis. Detailed literature reviews can be found in the individual chapters.

1.2 Control via Networked Communication

Classical control theory relies on the fact that communication between sensors, actuators and controllers is ideal, meaning that data is communicated and computed with zero delay and infinite precision. Of course, due to physical limitations, any means of digital communication (and computation) induces a non-zero delay and finite precision. Nevertheless, this assumption is acceptable when the communication (and computation) hardware used for control is sufficiently fast compared to the dynamics. In fact, under this assumption, a vast litera-

ture on control theory is available which led to great advancements in automotive applications, commercial air transportation, electron microscope technology, high-tech manufacturing, satellite precision and reliability, space exploration and many more high-tech applications.

However, the assumption of sufficiently fast hardware can be (extremely) expensive to accommodate in practice, and in some situations, this assumption simply cannot be met. Two specific examples of applications in which this assumption can be only met at extreme cost are the regulation of a city-wide water distribution system and cooperative vehicle following, mentioned earlier. When the assumption on ideal communication (and computation) cannot be made, the classical theory is no longer applicable and there are basically two options. The first option is to purchase the most expensive hardware available to attain as close to ideal communication (and computation) as financially possible (which still may produce unsatisfactory behavior). The second option is to develop and use (NCS) theory to be able to specify under what conditions slower, less expensive, hardware can be used reliably in the sense of still guaranteeing proper closed-loop behavior. This thesis presents results which work towards making the second option possible by contributing towards the development of NCS theory.

Networked control theory, which studies the properties of control systems such as the one schematically depicted in Fig. 1.1, is built upon the realization that there exist many relevant real-time feedback control applications where communication between sensors, actuators and controllers is not always perfect. Specifically, several network-induced effects can be introduced in the closed-loop:

- the presence of a shared communication medium,
- varying sampling/transmission intervals,
- varying transmission delays,
- packet dropouts,
- quantization.

It is generally known that any of these phenomenon degrade performance and can even threaten closed-loop stability, see, e.g. [29, 63, 128]. Depending on the network and/or application, the influence/importance of each of these five effects can vary quite significantly. For example, systems which are able to transmit sensor data in one packet do not suffer from a shared communication medium but it might take longer to collect all the data and prepare the transmission of the packet, resulting in delays (latencies) being dominant. Another example is constituted by the case where sensors share a wired CAN bus, inducing network sharing between the sensors, but packet dropouts and delays rarely occur due to the high QoS of the CAN communication infrastructure. The final example we would like to present is the wireless stabilization of an experimental pendulum/cart setup described in Chapter 5 of this thesis in which all five network-induced effects are present simultaneously. Therefore, it is important

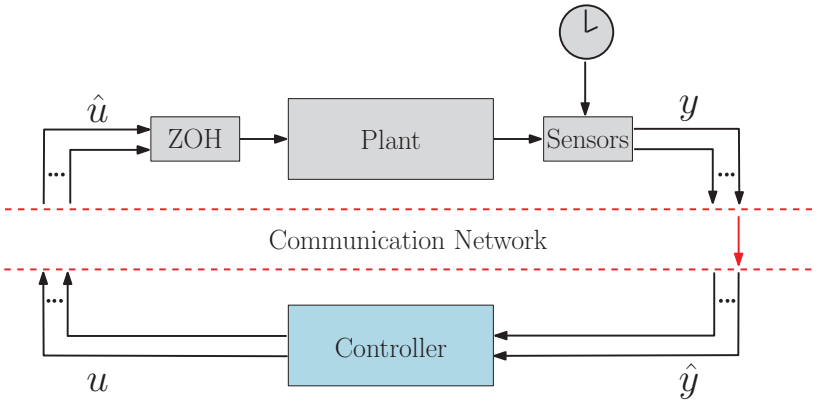


Figure 1.1: Block schematic of a networked control system.

to work with modeling and analysis frameworks which are general enough to include all five effects.

In fact, in this thesis, we present two such frameworks: in Chapter 2, we describe a framework based on jump-flow systems [52] and, in Chapter 3, we describe a framework based on discrete-time switched linear systems [76,82] with parametric uncertainty. These two frameworks are special cases within hybrid system theory [32, 51, 56], which one typically resorts to when analyzing NCSs subject to the mentioned network-induced imperfections. Hybrid system theory was created to understand systems with both continuous flow dynamics and discrete jump dynamics. Clearly, in an NCS the plant dynamics are continuous while the network acts in a discrete, packet-based, fashion. Thus, the tools of hybrid system theory can be used to determine specific properties of an NCS. In fact, advances in hybrid system theory have enabled the NCS community to relatively swiftly produce many breakthroughs over the past decade [12, 63, 72, 128].

Even though the two mentioned modeling and analysis frameworks for NCSs are capable of dealing with all five network-induced imperfections, in this thesis, we concentrate our attention on the first three effects. The first effect, the presence of a shared communication medium, requires a communication protocol which schedules the control related data. Although other authors have considered the case when the protocol is stochastic, see, e.g. [6, 120], we assume that the communication protocol is a deterministic process, as also assumed in [38, 58, 94, 122]. The next two effects, varying transmission intervals and varying transmission delays, are typically not considered to be deterministic and models for these effects come in different flavors in the NCS literature. Many authors have focused on the case where stochastic information is included in models for delays and transmission intervals via a probability dis-

tribution function [37, 63, 84, 114] or a Markov chain [112, 113]. In this thesis we focus our attention on the 'probability distribution free' case, as also studied in [28, 29, 38, 44, 58, 88, 94, 128]. In this case, the time-varying and uncertain transmission intervals and transmission delays are taken from a bounded set, without presuming any knowledge on the particular probability distribution of this bounded support. Of course, if a reliable and accurate stochastic model of the communication channels can be obtained, it is always more beneficial to use incorporate this information into the model and analysis. However, in general, an accurate stochastic model can be difficult to obtain, see, e.g. [9] and [72, p31-74]. In any case, having a variety of tools available which consider different network modeling assumptions is beneficial since networks exhibit very different behaviors depending on the network configurations and the network standards (e.g. 3G, WiFi, Bluetooth, CAN, etc.). For example, a 'well-behaved' dedicated network which can be accurately modeled can exploit the advances in stochastic NCS theory, while a more 'volatile' network subject to data traffic uncertainties does not have an accurate stochastic representation, and, can make use of the advances in the 'probability distribution free' case.

Besides the consideration of a variety of modeling assumptions of network-induced effects, the NCS literature contains many different results regarding different classes of plant and controller models. Currently, there are many existing results available on robust stability analysis of NCSs tailored to linear time-invariant (LTI) plants and LTI controllers. For example, linear static controllers were considered in [29, 44, 47, 91, 123], linear dynamic controllers were considered in [38, 122] and observer-based controllers were considered in [84]. Robust stability analysis methods, which not only admit linear plants and controller but also nonlinear plants and controllers, have been based on the jump-flow system framework in [58, 94, 95]. One main drawback of their approach is that it was shown to be conservative when applied to the special case of linear systems [38]. Therefore, it is important to further develop analysis techniques in the jump-flow system framework which preserve the rare ability to analyze nonlinear systems, while reducing the conservatism of the analysis. Chapter 2 describes a robust stability analysis procedure which has been shown to reduce the conservatism of the stability analysis within the jump-flow system framework.

1.3 Decentralized Control via Networked Communication

The decentralization of controllers offers many possible benefits, namely, a decrease in computation demand, an increase in the ease of reconfigurability, and an increase in robustness towards a centralized failure. Decentralized control is of particular interest for large-scale systems that are physically distributed over a wide area, thereby rendering the wiring of all sensors, controllers and actuators

expensive or impractical. Examples of such distributed systems are electrical power distribution networks [20], water transportation networks [23], industrial factories [85] and energy collection networks (such as wind farms [69,98]). Hence, creating analysis and design tools for decentralized controllers operating over (wireless) links is an important enabler for making decentralized control more realistically applicable.

Although the benefits of combining decentralized control with networked control (as depicted in Fig. 1.2) are clear, solving the controller synthesis and stability analysis problem, however, is extremely challenging. In addition to the challenges introduced by uncertain networked communication, decentralization, which has been studied for the past forty years, see, e.g. the early results [3,111], introduces different challenges and, still to this day, includes many open problems. Most notably, the problem of optimal decentralized controller synthesis still remains (mostly) open. Recently, the authors in [110] were able to characterize the set of decentralized problems which allow for convex synthesis of optimal static feedback controllers via a condition called quadratic invariance. Other recent challenges in decentralized control can be found in the survey [13] and the references therein.

Despite the challenges, both decentralized control and networked control have many useful theoretical tools available in the literature. Forging new theoretical tools that are based on combining the theoretical tools available in both of these fields and investigating their effectiveness serve as the first steps towards developing a more comprehensive theory leading to control structures for this increasingly relevant field. Working within the discrete-time switched linear systems framework, Chapter 3 is the result from our effort in this endeavor toward solving this complex but increasingly relevant problem.

1.4 NCS Analysis Software

The previous two sections motivated the development of novel theoretical tools for the analysis and synthesis of (decentralized) controllers which are communicating via shared networks. Just as important to developing such theory, is to make the results readily accessible for the control community by developing numerical tools that can easily analyze and synthesize NCSs. Despite the number of breakthroughs in the field of NCS [12,63,72,128], currently there does not exist a software toolbox so that engineers and theorists can easily apply the developed NCS theory. Although toolboxes which *analyze* NCSs currently do not exist, toolboxes which *simulate* NCSs have existed for almost a decade, see, e.g. [24]. Simulation toolboxes are useful because they can provide an accurate snapshot impression of how the NCS will behave, but due to the inherent uncertainties of NCSs, simulations can not provide any (long term) guarantees of control relevant properties. Therefore, analysis toolboxes that guarantee control relevant properties of NCSs are needed to provide control engineers with tools that can bestow

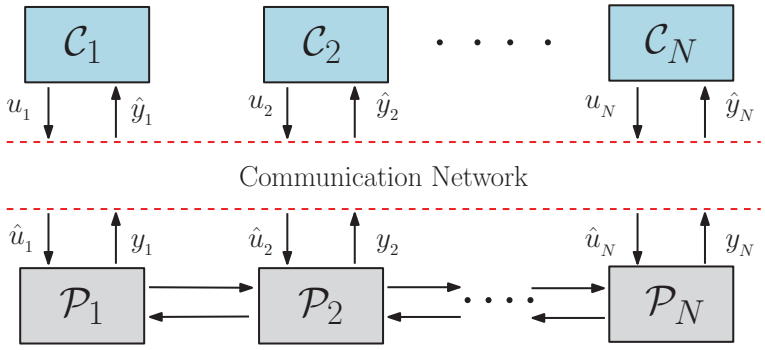


Figure 1.2: A decentralized NCS.

the peace of mind which comes with the guarantee of control relevant properties. To pioneer the development of a general NCS toolbox which can assess control relevant properties, Chapter 4 presents a prototype toolbox that has (amongst other benefits) automated the theoretical developments in [29, 38, 58, 79].

In addition to providing software which can analyze the network configurations addressed in the supporting literature, we would like our toolbox to be versatile enough to aid in the analysis of very general network configurations and the network standards (e.g. 3G, WiFi, Bluetooth, CAN, etc.). Therefore, the toolbox design process itself is a fruitful process since it forces the designers to think about how the theory can be generalized in a way that the (networked) control community finds useful. Specifically, through the development of this toolbox, we have been able to generalize the type of closed-loop NCS models which are able to be analyzed. Meaning that, in addition to being able to analyze the closed-loop models studied in the supporting literature, a user can also analyze custom closed-loop models with the toolbox (given the closed-loop model is able to be written in the presented general form). In this way, we are empowering the community with software tools which can assess control relevant properties in a variety of different network configurations. With the creation of this toolbox, we are advocating that the NCS community also starts making tools available which are able to not only reproduce published results, but also to be customizable. Chapter 4 provides the details regarding our effort to achieve this secondary goal.

1.5 Closing the Wireless Feedback Loop Experimentally

Once the tools for modeling and analysis of NCSs are available in software, the next important step is validating the theory via experimental studies. In the NCS setting, experiments are important for a number of reasons: (i) it provides insight into how real-life NCSs behave, (ii) it provides insight into which network modeling assumptions are valid for different practical network configurations and (iii) it provides an indication of the conservatism of the applied analysis/synthesis techniques. Linking developed theories to numerical tools and experimental setups creates a "research feedback loop" from which deeper insight can be gained.

Fortunately, it is currently becoming increasingly easier to close this feedback loop (i.e. building an experimental networked control setup) partially due to the success of WSNs. The TelosB [105] and the WaspMote are examples of versatile wireless devices that have open-source operating systems systems which have been developed by computer science communities to enable developers to easily interact with these devices and to contribute to the constantly growing software library. By using these open-source devices which are commercially available, the code developed as a result of the work done in this thesis can be directly implemented on compatible devices, thereby enabling other research groups to quickly build similar experimental setups to verify their own theory or use our developed software for analysis. In Chapter 5, we use TelosB devices to close the wireless feedback loop on an experimental inverted pendulum/cart setup.

1.6 Objectives and Contributions

It is important to develop techniques and tools that can be used to analyze and/or improve robustness properties of (decentralized) control systems which communicate via a shared communication network. Therefore, the research objectives addressed in this thesis are stated as follows:

- develop theoretical tools for stability analysis of NCSs,
- develop theoretical tools for net-aware decentralized controller synthesis,
- develop a software toolbox for the analysis of NCSs,
- validate the theory and tools on a wireless control experimental case study.

In the area of analyzing robustness properties of stability with respect to network-induced effects, a sum of squares (SOS) approach [73, 100, 101] for a class of nonlinear NCSs incorporating bounded time-varying delays, bounded time-varying transmission intervals and a shared communication medium will be developed in Chapter 2. Mathematical models that describe these nonlinear

NCSs will be provided and transformed into suitable hybrid systems formulations. Based on these hybrid systems formulations, Lyapunov functions will be constructed using SOS techniques that can be solved using linear matrix inequality (LMI)-based computational techniques. This will lead to several beneficial features: (i) plants and controllers which are described by piecewise polynomial differential equations can be analyzed, (ii) in contrast with various existing approaches for NCSs, non-zero lower bounds on the delays and transmission intervals can be incorporated, (iii) more flexibility in the Lyapunov functions is allowed, thereby obtaining less conservative estimates of the maximal allowable transmission intervals (MATI) and maximal allowable delay (MAD), and finally (iv) it provides an automated method to address stability analysis problems in nonlinear NCS.

Regarding the design of network-aware decentralized controllers guaranteeing robust stability properties with respect to network-induced effects, a systematic technique will be developed in Chapter 3. This technique offers an approach to the synthesis of decentralized observer-based output-feedback controllers for linear plants where the controllers, sensors and actuators are connected via a shared communication network subject to time-varying transmission intervals and delays. To effectively deal with the shared communication medium using observer-based controllers, we will adopt a switched observer structure that switches based on the received measured outputs and a switched controller structure that switches based on the received control inputs at each transmission time. By taking a discrete-time switched linear system perspective on modeling these decentralized NCSs, we will be able to derive a general model that captures all these networked and decentralized control aspects. The proposed synthesis method is based on decomposing the closed-loop model into a multi-gain switched static output-feedback form. This decomposition will allow for the formulation of LMI-based synthesis conditions which, if satisfied, provide stabilizing observer-based controllers, which are both decentralized and robust to network effects.

The developed theoretical contributions are crucial to obtain an improved understanding of tradeoffs in NCSs. However, developing tools which implement the theory is an equally important step towards making the results available for practical usage and dissemination. For this reason, Chapter 4 introduces the first prototype of a toolbox which was developed to automate (robust) stability analysis and controller design for NCSs. Specifically, it will be shown that the toolbox can be employed to efficiently verify if a linear time-invariant (LTI) plant and an LTI controller interconnected with a shared network are robust to certain network imperfections. We will explain how we aim to achieve the main intention of the toolbox, which is to make the available theory readily accessible to and applicable for the general control community. Additionally, it will be seen that the software structure enables the incorporation of custom models or custom stability/performance analysis conditions, enabling the control community to

contribute to the toolbox in an easy manner.

Finally, in Chapter 5, the experimental case study involving a wireless control system used to stabilize an inverted pendulum will be investigated. In particular, the communication network itself will be analyzed such that the network-induced effects can be characterized, in terms of bounds on the transmission intervals and transmission delays. With these bounds, the numerical toolbox discussed above will be applied. The analysis done will provide the robustness regions for different performance specifications, which aid in *tuning* the controller to achieve more closed-loop robustness with respect to the network-induced effects. This will be used to illustrate the effectiveness of the developed theory in an experimental setting.

1.7 Outline of Thesis

This thesis is divided into four main chapters. Each of the chapters is based on a research paper and is therefore self-contained with respect to the other chapters and chapters can be read independently.

Chapter 2: In this chapter, the stability analysis of NCSs based on sum of squares (SOS) techniques will be considered. The network-induced effects considered are time-varying transmission intervals, time-varying delays and a shared communication medium. This chapter is based on the journal paper [17], of which a preliminary version was presented in [16].

Chapter 3: In this chapter, the problem of decentralized observer-based controller synthesis for NCSs is considered using a discrete-time switched linear systems approach. The network-induced effects considered are time-varying transmission intervals, time-varying delays and a shared communication medium. This chapter is based on [14], of which an earlier version was presented in [15].

Chapter 4: In this chapter, the software toolbox aiding stability analysis for NCS is discussed. This chapter is based on [18].

Chapter 5: In this chapter, wireless control experiments on a cart/pendulum setup are provided and analyzed. The experiments were conducted to validate the recently developed theory.

Finally, conclusions will be drawn in Chapter 6 and recommendations for future research will be made.

Chapter 2

Stability Analysis of NCSs: A Sum of Squares Approach

"We think in generalities, but we live in details."
- Alfred North Whitehead

2.1 Introduction

Stability of networked control systems (NCSs) received considerable attention in recent years and several approaches are currently available for tackling this challenging problem. A distinction between the various approaches can be made on the basis of the modeling setup being adopted, the types of plants and controllers being allowed, and the network-induced imperfections being covered. In particular, the network-induced imperfections can roughly be divided into the following five categories: (i) quantization errors in the signals transmitted over the network due to the finite word length of the packets, (ii) packet dropouts caused by the unreliability of the network, (iii) variable sampling/transmission intervals, (iv) variable communication delays (being smaller or larger than the transmission interval) and (v) a shared communication medium imposing that only one node consisting of several actuators and sensors is allowed to transmit its information per transmission. The latter requires the presence of a network protocol that schedules which node gets access to the network at a transmission instant. In the analysis of NCSs exhibiting one or more of these networked-induced imperfections, several research lines can be distinguished.

A first line of research that can be distinguished is the *discrete-time modeling* approach, see e.g. [28, 29, 38, 47, 50, 59, 64, 115, 119, 123], which applies to linear plants and linear controllers and is based on exact discretization of the NCS between two transmission times. The resulting discrete-time linear model in which

the uncertain parameters such as the delay and transmission interval appear in an exponential form are overapproximated by polytopic models in which new uncertain parameters are introduced that appear in an affine form. These models are amendable for robust stability analysis methods based on linear matrix inequalities (LMIs). The most general discrete-time modeling approaches, in the sense of including the largest number of the above network-induced imperfections, are [28] including (ii), (iii) and (iv) (small and large delays), and [38] (ii), (iii) and (v) (only allowing so-called periodic and quadratic protocols), where (iv) with small delays can be easily included. Interestingly, in [38] the controllers can be both in a continuous-time or discrete-time form. Based on these models any of the available overapproximation techniques [59] can be applied to obtain polytopic models. In [59] a comparison is presented between the different overapproximation methods and the subsequent LMI-based stability analysis.

A second line of research on NCSs adopts the *sampled-data modeling* approach, which uses continuous-time models that describe the NCS dynamics in the continuous-time domain (so without exploiting any form of discretization) and perform stability analysis based on these sampled-data NCS models directly, see e.g. [43, 45, 125, 127]. The models are in the form of delay-differential equations (DDEs) and Lyapunov-Krasovskii-functionals are used to assess stability based on LMIs. In [83] it is shown that the use of such an approach for digital control systems neglects the piecewise constant nature of the control signal due to the zero-order-hold mechanism thereby introducing conservatism when exploiting such modeling for stability analysis. This spurred the development of an alternative approach, recently proposed in [90, 91], which is based on impulsive DDEs that do take into account the piecewise constant nature of the control signal. Constructive LMI-based stability conditions in the latter line of work apply for linear plants and linear controllers. These LMI conditions cannot exploit the availability of non-zero lower bounds on transmission intervals and delays in the sense that they always apply to the case where the minimum delay and transmission interval are equal to zero. This approach includes (ii)-(iv) as network-induced imperfections.

A third line of research is formed by the *continuous-time modeling* (or emulation) approaches, which are inspired by the work in [122], and extended in [22, 25, 58, 92, 94, 95] and the recent work [57] that includes all five of the mentioned communication imperfections albeit under some restrictive assumptions. To describe the NCS, this research line exploits the hybrid modeling formalisms as advocated in [51]. The stability of the resulting hybrid system model is based on Lyapunov functions constructed by combining separate Lyapunov functions for the network-free closed-loop system (which has to be designed to satisfy certain stability properties) on the one hand and the network protocol on the other hand (or, alternatively, adopting directly small gain arguments). General network protocols are allowed and the plant and controller in this context can also be of a general nonlinear form, but have to be given as continuous-time

differential equations. In addition $\mathcal{L}_p$ performance analysis [58, 94] can be easily incorporated in this framework. The available stability conditions all apply only for the case of zero lower bounds on the transmission intervals and delays.

In this chapter we propose an alternative computational method for stability analysis of NCSs, which from a modeling point of view is closest to the continuous-time modeling approach as just discussed, although it includes also the models based on impulsive DDEs [90, 91, 123], see Remark 2.4 in the main text. In particular, we will consider here NCSs that exhibit varying transmission intervals, varying delays and a shared communication medium. In fact, a preliminary version of our approach, applying to the case where the communication medium is not shared, is presented in [16]. In this extended version, we adopt rather general NCS models including a shared communication medium. These models will be converted into hybrid systems formulations [51]. Assuming piecewise polynomial plant dynamics (including piecewise affine systems), piecewise polynomial controller dynamics, and piecewise polynomial protocols (characterized more precisely below) which include the well-known Round Robin (RR) and Try-Once-Discard (TOD) protocols, Lyapunov functions can be constructed using sum of squares (SOS) tools [73, 100, 101]. As a result, this will lead to LMI-based tests for stability given bounds on the delays and transmission intervals. As such, this approach has the following beneficial features: (1) we can deal with nonlinear (piecewise) polynomial controllers and systems, while the constructive conditions in the discrete-time and sampled-data approach only can handle linear plants and controllers, (2) we can easily incorporate non-zero lower bounds on the transmission interval and delays, (3) we allow more flexibility in the Lyapunov functions thereby obtaining less conservative results than prior emulation-based approaches, see, e.g. [58, 94], (4) we can include performance analysis in terms of $\mathcal{L}_p$ gains of the continuous-time NCS, which is much harder for, for instance, the discrete-time approach due to the absence of intersample behavior in the discrete-time models, (5) we obtain an automated one-shot method to address stability analysis problems in nonlinear NCS, and (6) we do not have to discretize nor perform any polytopic overapproximations as in the discrete-time approach. Hence, the SOS-based stability analysis for NCSs offers many (different) advantages with respect to the various methods that already exist in the literature.

The rest of the chapter is organized as follows. In Section 2.2, we will present the general NCS model. Next in Section 2.3, we will provide the relevant theory for hybrid system stability and show how the stability conditions can be transformed into SOS problems. In particular, we present a procedure that can deal with the characteristics and peculiarities of the NCS models at hand. Finally in Section 2.4, four examples will demonstrate the strengths of the proposed approach.

2.2 NCS Model

In this section, we describe a general NCS model that is schematically depicted in Fig. 1.1. The specific model we consider here was discussed in [58], which extends earlier work [94], that was inspired by [122]. This model includes time-varying delays, time-varying transmission intervals and a shared communication medium. In addition, dropouts might be included by modeling them as prolongations of transmission intervals.

2.2.1 Description of the NCS

Consider the continuous-time plant

$$\dot{x}_p = f_p(x_p, \hat{u}), \qquad y = g_p(x_p), \tag{2.1}$$

in which $x_p \in \mathbb{R}^{n_p}$ denotes the state of the plant, $\hat{u} \in \mathbb{R}^{n_u}$ denotes the control values being implemented at the plant and $y \in \mathbb{R}^{n_y}$ is the output of the plant. The plant is controlled over a shared communication network by a controller, given by

$$\dot{x}_c = f_c(x_c, \hat{y}), \qquad u = g_c(x_c, \hat{y}), \tag{2.2}$$

where the variable $x_c \in \mathbb{R}^{n_c}$ is the state of the controller, $\hat{y} \in \mathbb{R}^{n_y}$ contains the most recent output measurements of the plant that are available at the controller and $u \in \mathbb{R}^{n_u}$ denotes the controller output. The presence of a communication network causes $u \neq \hat{u}$ and $y \neq \hat{y}$, as indicated in Fig. 1.1. In particular, the considered NCS setup assumes that the sensor acts in a time-driven fashion and that both the controller and the actuator act in an event-driven fashion (i.e. responding instantaneously to newly arrived data). The controller, sensors, and actuators are connected through a shared network subject to varying transmission intervals and varying delays. The fact that the network imposes a shared communication medium implies that not all sensor and actuator data can be transmitted simultaneously and scheduling by a network protocol is necessary.

Varying Transmission Intervals

At the transmission instants, $t_k \in \mathbb{R}_{\geq 0}$, $k \in \mathbb{N}$, the plant outputs and control values are sampled and sent over the network. The transmission instants t_k satisfy

$$t_k = \sum_{i=0}^{k-1} h_i \quad \forall k \in \mathbb{N}, \tag{2.3}$$

which are non-equidistantly spaced in time due to the time-varying transmission intervals $h_k := t_{k+1} - t_k$, with $h_k \in [h_{min}, h_{max}]$ for all $k \in \mathbb{N}$, for some $0 \leq h_{min} \leq h_{max}$. In the literature, see e.g. [58, 94, 122], where often $h_{min} = 0$, h_{max} is sometimes called the maximally allowable transmission interval (MATI).

We assume that the sequence of transmission instants $t_0, t_1, t_2, \ldots$ is strictly increasing in the sense that $t_{k+1} > t_k$, for all $k \in \mathbb{N}$ and $\lim_{k\to\infty} t_k = \infty$.

Varying Delays

The transmitted control values and measurement values are received after a delay $\tau_k \in \mathbb{R}_{\geq 0}$, with $\tau_k \in [\tau_{min}, \tau_{max}]$, for all $k \in \mathbb{N}$, where $0 \leq \tau_{min} \leq \tau_{max}$. More specifically, the sensor measurements received by the controller, denoted by $\hat{y}$, and the control commands received by the actuator(s), denoted by $\hat{u}$, can be expressed as $\hat{y}(t_k + \tau_k) = y(t_k)$ and $\hat{u}(t_k + \tau_k) = u(t_k)$, respectively. In the literature (e.g. [58], where $\tau_{min} = 0$), τ_{max} is sometimes called the maximal allowable delay (MAD). The delay is primarily due to the speed at which the data travels through the network. Given the speed of current communication hardware, it is realistic in several practical situations to assume that the delay is smaller than the transmission interval required for control purposes. As such, the following standing assumption is adopted to describe the admissible range of transmission intervals and delays.

Assumption 2.1. *The transmission intervals satisfy* $0 \leq h_{min} \leq h_k \leq h_{max}$ *and* $h_k > 0$ *for all* $k \in \mathbb{N}$ *such that* $\lim_{k\to\infty} t_k = \infty$*, and the delays satisfy* $0 \leq \tau_{min} \leq \tau_k \leq \min\{\tau_{max}, h_k\}$*,* $k \in \mathbb{N}$*.*

The latter condition implies that each transmitted packet arrives before the next sample is taken (i.e. the so-called small delay case). Hence, under Assumption 2.1, without loss of generality we can assume that $\tau_{max} \leq h_{max}$.

Remark 2.1. Note that the small-delay case can always be obtained in practice if the delays are bounded. This can be achieved by choosing the minimal transmission/sampling interval h_{min} such that Assumption 2.1 is satisfied. However, since increasing the minimal transmission interval might be undesirable from both a stability and performance perspective, we refer the reader to, e.g. the recent work [28, 29], for the case where 'large-delays' are considered (along with time-varying transmission intervals and dropouts), albeit in a different modeling framework. ◁

Remark 2.2. Here we consider a channel-independent (global) bound on the delay τ_k as considered in [58]. This implies that the delay bound is the same for every input and output channel. Although the extension toward channel-dependent delay bounds, i.e. $\tau_k^i \in [\tau_{min}^i, \tau_{max}^i]$, $i \in \{1, 2, ..., n_u + n_y\}$, can be envisioned within this framework, here we focus only on the channel-independent delay. The reason for doing so is to keep the main focus of the chapter on (i) detailing how the SOS conditions which can guarantee robust stability can be formulated, and, (ii) showing that the resulting robustness bounds can be improved compared to the previous work [58]. ◁

Shared Communication Medium

A shared communication medium prevents all control values and measurement values from being sent at each transmission time. A collection of sensors and/or actuators that are allowed to transmit their data simultaneously over the network is called a node. In practical situations each node typically corresponds to either a collection of sensors or a collection of actuators. At each transmission time t_k, $k \in \mathbb{N}$, a protocol determines which node $\sigma_k \in \{1, 2, \ldots, N\}$ is granted access to the network. We will use the left-continuous piecewise constant function $\sigma : \mathbb{R}_{\geq 0} \to \{1, ..., N\}$ given by

$$\sigma(t) = \sigma_k, \ t \in (t_k, t_{k+1}], \ k \in \mathbb{N}, \tag{2.4}$$

such that $\sigma(t), t \in \mathbb{R}_{\geq 0}$, is the node that was granted access at the most recent transmission instant before time t. The sensors/actuators corresponding to the node, which are granted access, collect their corresponding values in $y(t_k)$ or $u(t_k)$ that will be sent over the communication network. They will arrive after a transmission delay of τ_k time units, i.e. at time $t_k + \tau_k$.

To model this setup properly, we will denote the latest information available at the plant as $\hat{u}$ and the latest information available at the controller as $\hat{y}$. In addition, we will define the network-induced errors $e_y = \hat{y} - y$ and $e_u = \hat{u} - u$ to describe the difference between what is the most recent information that is available at the controller/plant and the current value of the plant/controller output, respectively. Sometimes we will write $e = (e_u, e_y) \in \mathbb{R}^{n_e}$, where we use the notation $(e_y, e_u) := [e_y^\top, e_u^\top]^\top$ and $n_e = n_y + n_u$. In between the updates of $\hat{y}$ and $\hat{u}$, the network is assumed to operate in a zero-order-hold (ZOH) fashion, meaning that the values of $\hat{y}$ and $\hat{u}$ remain constant in between the updating times $t_k + \tau_k$ and $t_{k+1} + \tau_{k+1}$. At times $t_k + \tau_k$, $k \in \mathbb{N}$, the updates satisfy

$$\hat{y}((t_k + \tau_k)^+) = y(t_k) + h_y(\sigma(t_k), e(t_k)) \tag{2.5a}$$

$$\hat{u}((t_k + \tau_k)^+) = u(t_k) + h_u(\sigma(t_k), e(t_k)). \tag{2.5b}$$

The functions h_y and h_u are update functions that are related to the protocol, which will be explained in more detail in the next paragraph. Based on (2.5) and the ZOH assumption, the network-induced error e_y for y behaves at the update times $t_k + \tau_k$ according to

$$\begin{aligned} e_y((t_k + \tau_k)^+) &= \hat{y}((t_k + \tau_k)^+) - y(t_k + \tau_k) \\ &= y(t_k) + h_y(\sigma(t_k), e(t_k)) - y(t_k + \tau_k) \\ &= h_y(\sigma(t_k), e(t_k)) + \underbrace{y(t_k) - \hat{y}(t_k)}_{-e_y(t_k)} + \underbrace{\hat{y}(t_k + \tau_k) - y(t_k + \tau_k)}_{e_y(t_k+\tau_k)} \\ &= h_y(\sigma(t_k), e(t_k)) - e_y(t_k) + e_y(t_k + \tau_k), \end{aligned}$$

where we used that $\hat{y}(t_k) = \hat{y}(t_k + \tau_k)$, which holds due the ZOH assumption. A similar derivation holds for the network-induced error e_u. Combining h_u and

h_y into one function h as $h = (h_u, h_y)$ leads to the network-induced error model

$$e((t_k + \tau_k)^+) = h(\sigma(t_k), e(t_k)) - e(t_k) + e(t_k + \tau_k). \tag{2.6}$$

Remark 2.3. Note that here we adopt a ZOH assumption between updates, as it is the standard convention for digital-to-analog signal conversion. Alternatively, a generalized (possibly model-based) hold which predicts future control values can be implemented at the actuators and a model-based estimator can be implemented at the controller, as in e.g. [25, 106]. ◁

To provide insight into how the protocol h operates, let us consider the two most well-known protocols being the Round Robin (RR) protocol and the Try-Once-Discard (TOD) protocol. To introduce them, the error vector e will be partitioned as $e = (e_1, e_2, \ldots, e_N)$ according to the nodes and the function h will be partitioned similarly as $h = (h_1, ..., h_N)$, where N is the number of nodes. In many situations, including the RR and TOD protocols, the protocol function h is given as $h(\sigma, e) = \tilde{h}(\delta(\sigma, e), e)$, where $\tilde{h}(j, e)$ describes how the updates in (2.5) take place when node j gets access to the network, and $\delta : \{1, ..., N\} \times \mathbb{R}^{n_e} \to \{1, ..., N\}$ describes which node is given access. In particular, which node $\sigma_k \in \{1, ..., N\}$ gets access to the network at the transmission time t_k is based on the previous node $\sigma(t_k)$ that got access at t_{k-1}, and the network-induced error $e(t_k)$ at t_k. Although other choices for $\tilde{h}$ are possible (see e.g. Example 2 in [95]), in most cases $\tilde{h}$ is defined as

$$\tilde{h}_i(j, e) := \begin{cases} 0, & \text{if } i = j, \\ e_i, & \text{otherwise} \end{cases} \tag{2.7}$$

for $i, j \in \{1, 2, \ldots, N\}$ and $e \in \mathbb{R}^{n_e}$. Hence, $\tilde{h}$ expresses that when node j gets network access, the outputs y_i that belong to node j satisfy $\hat{y}_i((t_k + \tau_k)^+) = y_i(t_k)$, according to (2.5), while the outputs not corresponding to node j are not updated. Similar statements hold, of course, for the control inputs u. This indicates that the values of $\hat{y}$ and $\hat{u}$ corresponding to the node that gets access are updated at time $t_k + \tau_k$ to the values of y and u at time t_k, which is natural for many practical cases. Using the definition of $\tilde{h}$ as in (2.7), the RR protocol is now specified by $h(\sigma, e) = \tilde{h}(\delta_{RR}(\sigma, e), e)$ with $\delta_{RR}(\sigma, e)$ defined for $\sigma \in \{1, ..., N\}$ and $e \in \mathbb{R}^{n_e}$ by

$$\delta_{RR}(\sigma, e) := (\sigma \ mod \ N) + 1 \tag{2.8}$$

where *mod* denotes the modulo operator. For the TOD protocol, $h(\sigma, e) = \tilde{h}(\delta_{TOD}(\sigma, e), e)$ with $\delta_{TOD}(\sigma, e)$ defined for $\sigma \in \{1, ..., N\}$ and $e \in \mathbb{R}^{n_e}$ by

$$\delta_{TOD}(\sigma, e) := \arg\max |e_j|, \quad j = 1, ..., N. \tag{2.9}$$

where $|\cdot|$ denotes the Euclidian norm. Hence, the RR protocol grants each node access to the network in a periodic fashion, whereas the TOD protocol grants access to whichever node has the largest network-induced error $|e_i(t_k)|$ at

transmission time $t_k, k \in \mathbb{N}$. In the case where two nodes have the largest error magnitude, one node is chosen arbitrarily.

The problem that we aim to solve in this chapter is to determine stability of the NCS given the protocol, such as the RR or TOD protocol, and bounds $h_{min}, h_{max}, \tau_{min}$ and τ_{max} as in Assumption 2.1, or to determine these bounds such that stability is guaranteed.

2.2.2 Hybrid System Formulation

To facilitate the stability analysis, the NCS model is transformed into a hybrid system [51], [58] of the form

$$\dot{\xi} = F(\xi), \quad \xi \in C, \tag{2.10a}$$

$$\xi^+ = G(\xi), \quad \xi \in D, \tag{2.10b}$$

where C and D are subsets of $\mathbb{R}^{n_\xi}$, $F : C \to \mathbb{R}^{n_\xi}$ and $G : D \to \mathbb{R}^{n_\xi}$ are mappings and ξ^+ denotes the value of the state directly after the reset. We denote the hybrid system (2.10) in brief sometimes by its data (C, D, F, G).

To transform the NCS setup given by (2.1)-(2.2) and (2.5) into (2.10), the auxiliary variables[1] $s \in \mathbb{R}^{n_e}$, $\tau \in [0, h_{max}]$, $\sigma \in \{1, ..., N\}$ and $\ell \in \{0, 1\}$ are introduced to reformulate the model in terms of so-called flow equations (2.10a) and reset equations (2.10b). The variable s is an auxiliary variable containing the memory storing the value $h(\sigma(t_k), e(t_k)) - e(t_k)$ at t_k for the update of e at the update instant $t_k + \tau_k$ as in (2.6), τ is a timer to keep track of how much time elapsed since the last transmission event, σ denotes the latest node that got access to the network, as in (2.4), and ℓ is a Boolean keeping track whether the next event is a transmission event or an update event. To be precise, when $\ell = 0$ the next event will be related to transmission (at times $t_k,\ k \in \mathbb{N}$) and when $\ell = 1$ the next event will be an update (at times $t_k + \tau_k,\ k \in \mathbb{N}$).

The state of our hybrid system Σ_{NCS} is chosen as $\xi = (x, e, s, \tau, \sigma, \ell) \in \mathbb{R}^{n_\xi}$, where $x = (x_p, x_c)$. The continuous flow map F can now be defined as

$$F(\xi) := (f(x, e), g(x, e), 0, 1, 0, 0), \tag{2.11}$$

where f, g are appropriately defined functions depending on f_p, g_p, f_c and g_c. See [94] for the explicit expressions of f and g. The state s is specified as a memory variable, therefore, $\dot{s} = 0$. The state τ is a timer variable and thus $\dot{\tau} = 1$. Finally, the states σ and ℓ are integer valued, which implies $\dot{\sigma} = \dot{\ell} = 0$. Flow according to $\dot{\xi} = F(\xi)$ occurs when the state ξ lies in the flow set

$$\begin{aligned} C \ := \quad \{\xi \in \mathbb{R}^{n_\xi} \mid \ & (\ell = 0 \wedge \tau \in [0, h_{max}]) \\ & \vee (\ell = 1 \wedge \tau \in [0, \tau_{max}])\}, \end{aligned} \tag{2.12}$$

[1] Here we slightly divert from the models as used in [58, 94, 122] where the hybrid system had the state $\kappa \in \mathbb{N}$, which indicated the number of the transmission, instead of $\sigma \in \{1, ..., N\}$. This modification is more convenient for our purposes of stability analysis based on sum of squares.

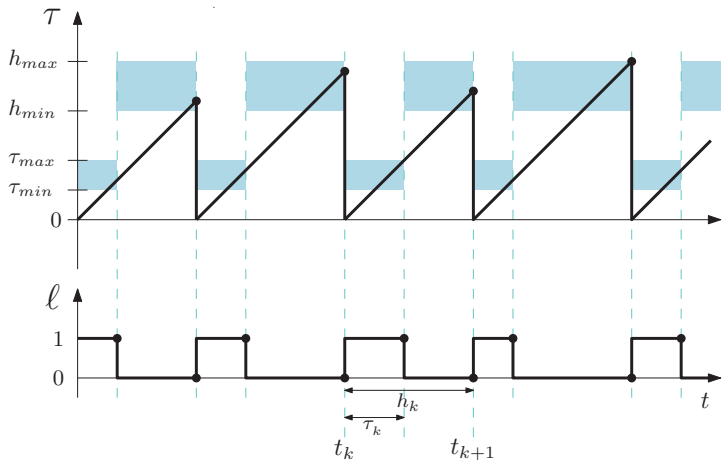


Figure 2.1: One possible evolution of the states τ and ℓ. This visually conveys how these states are used in the definition of the sets C and D (given in (2.12) and (2.15), respectively) in order to model transmissions (at times t_k, $k \in \mathbb{N}$), i.e. applying (2.13), and receptions/updates (at times $t_k + \tau_k$, $k \in \mathbb{N}$), i.e. applying (2.14), according to Assumption 2.1.

where $\wedge$ denotes the logical 'and' operator and $\vee$ denotes the logical (non-exclusive) 'or' operator. The jump map G inducing resets

$$(x^+, e^+, s^+, \tau^+, \sigma^+, \ell^+) = G(x, e, s, \tau, \sigma, \ell),$$

is obtained by combining the "transmission reset relations," active at the transmission instants $\{t_k\}_{k\in\mathbb{N}}$, and the "update reset relations", active at the update instants $\{t_k+\tau_k\}_{k\in\mathbb{N}}$. Using (2.6), the jump map G is defined at the transmission resets (when $\ell = 0$) as

$$G(x, e, s, \tau, \sigma, 0) := (x, e, h(\sigma, e) - e, 0, \delta(\sigma, e), 1) \tag{2.13}$$

and the update resets (when $\ell = 1$) as

$$G(x, e, s, \tau, \sigma, 1) := (x, s + e, 0, \tau, \sigma, 0). \tag{2.14}$$

Note that when $\ell = 1$ the update s^+ can be chosen arbitrarily since the state variable s does not influence the future evolution of other state variables. We choose $s^+ := 0$ for convenience. The jump map G is allowed to reset the system when the state is in the jump set

$$\begin{aligned} D \;:=\; \{\xi \in \mathbb{R}^{n_\xi} \mid\; & (\ell = 0 \wedge \tau \in [h_{min}, h_{max}]) \\ & \vee (\ell = 1 \wedge \tau \in [\tau_{min}, \tau_{max}])\}. \end{aligned} \tag{2.15}$$

Note that the sets C and D are defined by a combination of the state variables ℓ and τ. Moreover, notice that in (2.13) and (2.14), ℓ changes its value from 0 to 1 and vice versa, respectively, and τ is reset to 0 only in (2.13). Due to this combination of ℓ and τ and Assumption 2.1, only one timer variable is needed for constraining both delays and transmission times. Fig. 2.1 is provided to help visualize this explanation.

Finally, we define the equilibrium set of the hybrid system

$$\mathcal{A} := \{\xi \in C \cup D |\ x = 0 \wedge e = s = 0\}, \tag{2.16}$$

for which we would like to prove stability. Hence, the informal stability problem phrased at the end of Section 2.2.1 translates now to the question of determining global asymptotic stability (GAS) of the set $\mathcal{A}$ for $\Sigma_{NCS} := (C, D, F, G)$ (see [51] for exact definitions of global asymptotic stability of sets). For the remainder of the chapter, we will define

$$\chi := (x, e, s) \in \mathbb{R}^{n_\chi}, \tag{2.17}$$

as this is the part of the state vector for which we would like to show convergence to zero.

Remark 2.4. The sampled-data system as considered in [89], which lumped the sensor-controller and controller-actuator delays into one delay, was modeled as an impulsive delay-differential equation and focused on linear plant dynamics $\dot{x}_p = Ax_p + B\hat{u}$ with system matrix A, input matrix B and state feedback controllers of the form $u = -Kx_p$ (i.e. $y = x_p = x$). This model can also be expressed in this hybrid framework by omitting σ, e_u and x_c and taking $f(x, e) = (A - BK)x - BKe$, $g(x, e) = (-A + BK)x + BKe$ and $h(\sigma, e) = 0$. Note that, $h(\sigma, e) = 0$ implies that (2.5) simplifies to $\hat{y}((t_k + \tau_k)^+) = y(t_k)$. $\lhd$

2.3 Stability Analysis

In this section, we will show how the set $\mathcal{A}$ of the hybrid NCS model Σ_{NCS} can be shown to be GAS by exploiting SOS techniques. We will first state some fundamental hybrid system stability results relevant to our purposes and then present the corresponding SOS theorems, which will be exploited to set up SOS-based stability conditions for the presented NCS model.

2.3.1 Stability of Hybrid Systems

In this section, we will briefly summarize the relevant stability theory for a hybrid system (2.10), as detailed in [51, 52]. First, we will formalize what is meant by GAS for a compact set $\mathcal{A}$ of a hybrid system (2.10). Then we will provide the relevant theorem on how GAS can be proven via Lyapunov-based arguments. Before formalizing the GAS property, we define the distance of a vector ξ to the set $\mathcal{A}$ as $|\xi|_{\mathcal{A}} := \inf_{\xi' \in \mathcal{A}} |\xi - \xi'|$.

Definition 2.1. *[51] A compact set $\mathcal{A} \subset \mathbb{R}^{n_\xi}$ of a hybrid system $\Sigma = (C, F, D, G)$, for which every solution exists for all times $t \in [0, \infty)$, is globally asymptotically stable (GAS) if*

(i.) for each $\epsilon > 0$ there exists a $\delta > 0$ such that $|\xi(0)|_{\mathcal{A}} \leq \delta$ implies $|\xi(t)|_{\mathcal{A}} \leq \epsilon$ for all solutions ξ of Σ and all $t \in \mathbb{R}_{\geq 0}$ (i.e. $\mathcal{A}$ is stable),

(ii.) for any $\xi(0) \in \mathbb{R}^{n_\xi}$ it holds that $|\xi(t)|_{\mathcal{A}} \to 0$ as $t \to \infty$ (i.e. $\mathcal{A}$ is globally attractive).

Next, we will use the following definition to specify a Lyapunov function candidate $V : \text{dom } V \to \mathbb{R}$, with dom $V \subseteq \mathbb{R}^{n_\xi}$, for a hybrid system as in (2.10). We will use the concept of a sublevel set of V on a subset Ξ of dom V, which is a set of the form $\{\xi \in \Xi \mid V(\xi) \leq c\}$ for some $c \in \mathbb{R}$.

Definition 2.2. *[51] Consider a hybrid system $\Sigma = (C, D, F, G)$ and a compact set $\mathcal{A} \subset \mathbb{R}^{n_\xi}$. The function $V : dom\, V \to \mathbb{R}$, with $C \cup D \subseteq dom\, V$, is a Lyapunov function candidate for $(\Sigma, \mathcal{A})$ if*

(i.) V is continuous and nonnegative on $(C \cup D) \backslash \mathcal{A} \subset dom V$,

(ii.) V is continuously differentiable on an open set $\mathcal{O}$ satisfying $C \backslash \mathcal{A} \subset \mathcal{O} \subset dom\, V$,

(iii.)

$$\lim_{x \to \mathcal{A}, x \in dom\ V \cap (C \cup D)} V(x) = 0,$$

(iv.) the sublevel sets of V on dom $V \cap (C \cup D)$ are compact.

Finally, to prove GAS of the set $\mathcal{A}$, we will make use of the following theorem, where we use the notation $\langle \cdot, \cdot \rangle$ to denote the standard inner product in Euclidean spaces.

Theorem 2.1. *Consider a hybrid system $\Sigma = (C, F, D, G)$ and a compact set $\mathcal{A} \subset \mathbb{R}^{n_\xi}$ satisfying $G(D \cap \mathcal{A}) \subset \mathcal{A}$. If every solution of Σ exists for all times $t \in [0, \infty)$ and there exists a Lyapunov function candidate V for $(\Sigma, \mathcal{A})$ that satisfies Definition 2.2 and*

$$\langle \nabla V(\xi), F(\xi) \rangle < 0 \quad \text{for all } \xi \in C \backslash \mathcal{A} \tag{2.18}$$

$$V(G(\xi)) - V(\xi) \leq 0 \quad \text{for all } \xi \in D \backslash \mathcal{A}, \tag{2.19}$$

then the set $\mathcal{A}$ is GAS.

Proof The proof can be based on [51].

2.3.2 Stability Using SOS Techniques

Constructing suitable Lyapunov functions to prove stability is known to be a hard problem, certainly in the nonlinear and hybrid context. Here, we provide

a computational approach to this problem based on polynomial Lyapunov functions and sum of squares (SOS) techniques [26,60,73,99–101]. To formally define a SOS we use the following definition, in which we use the notation $\mathbb{R}[x_1, ..., x_n]$ to denote the set of polynomials in n variables $x_1, ..., x_n$ with real coefficients.

Definition 2.3. *A polynomial $p \in \mathbb{R}[x_1, ..., x_n]$ is called a sum of squares if there exist $m \in \mathbb{N}$ and polynomials $p_1, p_2, ..., p_m \in \mathbb{R}[x_1, ..., x_n]$ such that $p(x) = \sum_{i=1}^{m} p_i^2(x)$ for all $x \in \mathbb{R}^n$.*

Based on this definition, which implies that sum of squares are always non-negative, inequalities, as in (2.18)[2] and (2.19), can be guaranteed to be true if their left-hand sides can be expressed as sums of squares (where S-procedure like relaxations [21] can be used to incorporate the regional information $\xi \in C \backslash \mathcal{A}$ in (2.18) and $\xi \in D \backslash \mathcal{A}$ in (2.19)). The appeal of SOS is that the solution can be computed using convex semidefinite programming techniques. Indeed $p(x) = \sum_{i=1}^{m} p_i^2(x)$ can be checked by finding a positive semidefinite matrix Q, and a vector of monomials $Z(x)$ such that $p(x) = Z^\top(x) Q Z(x)$ for all x. If a positive semidefinite matrix Q is found, we can use Choleski factorization to obtain $Q = L^\top L$ for some matrix L and thus write $p(x)$ as a sum of squares $p(x) = Z^\top(x) L^\top L Z(x) = |LZ(x)|^2 = \sum_i (L_i Z(x))^2$, where L_i denotes the i^{th} row of L, see e.g. [26, 73, 101] for more details.

In the context of stability of hybrid systems (2.10), when F and G are piecewise polynomial functions (which in the case of the NCS models presented earlier, is true when f_c, f_p, h are piecewise polynomial and g_c, g_p are polynomial) on their domains C and D, the Lyapunov stability conditions in Theorem 2.1 can be transformed into a set of polynomial inequalities. To formalize this idea, we provide the following two definitions.

Definition 2.4. *A set $\mathcal{B}$ is called a basic semialgebraic set if it can be described as*

$$\mathcal{B} = \{\ x \in \mathbb{R}^n \mid b_j(x) \geq 0,\ j = 1, ..., n_{\mathcal{B}} \ \text{and}\ \bar{b}_l(x) = 0,\ l = 1, ..., \bar{n}_{\mathcal{B}}\}$$

for certain polynomials $b_j \in \mathbb{R}[x_1, ..., x_n]$, $j = 1, ..., n_{\mathcal{B}}$, and $\bar{b}_l \in \mathbb{R}[x_1, ..., x_n]$, $l = 1, ..., \bar{n}_{\mathcal{B}}$.

Definition 2.5. *A function $p : \Omega \to \mathbb{R}$ with $\Omega \subseteq \mathbb{R}^n$ is called piecewise polynomial if there are M basic semialgebraic sets $\Omega_1, ..., \Omega_M$ and polynomials $p_1, ..., p_M \in \mathbb{R}[x_1, ..., x_n]$ such that*

$$(i)\ \ \Omega = \bigcup_{i=1}^{M} \Omega_i,$$

$$(ii)\ \ \forall x \in \Omega \text{ there exists an } i \in \{1, ..., M\} \text{ such that } p(x) = p_i(x) \text{ and } x \in \Omega_i.$$

[2]Notice here that (2.18) is a strict inequality and requires special treatment, as done later.

Remark 2.5. Note that in this definition the sets $\Omega_1, ..., \Omega_M$ may overlap, which is convenient as basic semialgebraic sets are closed and are not given by strict inequalities. Basically stability of the hybrid system will be proven no matter which $p_i(x)$ (e.g. in the flow or jumps of (2.10)) will be chosen in regions that overlap. As such, also the arbitrariness in the TOD protocol (i.e. when two nodes have the same largest networked induced error, see (2.9) and the discussion after (2.9)) can be incorporated. ◁

To apply SOS techniques to the hybrid model (2.10), $F : C \to \mathbb{R}^{n_\xi}$ and $G : D \to \mathbb{R}^{n_\xi}$ need to be piecewise polynomial as in Definition 2.5. The sets C and D can then be expressed as $C = \cup_{i=1}^{I} C_i$ and $D = \cup_{m=1}^{M} D_m$ with $C_i, i = 1, ..., I$, and $D_m, m = 1, ..., M$, basic semialgebraic sets, meaning that

$$\begin{aligned} C_i \quad = \{\xi \in \mathbb{R}^{n_\xi} \mid\ & c_{i,j}(\xi) \geq 0, \text{ for } j = 1, .., n_C^i, \\ & \bar{c}_{i,l}(\xi) = 0, \text{ for } l = 1, .., \bar{n}_C^i\}, \end{aligned} \tag{2.20}$$

$$\begin{aligned} D_m \quad = \{\xi \in \mathbb{R}^{n_\xi} \mid\ & d_{m,j}(\xi) \geq 0, \text{ for } j = 1, .., n_D^m, \\ & \bar{d}_{m,l}(\xi) = 0, \text{ for } l = 1, .., \bar{n}_D^m\} \end{aligned} \tag{2.21}$$

where $c_{i,j}, \bar{c}_{i,l}, d_{m,j}$ and $\bar{d}_{m,l} \in \mathbb{R}[\xi]$ are polynomials. Moreover, there are polynomials $F_1, ..., F_I$ and $G_1, ..., G_M$ such that for each $\xi \in C$ there is an $i \in \{1, ..., I\}$ such that $F(\xi) = F_i(\xi)$ and $\xi \in C_i$, and for each $\xi \in D$ there is an $m \in \{1, ..., M\}$ such that $G(\xi) = G_m(\xi)$ and $\xi \in D_m$. Hence, the hybrid system (2.10) can then also be expressed as

$$\dot{\xi} = F_i(\xi), \qquad \xi \in C_i, \ i = 1, ..., I, \tag{2.22a}$$

$$\xi^+ = G_m(\xi), \quad \xi \in D_m, \ m = 1, ..., M. \tag{2.22b}$$

Remark 2.6. Note that, in general, a possible consequence of casting a hybrid system of the general form (2.10) into the specific form (2.22) is the admittance of a slightly richer set of solutions. This is due to the modeling restrictions we specify for the model (2.22), e.g. the fact that C_i and D_m must be expressed as semi-algebraic sets, in order to later apply SOS techniques. Interestingly, with the modeling assumptions considered in this chapter, the NCS can be cast into the form (2.22) without enriching the solution set. ◁

We will use the above notation to expand Theorem 2.1 in the spirit of [100] by applying a technique similar to the S-procedure [21], called the positivstellensatz [73,101], in order to encode the information that the inequalities (2.18) and (2.19) only have to be satisfied on the sets $C \backslash \mathcal{A}$ and $D \backslash \mathcal{A}$ into the inequalities.

Theorem 2.2. *Consider a hybrid system $\Sigma = (C, F, D, G)$ as in (2.22) with the sets $C = \cup_{i=1}^{I} C_i$ and $D = \cup_{m=1}^{M} D_m$, where C_i is of the form (2.20) and D_m is of the form (2.21), and F_i and G_m are polynomial functions for all $i = 1, ..., I$*

and $m = 1, ..., M$. Furthermore, consider a compact set $\mathcal{A} \subset \mathbb{R}^{n_\xi}$ satisfying $G(D \cap \mathcal{A}) \subset \mathcal{A}$. If every solution of Σ exists for all times $t \in [0, \infty)$ and there exist (i.) a Lyapunov function candidate V for $(\Sigma, \mathcal{A})$ that satisfies Definition 2.2, (ii.) polynomials $\bar{r}_{i,l},\ \bar{s}_{m,l} \in \mathbb{R}[\xi]$ and (iii.) SOS polynomials $r_{i,j},\ s_{m,j} \in \mathbb{R}[\xi]$ such that

$$\langle \nabla V(\xi), F_i(\xi) \rangle + \sum_{j=1}^{n_C^i} r_{i,j}(\xi) c_{i,j}(\xi) + \\ + \sum_{l=1}^{\bar{n}_C^i} \bar{r}_{i,l}(\xi) \bar{c}_{i,l}(\xi) < 0 \quad \forall\ \xi \notin \mathcal{A},\ i = 1, ..., I, \tag{2.23}$$

$$V(G_m(\xi)) - V(\xi) + \sum_{j=1}^{n_D^m} s_{m,j}(\xi) d_{m,j}(\xi) + \\ + \sum_{l=1}^{\bar{n}_D^m} \bar{s}_{m,l}(\xi) \bar{d}_{m,l}(\xi) \leq 0 \quad \forall\ \xi \notin \mathcal{A},\ m = 1, ...M, \tag{2.24}$$

then the set $\mathcal{A}$ is GAS.

Proof If (2.23) holds, then since the terms $r_{i,j}(\xi) c_{i,j}(\xi)$ are nonnegative and the terms $\bar{r}_{i,j}(\xi) \bar{c}_{i,j}(\xi)$ are zero when $\xi \in C_i$, it must be true that $\langle \nabla V(\xi), F_i(\xi) \rangle < 0$ when $\xi \in C_i \backslash \mathcal{A}$. Using the fact that $C = \cup_{i=1}^{I} C_i$ and for all $\xi \in C$ there exists an i such that $\xi \in C_i$ and $F(\xi) = F_i(\xi)$, we have that $\langle \nabla V(\xi), F(\xi) \rangle < 0$ when $\xi \in C \backslash \mathcal{A}$. Hence, (2.18) holds. By the same reasoning, if (2.24) holds then (2.19) must hold. Thus, if the function V satisfies the hypotheses of Theorem 2.2, then it also satisfies the hypotheses of Theorem 2.1 thereby proving GAS of $\mathcal{A}$. ■

Remark 2.7. The SOS relaxation technique as in Theorem 2.2 can also be applied to encode that the (polynomial) function V only has to be nonnegative on $(C \cup D) \backslash \mathcal{A}$ (as required in Definition 2.2) into (polynomial) inequalities. We encode this by introducing polynomials $\bar{p}_{m,l},\ \bar{q}_{i,l} \in \mathbb{R}[\xi]$ and SOS polynomials $p_{m,j},\ q_{i,j} \in \mathbb{R}[\xi]$ leading to the following inequalities

$$V(\xi) - \sum_{j=1}^{n_C^i} q_{i,j}(\xi) c_{i,j}(\xi) - \sum_{l=1}^{\bar{n}_C^i} \bar{q}_{i,l}(\xi) \bar{c}_{i,l}(\xi) \geq 0, \\ \forall \xi \notin \mathcal{A},\ \forall i = 1, ..., I, \tag{2.25}$$

$$V(\xi) - \sum_{j=1}^{n_D^m} p_{m,j}(\xi) d_{m,j}(\xi) - \sum_{l=1}^{\bar{n}_D^m} \bar{p}_{m,l}(\xi) \bar{d}_{m,l}(\xi) \geq 0, \\ \forall \xi \notin \mathcal{A},\ \forall m = 1, ..., M, \tag{2.26}$$

which indeed guarantee that $V(\xi) \geq 0$ for $\xi \in (C \cup D) \backslash \mathcal{A}$. ◁

Remark 2.8. In Theorem 2.2 and Remark 2.7 we only used the polynomial constraints that define C_i and D_m themselves to relax the stability conditions, and not their products. For instance, in (2.23) we could have also added terms as $\sum_l \sum_{\tilde{l}} r_{i,l,\tilde{l}}(\xi) c_{i,l}(\xi) c_{i,\tilde{l}}(\xi)$ with $r_{i,l,\tilde{l}}(\xi)$ SOS and still guarantee that $\langle \nabla V(\xi), F_i(\xi) \rangle < 0$ when $\xi \in C_i \backslash \mathcal{A}$. Clearly this would result in a further relaxation with respect to (2.23) (at the expense of a larger SOS program). See [26,73,101] for more details on these relaxations, which can be implemented for the stability analysis of NCSs along the same lines as discussed in this chapter. For ease of exposition we restricted ourselves to the form in (2.23), (2.24), (2.25) and (2.26) excluding such products. ◁

SOS conditions only guarantee non-negativity of polynomials (i.e. non-strict inequalities) but proving asymptotic stability requires the Lyapunov derivative (2.23) being negative definite (satisfying a strict inequality). Thus, we need a way to verify that a given polynomial function is negative or positive definite by checking SOS (positive semidefinite) conditions. We will use the following proposition from [99] to check for positive definiteness of a given polynomial.

Proposition 2.1. *Given a polynomial $p \in \mathbb{R}[\xi]$ of degree $2d$, let $W(\xi) = \sum_{i=1}^{n_\xi} \sum_{j=1}^{d} \epsilon_{i,j} \xi_i^{2j}$ be such that*

$$\sum_{j=1}^{d} \epsilon_{i,j} > \gamma \quad \text{for all} \quad i = 1, ..., n \tag{2.27}$$

with γ a positive number, and $\epsilon_{i,j} \geq 0$ for all i and j. Then the condition

$$p(\xi) - W(\xi) \geq 0 \qquad (p(\xi) - W(\xi) \text{ is SOS}) \tag{2.28}$$

guarantees the positive definiteness of p, i.e. $p(\xi) > 0$ for all $\xi \neq 0$.

Proposition 2.1 and Theorem 2.2 form the basis to build the SOS programs that can prove stability of our NCS model (2.10) with (2.11)-(2.15).

2.3.3 Stability of Hybrid NCS Models via SOS Techniques

In this section we will specify how to verify GAS of the set $\mathcal{A}$, as defined in (2.16), of the hybrid NCS models using SOS techniques. We will distinguish three cases. In all three cases, we choose a Lyapunov function candidate satisfying Definition 2.2. Specifically, the candidate we choose is polynomial and we impose as a constraint that it can be written as a SOS (using S-procedure like relaxations as in Remark 2.7) so that (*i.*) and (*ii.*) are satisfied. Condition (*iii.*) will be guaranteed by the explicit form and the continuity of the Lyapunov function candidate. Condition (*iv.*) will be satisfied by imposing that $V(\xi) \geq W(\chi)$ for all $\xi \in C \cup D \backslash \mathcal{A}$ where χ is defined in (2.17) and $W(\chi)$ is a (radially unbounded)

function as given in Proposition 2.1. In this way, the conditions of Definition 2.2 are immediately satisfied.

Remark 2.9. In the case when f and g in (2.11) are linear functions, we do not need to satisfy Condition $(iv.)$ of Definition 2.2 to guarantee GAS of the set $\mathcal{A}$. Removing Condition $(iv.)$ implies that the Lyapunov function candidate can only prove local asymptotic stability in Theorem 2.1, see [51]. However, if f and g are linear functions and the protocol function h satisfies $h(\sigma, \alpha e) = \alpha h(\sigma, e)$ for all $\sigma \in \{1, ..., N\}$, $e \in \mathbb{R}^{n_e}$ and $\alpha \geq 0$ (as is the case for the RR and TOD protocols), local asymptotic stability implies GAS due to the fact that an initial state $(x_0, e_0, s_0, \sigma_0, \ell_0, \tau_0)$ at $t = 0$ has a solution $(x, e, s, \sigma, \ell, \tau)$ if and only if initial state $(\alpha x_0, \alpha e_0, \alpha s_0, \sigma_0, \ell_0, \tau_0)$ has a solution $(\alpha x, \alpha e, \alpha s, \sigma, \ell, \tau)$ for $\alpha > 0$. Due to this positive homogeneity property, local asymptotic stability of $\mathcal{A}$ implies GAS of $\mathcal{A}$ in the linear case. $\lhd$

In the remainder of this section we show how the SOS program can be set up to solve the NCS stability problem in three cases. First we will address the simplest case of an NCS without a shared communication medium, i.e. all sensor and controller data is transmitted at every transmission time as discussed briefly in Remark 2.4. Then we will expand those results to solve the case with RR protocols and then finally consider TOD protocols. In each of these cases, the essential steps are the transformation of these three NCS variants into hybrid models (2.10) with $F : C \to \mathbb{R}^{n_\xi}$ and $G : D \to \mathbb{R}^{n_\xi}$ being piecewise polynomial as in Definition 2.5 leading to (2.22), and applying Theorem 2.2 together with Remark 2.7 and Proposition 2.1 to derive suitable SOS programs. For ease of exposition we will assume that f and g are polynomials. The extension to piecewise polynomials is straightforward. Also the extension to other protocols for which $\delta : \{1, ..., N\} \times \mathbb{R}^{n_e} \to \{1, ..., N\}$ is a piecewise constant function where the sets $S_{\sigma,j} = \{e \in \mathbb{R}^{n_e} \mid \delta(\sigma, e) = j\}$ are a finite union of basic semi-algebraic sets for each $\sigma, j \in \{1, ..., N\}$ can be obtained by following the same line of reasoning as below. In fact, in line with Definition 4, we call the protocols with the above mentioned property piecewise polynomial protocols.

Unshared Communication

Without the communication medium being shared, all sensor measurement and control commands are sent (i.e. updated) at every transmission time. Therefore, in accordance with (2.5), it holds that $h(\sigma, e) = 0$ for all σ and e, and consequently, we can simplify the hybrid model by omitting σ from the state ξ, resulting in $\xi = (\chi, \tau, \ell)$. Given the definitions of C and D for Σ_{NCS}, it is necessary to partition C and D based on the discrete state $\ell \in \{0, 1\}$ as

$$C_0 = \{\ \xi \in \mathbb{R}^{n_\xi} \mid \ell = 0,\ \tau \geq 0,\ h_{max} - \tau \geq 0\}, \tag{2.29a}$$

$$C_1 = \{\ \xi \in \mathbb{R}^{n_\xi} \mid \ell = 1,\ \tau \geq 0,\ \tau_{max} - \tau \geq 0\}, \tag{2.29b}$$

with the corresponding polynomial flow map

$$F_0(\xi) = F_1(\xi) = F(\chi, \tau, \ell) = (f(x,e), g(x,e), 0, 1, 0) \tag{2.30}$$

and

$$D_0 = \{\ \xi \in \mathbb{R}^{n_\xi} \mid \ell = 0,\ \tau - h_{min} \geq 0,\ h_{max} - \tau \geq 0\}, \tag{2.31a}$$
$$D_1 = \{\ \xi \in \mathbb{R}^{n_\xi} \mid \ell = 1,\ \tau - \tau_{min} \geq 0,\ \tau_{max} - \tau \geq 0\}, \tag{2.31b}$$

with the corresponding polynomial jump maps

$$G_0(\xi) = G_0(\chi, \tau, \ell) = (x, e, -e, 0, 1), \tag{2.32a}$$
$$G_1(\xi) = G_1(\chi, \tau, \ell) = (x, s + e, 0, \tau, 0). \tag{2.32b}$$

Note that $C = C_0 \cup C_1$, with C_i, $i = 0, 1$, basic semialgebraic sets satisfying (2.20), and $D = D_0 \cup D_1$, with D_m, $m = 0, 1$, basic semialgebraic sets satisfying (2.21). In addition, the mappings G_0, G_1 and $F_0 = F_1 = F$ are polynomial functions, provided that f and g are polynomial functions. This shows that $F : C \to \mathbb{R}^{n_\xi}$ and $G : D \to \mathbb{R}^{n_\xi}$ are piecewise polynomial, under the standing assumption that f and g are polynomial. Using the above expressions for $C_i, i = 0, 1$ and $D_m, m = 0, 1$, the polynomials $c_{i,j}$ and $d_{m,j}$ are defined as shown in Table 2.1.

$c_{i,j}$	$d_{m,j}$
$c_{0,1} = \tau$	$d_{0,1} = \tau - h_{min}$
$c_{0,2} = h_{max} - \tau$	$d_{0,2} = h_{max} - \tau$
$c_{1,1} = \tau$	$d_{1,1} = \tau - \tau_{min}$
$c_{1,2} = \tau_{max} - \tau$	$d_{1,2} = \tau_{max} - \tau$

Table 2.1: SOS relaxations for an NCS with unshared communication.

We did not include the equality constraints (e.g. $\ell = 0$ for C_0 or $\ell = 1$ for C_1) in Table 2.1 as we will encode them through the use of multiple Lyapunov functions explicitly depending on ℓ. The Lyapunov function candidate we propose to use is of the form[3]

$$V(\xi) = V_\ell(\chi, \tau) = \varphi_\ell(\tau)\tilde{W}_\ell(\chi). \tag{2.33}$$

We specify that the function $\varphi_\ell \in \mathbb{R}[\tau]$ is a polynomial of any degree and $\tilde{W}_\ell \in \mathbb{R}[\chi]$ is a polynomial with an even degree for $\ell = 0, 1$. Although a more general (polynomial) Lyapunov function candidate could also be considered within this framework, the particular choice of Lyapunov function (2.33) offers a computational reduction in the resulting SOS program due to the fact that fewer

[3]Note that the multiple Lyapunov function $V(\xi) = V_\ell(\chi, \tau)$ can be written as one single polynomial Lyapunov function $V(\xi) = \ell V_1(\chi, \tau) + (1 - \ell)V_0(\chi, \tau)$.

coefficients need to be determined. This computational reduction is essential in enabling this SOS technique to analyze larger state-space dimensions effectively, as will be shown in Section 2.4. Moreover, the form (2.33) was motivated by the success of a similar form in the previous work [22,58]. To guarantee Condition (*i.*) of Definition 2.2, we will use the regional information $c_{i,j}$ (as in Theorem 2.2) to explicitly specify that $V_\ell(\xi)$ is nonnegative only when $\xi \in C \cup D = C$. Choosing $\tilde{W}_\ell$ such that $\tilde{W}_\ell(\chi) = 0$ when $\chi = 0$ ensures that Condition (*iii.*) of Definition 2.2 is satisfied. Combining Proposition 2.1 and Theorem 2.2 together with Remark 2.7 leads to the polynomial constraints as shown in Table 2.2, where the inequalities will be implemented through SOS conditions.

In Table 2.2, we use the notation $\bar{G}_i$, $i = 0, 1$, to denote the jump map $G_i, i = 0, 1$, restricted to the elements corresponding to χ and τ, i.e.

$$\bar{G}_0(\chi, \tau) = (x, e, -e, 0),$$
$$\bar{G}_1(\chi, \tau) = (x, s + e, 0, \tau).$$

The constraints in Table 2.2 must hold for all $\ell \in \{0, 1\}$ and $i \in \{1, 2, .., n_\chi\}$ and fixed $\gamma_1 > 0$ and $\gamma_2 > 0$. The functions $W_{1,\ell}, W_{2,\ell} \in \mathbb{R}[\chi]$ for $\ell = 0, 1$ are defined as

$$W_{1,\ell}(\chi) = \sum_{i=1}^{n_\chi} \sum_{j=1}^{d} \epsilon_{\ell,i,j} \chi_i^{2j} \tag{2.34a}$$

$$W_{2,\ell}(\chi) = \sum_{i=1}^{n_\chi} \sum_{j=1}^{d} \eta_{\ell,i,j} \chi_i^{2j} \tag{2.34b}$$

as in Proposition 2.1. These functions only need to depend on $\chi = (x, e, s)$ to guarantee (2.23) of Theorem 2.2 and (*iv.*) of Definition 2.2 because $\mathcal{A} = \{\xi \in C \cup D \mid \chi = 0\}$. Note that we did not need to apply (2.26) of Remark 2.7 since $D \subseteq C$. Constraint 3 in Table 2.2 is derived from combining (2.28) and (2.23).

Remark 2.10. Besides choosing the orders of φ_ℓ and $\tilde{W}_\ell$ in the Lyapunov function itself, the design freedom left available is to choose the order of the polynomials $q_{\ell,j}(\chi, \tau), r_{\ell,j}(\chi, \tau)$ and $s_{\ell,j}(\chi, \tau)$ such that the order of the product $q_{\ell,j}(\chi, \tau)c_{\ell,j}(\tau)$ matches the order of $V_\ell(\chi, \tau)$, the order of $r_{\ell,j}(\chi, \tau)c_{\ell,j}(\tau)$ matches the order of $\langle \nabla V_\ell(\chi, \tau), F(\xi) \rangle$ and the order of $s_{\ell,j}(\chi, \tau)d_{\ell,j}(\tau)$ matches the order of $V_\ell(\chi, \tau)$. The motivation behind this suggestion is to create sufficient freedom for the relaxation terms in Table 2.2 to be effective but do not become computationally burdensome. ◁

Hence, based on the previous reasoning, the feasibility of the SOS constraints in Table 2.2 guarantees GAS of an NCS with varying delays and varying transmission intervals, but without a shared communication medium. The communication medium being shared will be considered in the following two sections for the RR and TOD protocols. Other piecewise polynomial protocols can be handled in a similar fashion.

	Constraint Set - Unshared Communication
1a	$\sum_{j=1}^{d} \epsilon_{\ell,i,j} \geq \gamma_1, \quad \epsilon_{\ell,i,j} \geq 0$
1b	$\sum_{j=1}^{d} \eta_{\ell,i,j} \geq \gamma_2, \quad \eta_{\ell,i,j} \geq 0$
2	$V_\ell(\chi,\tau) - W_{1,\ell}(\chi) - \sum_{j=1}^{m_C} q_{\ell,j}(\chi,\tau) c_{\ell,j}(\tau) \geq 0$
3	$-\langle \nabla V_\ell(\chi,\tau), F(\xi)\rangle - W_{2,\ell}(\chi) - \sum_{j=1}^{2} r_{\ell,j}(\chi,\tau) c_{\ell,j}(\tau) \geq 0$
4a	$V_0(\chi,\tau) - V_1(\bar{G}_0(\chi,\tau)) - \sum_{j=1}^{2} s_{0,j}(\chi,\tau) d_{0,j}(\tau) \geq 0$
4b	$V_1(\chi,\tau) - V_0(\bar{G}_1(\chi,\tau)) - \sum_{j=1}^{2} s_{1,j}(\chi,\tau) d_{1,j}(\tau) \geq 0$
5	$q_{\ell,j}(\chi,\tau) \geq 0, r_{\ell,j}(\chi,\tau) \geq 0, s_{\ell,j}(\chi,\tau) \geq 0$

Table 2.2: SOS program for an NCS with unshared communication.

Round Robin Protocol

In the case of the Round Robin (RR) protocol, we exploit the periodicity present in the RR protocol $G_0(\chi,\tau,\sigma,\ell) = G_0(\chi,\tau,\sigma+N,\ell)$ to modify the region D_0 to be node-dependent in the sense that the jump map G_0 can be written in a piecewise polynomial form. Specifically, we partition $D_m, m = 0, 1$, as in (2.31), further as

$$\begin{aligned} D_{0,1} &= \{\, \xi \in \mathbb{R}^{n_\xi} \mid \ell = 0,\ \tau - h_{min} \geq 0,\ h_{max} - \tau \geq 0,\ \sigma = N\} \\ D_{0,j} &= \{\, \xi \in \mathbb{R}^{n_\xi} \mid \ell = 0,\ \tau - h_{min} \geq 0,\ h_{max} - \tau \geq 0,\ \sigma + 1 = j\} \\ D_1 &= \{\, \xi \in \mathbb{R}^{n_\xi} \mid \ell = 1,\ \tau - \tau_{min} \geq 0, \tau_{max} - \tau \geq 0\} \end{aligned}$$

for $j = 2, ..., N$. Now G is specified by

$$G_{0,j}(\chi,\tau,\sigma,\ell) = (x, e, \tilde{h}(j,e) - e, 0, j, 1) \tag{2.35a}$$

$$G_1(\chi,\tau,\sigma,\ell) = (x, s + e, 0, \tau, \sigma, 0) \tag{2.35b}$$

for $j = 1, ..., N$, where $\tilde{h}$ is given as in (2.7). Hence, $D_0 = \cup_{j=1}^{N} D_{0,j}$ and each function $G_{0,j}$ is polynomial, thus we have obtained a hybrid system of the form (2.22). The definitions of $D_{0,j},\ j = 1, ..., N$, and D_1 allow us to again use the polynomials defined in Table 2.1.

The Lyapunov function we consider to prove stability for Σ_{NCS} with RR protocols is the following:

$$V(\xi) = V_{\ell,\sigma}(\chi,\tau) = \varphi_{\ell,\sigma}(\tau)\tilde{W}_{\ell,\sigma}(\chi). \tag{2.36}$$

The explicit dependence on σ provides extra freedom in the Lyapunov function candidate. With this new Lyapunov candidate, we obtain, in a similar way as

Constraint Set - RR Protocol	
1a	$\sum_{j=1}^{d} \epsilon_{\ell,\sigma_1,i,j} \geq \gamma_1, \quad \epsilon_{\ell,\sigma_1,i,j} \geq 0$
1b	$\sum_{j=1}^{d} \eta_{\ell,\sigma_1,i,j} \geq \gamma_2, \quad \eta_{\ell,i,j} \geq 0$
2	$V_{\ell,\sigma_1}(\chi,\tau) - W_{1,\ell,\sigma_1}(\chi) - \sum_{j=1}^{m_C} q_{\ell,j}(\chi,\tau) c_{\ell,j}(\tau) \geq 0$
3	$-\langle \nabla V_{\ell,\sigma_1}(\chi,\tau), F(\xi)\rangle - W_{2,\ell,\sigma_1}(\chi) - \sum_{j=1}^{2} r_{\ell,j}(\chi,\tau) c_{\ell,j}(\tau) \geq 0$
4a	$V_{0,N}(\chi,\tau) - V_{1,1}(\bar{G}_{0,1}(\chi,\tau)) - \sum_{j=1}^{2} s_{0,j}(\chi,\tau) d_{0,j}(\tau) \geq 0$
4b	$V_{0,\sigma_2}(\chi,\tau) - V_{1,\sigma_2+1}(\bar{G}_{0,\sigma_2+1}(\chi,\tau)) - \sum_{j=1}^{2} s_{0,j}(\chi,\tau) d_{0,j}(\tau) \geq 0$
4c	$V_{1,\sigma_1}(\chi,\tau) - V_{0,\sigma_1}(\bar{G}_1(\chi,\tau)) - \sum_{j=1}^{2} s_{1,j}(\chi,\tau) d_{1,j}(\tau) \geq 0$
5	$q_{\ell,j}(\chi,\tau) \geq 0, r_{\ell,j}(\chi,\tau) \geq 0, s_{\ell,j}(\chi,\tau) \geq 0$

Table 2.3: SOS program for an NCS with the RR Protocol.

in Section 2.3.3, the constraint set as shown in Table 2.3. We use the notation $\bar{G}_{0,j}$ for $j = 1, ..., N$ to denote the jump map $G_{0,j}$ restricted to the first elements corresponding to χ and τ, i.e.,

$$\bar{G}_{0,j}(\chi,\tau) = (x, e, \tilde{h}(j,e) - e, 0).$$

The constraints in Table 2.3 must hold for all $\ell \in \{0,1\}$, $i \in \{1,2,..,n_\chi\}$, $\sigma_1 \in \{1,2,...,N\}$ and $\sigma_2 \in \{1,2,...,N-1\}$ and fixed $\gamma_1 > 0$ and $\gamma_2 > 0$. The functions $W_{1,\ell,\sigma_1},\ W_{2,\ell,\sigma_1} \in \mathbb{R}[\chi]$ are of the form (2.34).

Try-Once-Discard Protocol

For the TOD protocol, we know that at the time a jump occurs when $\ell = 0$, the node that is granted network access has a greater (or equal) error $|e_j|$ than all other nodes, i.e. $|e_j| \geq |e_i|$ for all i (see (2.9)). We will use this knowledge to modify the region D_0 so that the jump map G_0 is written as a piecewise polynomial function. We express D_0 as $\cup_j D_{0,j}$ with $D_{0,j},\ j = 1, ..., N$, basic semialgebraic sets given by

$$D_{0,j} = \{\ \xi \in \mathbb{R}^{n_\xi} \mid \ell = 0,\ \tau - h_{min} \geq 0,\ h_{max} - \tau \geq 0,\ |e_j|^2 - |e_i|^2 \geq 0\ \forall i = 1, ..., N\}.$$

	Constraint Set - TOD Protocol
1a	$\sum_{j=1}^{d} \epsilon_{\ell,\sigma_1,i,j} \geq \gamma_1, \quad \epsilon_{\ell,\sigma_1,i,j} \geq 0$
1b	$\sum_{j=1}^{d} \eta_{\ell,\sigma_1,i,j} \geq \gamma_2, \quad \eta_{\ell,i,j} \geq 0$
2	$V_{\ell,\sigma_1}(\chi,\tau) - W_{1,\ell,\sigma_1}(\chi) - \sum_{j=1}^{m_C} q_{\ell,j}(\chi,\tau) c_{\ell,j}(\tau) \geq 0$
3	$-\langle \nabla V_{\ell,\sigma_1}(\chi,\tau), F(\xi)\rangle - W_{2,\ell,\sigma_1}(\chi) - \sum_{j=1}^{2} r_{\ell,j}(\chi,\tau) c_{\ell,j}(\tau) \geq 0$
4a	$V_{0,\sigma_1}(\chi,\tau) - V_{1,\sigma_2}(\bar{G}_{0,\sigma_2}(\chi,\tau)) - \sum_{j=1}^{2} s_{0,j}(\chi,\tau) d_{0,j}(\tau) - \sum_{j=1}^{N} (\lvert e_{\sigma_2}\rvert^2 - \lvert e_j\rvert^2) b_{\sigma_2,j}(\chi,\tau) \geq 0$
4b	$V_{1,\sigma_1}(\chi,\tau) - V_{0,\sigma_1}(\bar{G}_1(\chi,\tau)) - \sum_{j=1}^{2} s_{1,j}(\chi,\tau) d_{1,j}(\tau) \geq 0$
5	$q_{\ell,j}(\chi,\tau) \geq 0,\ r_{\ell,j}(\chi,\tau) \geq 0, \ s_{\ell,j}(\chi,\tau) \geq 0, b_{\sigma_2,j}(\chi,\tau) \geq 0$

Table 2.4: SOS program for an NCS with the TOD Protocol.

In addition,

$$D_1 = \{\ \xi \in \mathbb{R}^{n_\xi} \mid \ell = 1,\ \tau - \tau_{min} \geq 0, \tau_{max} - \tau \geq 0\}.$$

The corresponding jump maps $G_{0,j}$ and G_1 are given by (2.35),the flow map F_i is again described by (2.30) and the corresponding flow set $C_i, i = 0, 1$, is described by (2.29). Now we have arrived at a system of the form (2.22) and we can apply Theorem 2.2 together with Remark 2.7 and Proposition 2.1.

We consider the Lyapunov function (2.36) to prove stability for Σ_{NCS} with TOD protocols. Notice that we do not need σ in the hybrid model of the NCS as $h(\sigma, e) = h(e)$. However, we do use σ to provide extra freedom in the (multiple) Lyapunov function (as in the RR case). The resulting constraint set guaranteeing GAS of $\mathcal{A}$ is shown in Table 2.4. The constraints must hold for all $\ell \in \{0, 1\}$, $i \in \{1, 2, .., n_\chi\}$, and $\sigma_1, \sigma_2 \in \{1, 2, ..., N\}$ and fixed $\gamma_1 > 0$ and $\gamma_2 > 0$. The functions $W_{1,\ell,\sigma_1},\ W_{2,\ell,\sigma_1} \in \mathbb{R}[\chi]$ are of the form (2.34).

2.4 Comparative Examples

We will illustrate our SOS approach by applying it to four different NCS examples. In the first example, we use the sampled-data model without the communication medium being shared. In the second and third examples, we study a linear plant connected to a linear controller via a shared network, and thus

include network protocols. We compare the SOS approach developed in this chapter with the technique developed in [58]. We show that our results are less conservative than the results obtained using the technique developed in [58]. As these three examples consider linear plants and controllers, the fourth example will illustrate that the SOS approach can easily be used to show stability for plants and controllers with polynomial dynamics as well.

Our calculations result in tradeoff curves for combinations of h_{max} and τ_{max} (where we take $h_{min} = \tau_{min} = 0$ to be able to compare the results to [58]). These tradeoff curves are created by a gridding procedure in the sense that for different combinations of h_{max} and τ_{max}, the feasibility of the SOS program is assessed using SOSTOOLS [108].

Remark 2.11. Note that in the case of linear plants and controllers (as considered in the first three examples), GAS of $\mathcal{A}$ can be guaranteed by proving only local asymptotic stability of $\mathcal{A}$, as mentioned in Remark 2.9. This means that the SOS constraint sets detailed in Section 2.3.3 can be relaxed by removing the polynomial $W_{1,\ell}(\xi)$, as defined in (2.34a). This relaxation is possible because the only purpose of including $W_{1,\ell}(\xi)$ in the constraint sets is to ensure the Lyapunov function candidate is radially unbounded (and hence being capable of proving the global asymptotic stability property). $\lhd$

2.4.1 Example 1 - Unshared Communication

A 'classic' and well-studied system (see [77] and the reference therein), is given by (2.1), (2.2) where $\dot{x}_p = \hat{u}$, $y = x_p$, $u = -\hat{y}$. For constant transmission interval and no delays, the system can be guaranteed to be stable for transmission intervals up to 2 seconds. In [77], stability of the system for variable transmission intervals is guaranteed for $h_k \in [0\ 1.99]$, $k \in \mathbb{N}$, in a delay-free situation, which corresponds to $h_{min} = 0$ and a MATI h_{max} of 1.99 seconds. This does not include much conservatism, as can be concluded from the constant transmission interval result. The results obtained in [77], when delays are present, are given in Fig. 2.2.

Two SOS programs (SOSPs) are constructed with the constraints from Section 2.3.3. The first program used a third-order V_ℓ, which consists of a linear function $\varphi_\ell \in \mathbb{R}[\tau]$ and quadratic $\tilde{W}_\ell \in \mathbb{R}[\chi]$, whereas the second program uses a fifth-order V_ℓ, which consists of a third-order function for φ_ℓ and quadratic $\tilde{W}_\ell$. Already with φ_ℓ being a polynomial of third order, the results of [77], which according to the above do not contain much conservatism, are almost replicated, as shown in Fig. 2.2. The tradeoff curve for a linear φ_ℓ still includes considerable conservatism. The flexibility of our SOS approach allows us to gradually increase the order of φ_ℓ to reduce conservatism in the results, as Fig. 2.2 clearly shows.

To illustrate the form of Lyapunov functions used and to indicate some numerical values, let us provide them for the situation $(h_{max}, \tau_{max}) = (0.45, 0.1)$.

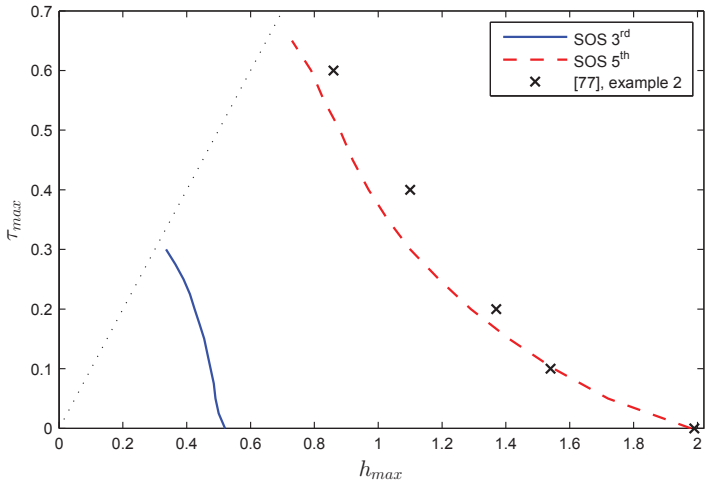


Figure 2.2: Tradeoff curves for a sampled-data NCS with an unshared communication medium.

The stability of this situation is proven by a third-order polynomial Lyapunov function as in (2.33) consisting of a linear function φ_ℓ and quadratic $\tilde{W}_\ell$ given by the SOS program as

$$\begin{aligned} V_0(\chi,\tau) &= 49.187x + 3.885\tau x^2 - 0.340\tau ex + 52.275e^2 \\ &\quad - 105.470\tau e^2 + 121.670s^2 - 69.402\tau s^2, \\ V_1(\chi,\tau) &= 49.187x + 31.165\tau x^2 - 18.355\tau ex + 61.087e^2 \\ &\quad - 132.550\tau e^2 - 56.680\tau xs + 112.200es \\ &\quad - 134.830\tau es + 55.747s^2 - 30.193\tau s^2. \end{aligned}$$

Note that although setting $h_{min} = \tau_{min} = 0$ was chosen for reasons of comparison to the earlier work [77] which cannot exploit nonzero lower bounds, in many practical situations there exists a nonzero lower bound due to hardware limitations. If we assume that the transmission frequency of the network cannot exceed 10Hz (i.e. $h_{min} = 0.1$) and the minimum delay is 10ms ($\tau_{min} = 0.01$) then new tradeoff curves in Fig. 2.2 can be computed. To provide some indication of how these non-zero lower bounds can lead to larger h_{max}, τ_{max} guaranteeing robust stability, we focus on the point $(h_{max},\ \tau_{max}) = (1.54,\ 0.1)$, which lies on the SOS fifth-order curve in Fig. 2.2 (and thus corresponds to $h_{min} = \tau_{min} = 0$). Adopting the nonzero lower bounds $h_{min} = 0.1$ and $\tau_{min} = 0.01$, this point becomes $(1.85,\ 0.1)$, which is a 20% increase in robustness, in terms of h_{max}, with respect to the case of zero lower bounds. This clearly shows the benefit of being able to include non-zero lower bounds on delays and transmission intervals

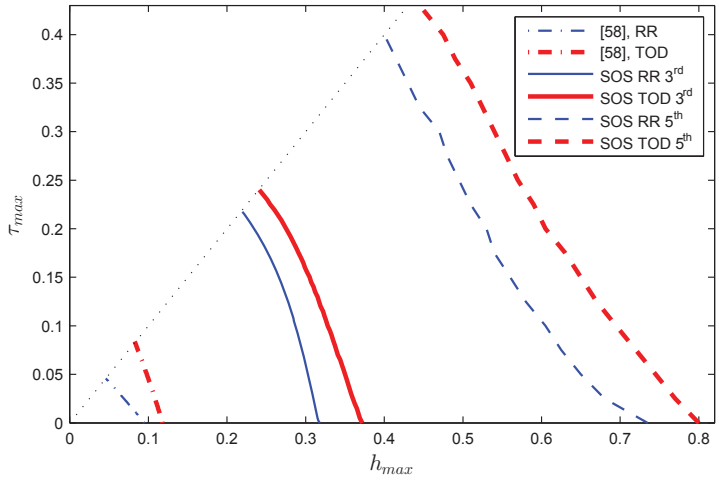


Figure 2.3: Tradeoff curves for an NCS with a shared communication medium.

in the analysis. While our method can easily exploit non-zero bounds, several existing methods cannot.

2.4.2 Example 2 - Shared Communication

In this example, we consider the plant and controller given in the form (2.1), (2.2) where

$$\begin{aligned} \dot{x}_p &= \begin{bmatrix} 0 & 1 \\ 0 & -0.1 \end{bmatrix} x_p + \begin{bmatrix} 0 \\ 0.1 \end{bmatrix} \hat{u}, \qquad y = x_p, \\ u &= -\begin{bmatrix} 3.75 & 11.5 \end{bmatrix} \hat{y}, \end{aligned}$$

which was studied in [128], although without a shared communication medium and, hence, without protocols. We will compare our method with the technique in [58].

We consider a shared communication medium, as well as varying transmission intervals and delays, and compute stability regions for both the RR and TOD protocols. We assume that only the two states are transmitted over the network, while $u(t) = \hat{u}(t)$ for all $t \in \mathbb{R}_{\geq 0}$. This results in the networked error $e = e_y$, where $y = x_p$. The constraints from Table 2.3 and Table 2.4 are implemented in a SOS program for the RR and TOD protocols, respectively. A third-order $V_{\ell,\sigma}$, which consists of a linear $\varphi_{\ell,\sigma} \in \mathbb{R}[\tau]$ and a quadratic $\tilde{W}_{\ell,\sigma} \in \mathbb{R}[\chi]$, and a fifth-order $V_{\ell,\sigma}$, which consists of a third-order $\varphi_{\ell,\sigma}$ and a quadratic $\tilde{W}_{\ell,\sigma}$, are

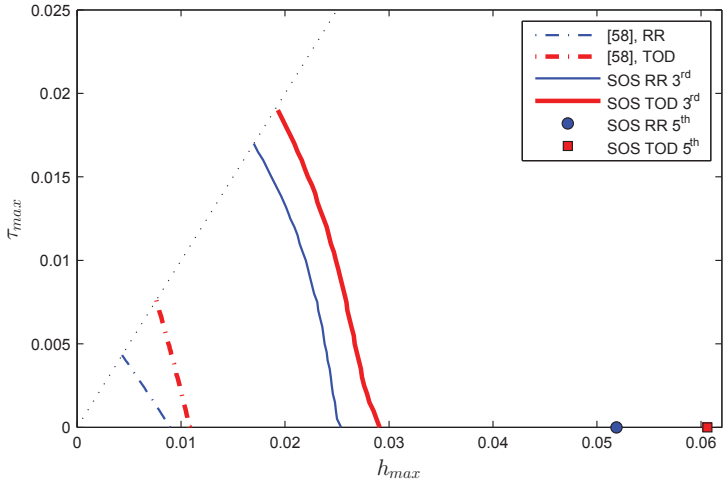


Figure 2.4: Tradeoff curves for the NCS benchmark batch reactor.

used. The resulting tradeoff curves for the method in [58] and our SOS method are shown in Fig. 2.3. The SOS method proposed in this chapter clearly reduces conservatism in the obtained tradeoff curves compared to [58] when considering a third-order $V_{\ell,\sigma}$, and reduced conservatism even further when considering a fifth-order $V_{\ell,\sigma}$ polynomial Lyapunov candidate.

2.4.3 Example 3 - Batch Reactor

In this example, the NCS benchmark system consisting of the batch reactor [22, 58, 94, 122] is compared. The batch reactor is given by a plant containing four states and a controller containing two states. For the actual system matrices, see the references provided. The two outputs y of the plant share the network, implying that they cannot communicate simultaneously, and the two controlled inputs u of the plant are considered wired to the controller (i.e. $u(t) = \hat{u}(t)$ for all $t \in \mathbb{R}_{\geq 0}$). The results for the batch reactor are shown in Fig. 2.4 and are compared to the results of the approach in [58].

From Fig. 2.4 it can be seen that the SOS approach presented in this chapter again results in less conservative results than in [58] when considering a third-order $V_{\ell,\sigma}$, which consists of a linear $\varphi_{\ell,\sigma} \in \mathbb{R}[\tau]$ and quadratic $\tilde{W}_{\ell,\sigma} \in \mathbb{R}[\chi]$. We also computed a fifth-order $V_{\ell,\sigma}$, which consists of a third-order $\varphi_{\ell,\sigma}$ and quadratic $\tilde{W}_{\ell,\sigma}$, for the delay-free case. Interestingly, in this delay-free case, the least conservative theoretical result, until recently, for h_{max} that still guarantees stability was given in [22] as $h_{max} = 0.0108$ using the TOD protocol, while we obtain $h_{max} = 0.0606$. In [122], h_{max} was estimated (using simula-

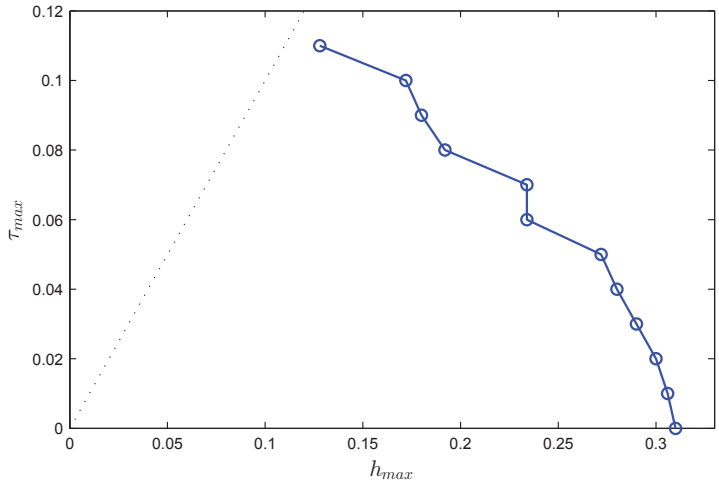


Figure 2.5: Tradeoff curves for an NCS with polynomial plant dynamics.

tions) to be between 0.06 and 0.08 for the TOD protocol. For the RR protocol, [22] provides the bound $h_{max} = 0.009$ in the delay-free case, while we obtain $h_{max} = 0.0519$. Only recently, based on a new linear discrete-time approach, [38] achieved $h_{max} = 0.066$ for the TOD protocol and $h_{max} = 0.064$ for the RR protocol in the delay-free case. These results are comparable to the results calculated with our SOS-based analysis by just using a fifth-order Lyapunov function. Hence, using the SOS-based approach, results are obtained which are close to the available upper bounds on h_{max}, while, at the same time, offering several additional beneficial features mentioned in the introduction. For instance, robust stability of the example in the next section cannot be assessed by [38].

The computational burden of the SOSP based on a fifth-order polynomial Lyapunov function for NCSs including delays becomes rather high and these results are therefore not provided. This indicates the only drawback of the proposed method: the computational complexity might grow quickly with the state dimension and the order of the Lyapunov function. Improved solvers for SOS programs are needed to reduce the computation time for such complex problems.

2.4.4 Example 4 - Polynomial Sampled-Data System

In this example, we will show that our method can also prove GAS for NCSs with the plant having polynomial dynamics. The system we consider is given by (2.1), (2.2) where $\dot{x}_p = x_p^3 + x_p^2\hat{u}$, $y = x_p$ and $u = -5\hat{y}$. The network effects

we consider for this example are varying transmission intervals and delays (the communication medium is not shared). In fact, we take $h_{min} = \tau_{min} = 0$ and aim at determining values for τ_{max} and h_{max} for which the NCS is GAS.

To do so, the constraints from Section 2.3.3 are implemented in a SOS program. For linear plants and controllers the order of $\tilde{W}_\ell \in \mathbb{R}[\chi]$ in (2.33) is often taken to be 2, i.e. quadratic, since the (x, e) flow map is then also linear. However, for polynomial plants and controllers (of at least order 2), the (x, e) flow map is nonlinear and higher-order terms are (commonly) needed to provide the stability characterization sufficient flexibility. Therefore we specify the order of $\tilde{W}_\ell$ to be 6. In addition, we specify that the function $\varphi_\ell \in \mathbb{R}[\tau]$ is linear. This results in the tradeoff curves as provided in Fig. 2.5, showing indeed that we can analyze an NCS with a polynomial plant and controller in a systematic manner.

2.5 Conclusions

In this chapter we have presented a sum of squares (SOS) approach for the stability analysis of NCSs that exhibit varying delays, varying transmission intervals and a shared communication medium. The NCS was modeled as a hybrid system, which allows for general continuous-time piecewise polynomial plant and controller dynamics. In order to use SOS techniques, the flow and jump maps of the hybrid system were transformed into piecewise polynomial descriptions. This transformation was explicitly shown for three important cases consisting of a pure sampled-data system without a shared communication medium, and NCSs with a shared communication medium and using either the Try-Once-Discard (TOD) or Round Robin (RR) protocol. We were able to show that by using SOS techniques, it is possible to improve existing results in the literature significantly. As expected, increasing the order of the polynomial Lyapunov functions leads to improved bounds on the delays and transmission intervals (at the cost of higher computational complexity). Next to a reduction in conservatism, our method offers various other beneficial features, such as dealing with non-zero lower bounds on varying delays and transmission intervals, dealing with nonlinear (polynomial) plants and controllers, not requiring an overapproximation of the NCS (as in the discrete-time approach) and finally, offering an automated method to tackle the stability problem for NCS including varying delays, transmission intervals and a shared communication medium. The only drawback of the proposed method is that the computational complexity grows quickly with the state dimension and the order of the Lyapunov function. Improved solvers for SOS programs are needed to reduce the computation time for such complex problems.

Chapter 3

Decentralized Observer-Based Control via Networked Communication

"Nobody climbs mountains for scientific reasons. Science is used to raise money for the expeditions, but you really climb for the hell of it."

- Sir Edmund Hillary

3.1 Introduction

Recently, there has been an enormous interest in the control of large-scale networked systems that are physically distributed over a wide area [86]. Examples of such distributed systems are electrical power distribution networks [20], water transportation networks [23], industrial factories [85] and energy collection networks (such as wind farms [69]). The purpose of developing control theory in this large-scale setting is to work towards the goal of a streamlined design process which consistently results in efficient operation of these vital systems. Our contribution towards this goal is in the area of stabilizing controller design. This problem setting has many features that seriously challenge the controller design.

The first feature which challenges controller design is that the controller is decentralized, in the sense that it consists of a number of local controllers that do not share information. Although a centralized controller could alternatively be considered, the achievable bandwidth associated with using a centralized control structure would be limited by long delays induced by the communication between the centralized controller and distant sensors and actuators over a (wireless) communication network [1]. The difficulty of decentralized control synthesis lies in the fact that each local controller has only local information to utilize for control, which implies that the other local control actions are unknown and can

be perceived as disturbances. This fundamental problem has received ample attention [3, 111, 121], but still many issues are actively researched today. A recent survey [13] highlights newly developed techniques to solve this problem in different settings and recommends that research should consider interconnected systems which are controlled over realistic communication channels. This forms the exact topic of the presented chapter.

The problem of synthesizing decentralized linear controllers is often referred to as the 'information-constrained' synthesis problem or the 'structured' synthesis problem due to the presence of zeros in the controller matrices corresponding to the decentralized structure. This synthesis problem is, in general, non-convex. It was shown in [110] that linear time-invariant systems which satisfy a property called 'quadratic invariance,' with respect to the controller information structure, allow for convex synthesis of optimal static feedback controllers. For the specific case of block diagonal *static state* feedback control design, [49] discovered that through a change of variable, linear matrix inequality (LMI) synthesis conditions could be formulated which guarantee robust stability. However, in the decentralized (block diagonal) dynamic output-feedback setting, the (robust) controller synthesis problem is far more complex [117].

The second feature which challenges controller design comes from the fact that when considering control of a large-scale system, it would be unreasonable to assume that all states are measured. Therefore an output-based controller is needed. This chapter will, in fact, consider an observer-based control setup, which offers the additional advantage of reducing the number of sensors needed. The later aspect alleviates the demands on the communication network design. However, it has been shown that, in general, it is hard to obtain decentralized observers providing state estimates converging to the 'true' states [121]. In [117, 129], synthesis conditions for robust decentralized observer-based control with respect to unknown non-linear subsystem coupling, which is sector bounded and state-dependent, were presented. In both the paper and this chapter, a decoupled quadratic Lyapunov function candidate was used to derive stabilizing gains that could be synthesized by transforming a linear minimization problem subject to a bilinear matrix inequality (BMI) into a two-step linear minimization problem subject to LMIs. It was also mentioned in [117] that in the simpler setting of the subsystem coupling matrices being linear and known, as is the setting in the current chapter, the robust synthesis conditions are still obtained by convexifying the overlying problem of linear minimization subject to a BMI. Finally, we point out that all the aforementioned decentralized results, excluding the notable exception of [110] which includes communication delays, consider the communication channels between sensors, actuators and controllers to be *ideal*.

The third feature which challenges controller design arises from the fact that the implementation of a decentralized control strategy may not be economically feasible without a way to inexpensively connect the sensors, actuators and controllers. Indeed, the advantages of using a wired/wireless network

compared to dedicated point-to-point (wired) connections between all sensors, controllers and actuators. are inexpensive and easily modifiable communication links. However, the drawback is that the control system is susceptible to undesirable (possibly destabilizing) side-effects such as time-varying transmission intervals, time-varying delays, packet dropouts, quantization and a shared communication medium (the latter implying that not all information can be sent over the network at once). Clearly, the decentralized observer-based controller needs to have certain robustness properties with respect to these effects. For modeling simplicity, we only consider time-varying transmission intervals and the communication medium to be shared in this work, although extensions including the other side effects can be envisioned within the presented framework. In fact, the extension to including time-varying delays will be discussed explicitly in Remark 3.6.

In the Networked Control System (NCS) literature, there are many existing results on stability analysis which consider linear static controllers [29, 44, 47, 91, 123], linear dynamic controllers [38,122], nonlinear dynamic controllers [17,58,94] and observer-based controllers [84]. However, results on controller synthesis for NCSs are rare [63]. LMI conditions for synthesis of state feedback [28] and static output-feedback [54] only became available recently. For general linear dynamic controller synthesis, [34] considered the simultaneous design of the protocol, without considering time-varying transmission intervals or delays, and resulted in a linearized BMI algorithm. General linear dynamic controller synthesis conditions were also formulated in [46], where the NCS included quantization, delay and packet dropout but without a shared communication medium, which resulted in LMI conditions only when a specific design variable (ϵ in [46]) is fixed. Synthesis conditions for observer gains that stabilize the state estimation error (but not the state of the plant itself) in the presence of a shared communication medium were given in [35]. The inclusion of varying transmission intervals were recently presented in [107]. In [126], gramian-based tools were used to synthesize observer-based gains that stabilize the closed-loop in the presence of a shared communication medium but they did not consider time-varying transmission intervals nor delays. Conditions for observer-based controller synthesis in the presence of time-varying delay, time-varying transmission intervals, and dropouts were given in [87]. The synthesis conditions were derived by changing a non-convex feasibility problem into a linear minimization problem via a linear cone complementarity algorithm. It is worth mentioning that all the aforementioned NCS results consider the *centralized* controller problem setting.

To summarize, we note that although a decentralized observer-based control structure is reasonable to use in practice, its design is extremely complex due to the fact that we simultaneously face the issues of (i) a decentralized control structure (ii) limited measurement information and (iii) communication side-effects. In this context, the contribution of this chapter is twofold: firstly, a model describing the controller decentralization and the communication side-

effects is derived, and, secondly, the most significant contribution is LMI-based synthesis conditions for decentralized switched observer-based controllers and decentralized switched static feedback controllers, which are robust to communication imperfections. For the simpler case of static output feedback, we refer the reader to [15].

3.1.1 Nomenclature

The following notation will be used. $\mathrm{diag}(A_1, \ldots, A_N)$ denotes a block-diagonal matrix with the matrices $A_1, \ldots, A_N$ on the diagonal and $A^\top \in \mathbb{R}^{m\times n}$ denotes the transpose of the matrix $A \in \mathbb{R}^{n\times m}$. For a vector $x \in \mathbb{R}^n$, $\|x\| := \sqrt{x^\top x}$ denotes its Euclidean norm. We denote by $\|A\| := \sqrt{\lambda_{\max}(A^\top A)}$ the spectral norm of a matrix A, which is the square-root of the maximum eigenvalue of the matrix $A^\top A$. For brevity, we sometimes write symmetric matrices of the form $\begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}$ as $\begin{bmatrix} A & B \\ \star & C \end{bmatrix}$. For a matrix $A \in \mathbb{R}^{n\times m}$ and two subsets $\mathbf{I} \subseteq \{1, ..., n\}$ and $\mathbf{J} \subseteq \{1, ..., m\}$, the $(\mathbf{I}, \mathbf{J})$-submatrix of A is defined as $(A)_{\mathbf{I},\mathbf{J}} := (a_{ij})_{i\in\mathbf{I}, j\in\mathbf{J}}$. In case $\mathbf{I} = \{1, ..., n\}$, we also write $(A)_{\bullet,\mathbf{J}}$.

3.2 The Model & Problem Definition

We consider a collection of coupled continuous-time linear subsystems $\mathcal{P}_1, ..., \mathcal{P}_N$ given by

$$\mathcal{P}_i : \begin{cases} \dot{x}_i(t) &= A_i x_i(t) + B_i \hat{u}_i(t) + \sum_{\substack{j=1 \\ j\neq i}}^{N} \left(A_{i,j} x_j(t) + B_{i,j} \hat{u}_j(t)\right), \\ y_i(t) &= C_i x_i(t) + \sum_{\substack{j=1 \\ j\neq i}}^{N} C_{i,j} x_j(t), \end{cases} \tag{3.1}$$

for $i \in \{1, ..., N\}$, where $x_i \in \mathbb{R}^{n_{x_i}}$, $\hat{u}_i \in \mathbb{R}^{n_{u_i}}$, and $y_i \in \mathbb{R}^{n_{y_i}}$ are the subsystem state, input and output vectors, respectively. The subsystem interaction matrices, $A_{i,j}, B_{i,j}, C_{i,j}$, $i \neq j$, represent how subsystem j affects subsystem i via state, input and output coupling, respectively. We consider this collection of subsystems to be disjoint, i.e. in the sense of [121], that is the entire collection can be compactly written as

$$\mathcal{P} : \begin{cases} \dot{x}(t) &= Ax(t) + B\hat{u}(t), \\ y(t) &= Cx(t), \end{cases} \tag{3.2}$$

with state $x = [x_1^\top, x_2^\top, ..., x_N^\top]^\top \in \mathbb{R}^{n_x}$, control input $\hat{u} = [\hat{u}_1^\top, \hat{u}_2^\top, ..., \hat{u}_N^\top]^\top \in \mathbb{R}^{n_u}$ and measured output $y = [y_1^\top, y_2^\top, ..., y_N^\top]^\top \in \mathbb{R}^{n_y}$. The matrices A, B and C are defined as

$$A := \begin{bmatrix} A_1 & A_{1,2} & \cdots & A_{1,N} \\ A_{2,1} & A_2 & & \vdots \\ \vdots & & \ddots & \\ A_{N,1} & \cdots & & A_N \end{bmatrix}, \qquad B := \begin{bmatrix} B_1 & B_{1,2} & \cdots & B_{1,N} \\ B_{2,1} & B_2 & & \vdots \\ \vdots & & \ddots & \\ B_{N,1} & \cdots & & B_N \end{bmatrix},$$

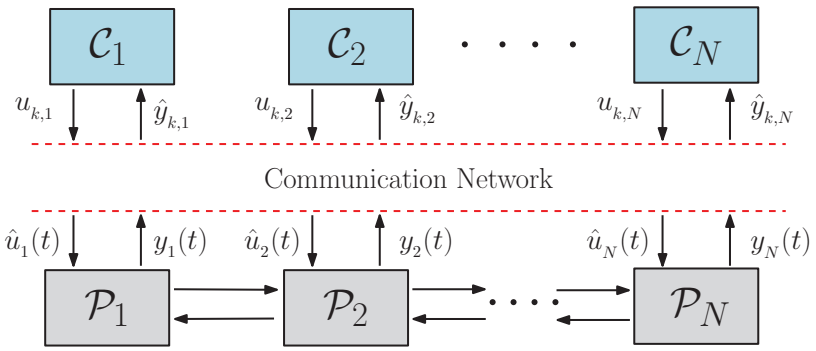


Figure 3.1: A decentralized NCS.

$$C := \begin{bmatrix} C_1 & C_{1,2} & \cdots & C_{1,N} \\ C_{2,1} & C_2 & & \vdots \\ \vdots & & \ddots & \\ C_{N,1} & \cdots & & C_N \end{bmatrix}.$$

The objective of this chapter is to present an approach for synthesis of a controller for system (3.2) that has the following features: (i) discrete-time (desirable for networked sampled-data implementation); (ii) decentralized; (iii) output-based; (iv) robustly stabilizes the origin of system (3.2) despite the uncertain time-varying transmission intervals $h_k \in [h_{min}, h_{max}]$; (v) operates in the presence of a shared communication medium: not all measured outputs and control inputs can be communicated simultaneously and a protocol schedules which information is sent at the transmission instants.

Due to these design features, we consider a decentralized control structure consisting of N local controllers $\mathcal{C}_i$, $i \in \{1, ..., N\}$, which communicate with the sensors and actuators of the plant via a shared network. The decentralized control structure we consider, 'parallels' the chosen plant decomposition as in (3.1). This is depicted in Fig. 3.1, where the i^{th} controller receives measurements from and sends control commands to the i^{th} subsystem only.

In the next sections, we will provide additional information regarding the setup in Fig. 3.1 by discussing the consequences of the design features of the controller in more detail. In particular, in Section 3.2.1, a description of the network imperfections is provided for which the controller has to be robust. In Section 3.2.1, a switching observer-based control structure will be presented that will switch based on the received measurement information and, finally, in Section 3.2.3 and Section 3.3, a closed-loop model suitable for controller synthesis is derived incorporating the aforementioned aspects.

3.2.1 Network Description

Communication between sensors, actuators and controllers will take place via a shared network, see Fig. 3.1. Here, we will consider two network effects: namely, time-varying transmission intervals and a shared communication medium, where the latter imposes the need for a scheduling protocol to determine what measurement and control command data is transmitted at each transmission instant. In Remark 3.6, we will also explain how time-varying delays can be incorporated in a straightforward manner.

Assuming that the transmission intervals $h_k = t_{k+1} - t_k$ are contained in $[h_{min}, h_{max}]$ for some $0 < h_{min} \leq h_{max}$, i.e. $h_k \in [h_{min}, h_{max}]$ for all $k \in \mathbb{N}$ and a zero-order-hold assumption on the inputs $\hat{u}$, meaning that

$$\hat{u}(t) = \hat{u}_k \text{ for all } t \in [t_k, t_{k+1}),\ k \in \mathbb{N}, \tag{3.3}$$

the exact discrete-time equivalent of (3.2) is

$$\mathcal{P}_{h_k} : \left\{ \begin{array}{rcl} x_{k+1} & = & \bar{A}_{h_k} x_k + \bar{B}_{h_k} \hat{u}_k, \\ y_k & = & C x_k, \end{array} \right. \tag{3.4}$$

where $\bar{A}_{h_k} := e^{Ah_k}$ and $\bar{B}_{h_k} := \int_0^{h_k} e^{As} ds B$. In (3.4), $x_k := x(t_k),\ y_k := y(t_k)$, with t_k the transmission instants, and $\hat{u}_k$ is the discrete-time control action available at the plant at $t = t_k$.

Since the plant and controller are communicating through a network with a shared communication medium, the actual input of the plant $\hat{u}_k \in \mathbb{R}^{n_u}$ is not equal to the controller output u_k and the actual input of the controller $\hat{y}_k \in \mathbb{R}^{n_y}$ is not equal to the sampled plant output y_k. Instead, $\hat{u}_k$ and $\hat{y}_k$ are 'networked' versions of u_k and y_k, respectively. In Section 3.2.1, we will detail how the controller output u_k will be determined based on $\hat{y}_k$ (see Fig. 3.1).

To explain the effect of the shared communication medium and thus the difference between $\hat{y}_k$ and y_k, and $\hat{u}_k$ and u_k, $k \in \mathbb{N}$, realize that the plant has n_y sensors and n_u actuators. In fact, the actuators and sensors are grouped into $\bar{N}$ nodes, where, in principle, it is allowed that a node can contain both sensors and actuators. The set of actuator and sensor indices corresponding to node $l \in \{1, ..., \bar{N}\}$ are denoted by

$$\bar{\mathbf{J}}_l^u \subseteq \{1, ..., n_u\}, \qquad \bar{\mathbf{J}}_l^y \subseteq \{1, ..., n_y\},$$

respectively.

At each transmission instant, only one node obtains access to the network and transmits its corresponding u and/or y values. Only the transmitted values will be updated, while all other values remain unchanged. This constrained data exchange can be expressed as

$$\hat{u}_k = \Gamma_{\sigma_k}^u u_k + (I - \Gamma_{\sigma_k}^u)\hat{u}_{k-1}, \tag{3.5a}$$

$$\hat{y}_k = \Gamma_{\sigma_k}^y y_k + (I - \Gamma_{\sigma_k}^y)\hat{y}_{k-1}, \tag{3.5b}$$

where the value of $\sigma_k \in \{1, ..., \bar{N}\}$ indicates which node is given access to the network at the transmission instant $k \in \mathbb{N}$, and $\Gamma_l^u \in \mathbb{R}^{n_u \times n_u}$ and $\Gamma_l^y \in \mathbb{R}^{n_y \times n_y}$, for $l \in \{1, ..., \bar{N}\}$, are diagonal matrices where

$$(\Gamma_l^u)_{i,i} := \begin{cases} 1, & \text{if } i \in \bar{\mathbf{J}}_l^u, \\ 0, & \text{otherwise,} \end{cases}$$
$$(\Gamma_l^y)_{i,i} := \begin{cases} 1, & \text{if } i \in \bar{\mathbf{J}}_l^y, \\ 0, & \text{otherwise.} \end{cases}$$

The mechanisms determining σ_k at transmission instant t_k are known as protocols. In this chapter, we focus on the general class of periodic protocols [38, 67], which are characterized by

$$\sigma_{k+\tilde{N}} = \sigma_k, \text{for all } k \in \mathbb{N}, \tag{3.6a}$$
$$\{\sigma_k |\ 1 \leq k \leq \tilde{N}\} \supseteq \{1, ..., \bar{N}\}, \tag{3.6b}$$

where $\tilde{N} \geq \bar{N}$ and $\tilde{N} \in \mathbb{N}$ is the period of the protocol. Note that $\{\sigma_k |\ 1 \leq k \leq \tilde{N}\} \supseteq \{1, ..., \bar{N}\}$ means that every node is addressed at least once in every period of the protocol. This condition is very natural as nodes that are never used do not need to be defined. The well-known Round Robin protocol [122] belongs to this class of periodic protocols, which is characterized by (3.6) and $\tilde{N} = \bar{N}$. Implementation of such a protocol can be accomplished by using the channel access method known as (multi-channel) time division multiple access (TDMA).

Remark 3.1. A commonly studied (dynamic) protocol in the NCS literature is the Try-Once-Discard (TOD) protocol, which was introduced in [122] and studied in Chapter 2 as well as in [34, 38, 58]. Although analysis of this protocol has shown improvement in the level of robustness with respect to network-induced effects (compared with periodic protocols), designing an output-based controller using a dynamic protocol is extremely challenging, even in the centralized setting, see, e.g. [34] in which constant transmission intervals have been considered. Considering the additional challenges that decentralization introduces into the NCS setting, in this paper, we choose to focus on periodic protocols for which an LMI-based design procedure will be offered. ◁

To characterize the decentralized NCS, we need to determine the sets of actuators and sensors that are associated with node $l \in \{1, ..., \bar{N}\}$ and belong to subsystem $i \in \{1, ..., N\}$. To achieve this we can use the structure present in the disjoint decomposition. Due to the fact that we consider the decomposition of (3.2) to be disjoint, as given in (3.1), we have that the input vector $\hat{u}_k$, output vector y_k and state vector x_k are ordered such that the set of indices corresponding to actuators $\hat{u}_k$, sensors y_k, and states x_k belonging to subsystem

i are defined as

$$\begin{aligned}\mathbf{J}_i^u &:= \Big\{ \textstyle\sum_{j=0}^{i-1} n_{u_j} + 1,\ \sum_{j=0}^{i-1} n_{u_j} + 2,\ \dots,\ \sum_{j=0}^{i} n_{u_j} \Big\},\\ \mathbf{J}_i^y &:= \Big\{ \textstyle\sum_{j=0}^{i-1} n_{y_j} + 1,\ \sum_{j=0}^{i-1} n_{y_j} + 2,\ \dots,\ \sum_{j=0}^{i} n_{y_j} \Big\},\\ \mathbf{J}_i^x &:= \Big\{ \textstyle\sum_{j=0}^{i-1} n_{x_j} + 1,\ \sum_{j=0}^{i-1} n_{x_j} + 2,\ \dots,\ \sum_{j=0}^{i} n_{x_j} \Big\},\end{aligned}$$

respectively, for $i \in \{1, ..., N\}$, where $n_{u_0} = n_{y_0} = n_{x_0} := 0$ and n_{u_i}, n_{y_i} and n_{x_i} denote the number of actuators, sensors and states, respectively, belonging to subsystem $i \in \{1, ..., N\}$. With these sets defined, we have that the set $\bar{\mathbf{J}}_l^u \cap \mathbf{J}_i^u$ consists exactly of the indices of the actuators which are associated with node l and belong to subsystem i. A similar interpretation holds for $\bar{\mathbf{J}}_l^y \cap \mathbf{J}_i^y$ regarding the indices of the sensors. We say that subsystem i is associated with node l if $\bar{\mathbf{J}}_l^u \cap \mathbf{J}_i^u \neq \emptyset$ or $\bar{\mathbf{J}}_l^y \cap \mathbf{J}_i^y \neq \emptyset$, meaning, that at least one sensor or actuator in node l belongs to subsystem i.

3.2.2 Decentralized Networked Observer-Based Controllers

In this chapter, we will use decentralized observer-based controllers in the sense that for each subsystem of the plant we have one observer-based controller which does not exchange information, see Fig. 3.1. Therefore, the individual observers have no information about externally coupled states, inputs, or outputs.

To obtain approximate discrete-time subsystem models for usage in the observer, we discretize (3.2) with a suitably chosen constant transmission interval $h_\star$ and then discard the subsystem coupling matrices (as the observers to be designed cannot use information about either the external coupling or the time-varying nature of the sampling interval). The resulting discrete-time plant model for the i^{th} subsystem is then

$$\mathcal{P}_{h_\star,i} : \left\{ \begin{array}{rcl} \check{x}_{k+1,i} &=& \bar{A}_{h_\star,i}\check{x}_{k,i} + \bar{B}_{h_\star,i}\hat{u}_{k,i},\\ \check{y}_{k,i} &=& C_i\check{x}_{k,i}, \end{array} \right. \tag{3.7}$$

for $i \in \{1, ..., N\}$, where $h_\star$ is a constant transmission interval, $\check{x}_{k,i} \in \mathbb{R}^{n_{x_i}}$, $\hat{u}_{k,i} := \hat{u}_i(t_k) \in \mathbb{R}^{n_{u_i}}$ and $\check{y}_{k,i} \in \mathbb{R}^{n_{y_i}}$ are the state, input and output vectors, respectively, of the i^{th} approximate discrete-time model at discrete time $k \in \mathbb{N}$, and $\bar{A}_{h_\star,i} := (\bar{A}_{h_\star})_{\mathbf{J}_i^x,\mathbf{J}_i^x}$ and $\bar{B}_{h_\star,i} := (\bar{B}_{h_\star})_{\mathbf{J}_i^x,\mathbf{J}_i^u}$ where $\bar{A}_{h_\star}$ and $\bar{B}_{h_\star}$ have been defined below (3.4). The observer-based controllers will use the approximate discrete-time subsystem models (3.7) that are based on the constant transmission interval $h_\star$, while the exact discrete-time model (3.4) corresponds to an uncertain and time-varying transmission interval h_k, which in general is not equal to $h_\star$. Hence, the variation in the transmission interval will act as a disturbance on the state estimation error dynamics as the observer model does not coincide with the

true model. Clearly, the coupling terms are neglected in (3.7) which contributes to a further difference between the models (3.7) and the true model (3.4). The designed observer and controller gains have to be designed in order to counteract these differences.

Using (3.7) as the i^{th} (approximate) subsystem model, we propose the i^{th} observer-based controller to be of the form

$$\mathcal{C}_{\sigma_k,i}: \left\{ \begin{array}{rcl} \tilde{x}_{k+1,i} & = & \bar{A}_{h_\star,i}\tilde{x}_{k,i} + \bar{B}_{h_\star,i}\hat{u}_{k,i} + L_{\sigma_k,i}\Gamma^y_{\sigma_k,i}(\hat{y}_{k,i} - C_i\tilde{x}_{k,i}), \\ u_{k,i} & = & K_{\sigma_k,i}\tilde{x}_{k,i}, \end{array} \right. \tag{3.8}$$

for $i \in \{1, ..., N\}$, where $\tilde{x}_{k,i} \in \mathbb{R}^{n_{x_i}}$, $\hat{y}_{k,i} \in \mathbb{R}^{n_{y_i}}$ and $u_{k,i} \in \mathbb{R}^{n_{u_i}}$ are the state estimate, input, and output vectors of the i^{th} observer-based controller at the discrete time $k \in \mathbb{N}$, respectively. The matrices $\Gamma^u_{l,i} := (\Gamma^u_l)_{\mathbf{J}^u_i,\mathbf{J}^u_i}$, $\Gamma^y_{l,i} := (\Gamma^y_l)_{\mathbf{J}^y_i,\mathbf{J}^y_i}$, $L_{l,i} \in \mathbb{R}^{n_{x_i}\times n_{y_i}}$ and $K_{l,i} \in \mathbb{R}^{n_{u_i}\times n_{x_i}}$ are defined for $i \in \{1, .., N\}$, $l \in \{1, ..., \bar{N}\}$, where $L_{l,i}$ and $K_{l,i}$ are the observer and controller gains, respectively. Hence, we adopt a switched observer and controller structure (notice the σ_k-dependence of $L_{\sigma_k,i}$ and $K_{\sigma_k,i}$ in (3.8)) to deal with the communication medium being shared. The presence of $\Gamma^y_{\sigma_k,i}$ in (3.8) is used so that the standard output injection is only applied to the newly received measurements. If no measurements are received from subsystem i at transmission time t_k (i.e. $\Gamma^y_{\sigma_k,i} = 0$) then (3.8) reduces to a standard model-based prediction step (according to the model in (3.7)).

Similar to the plant, the dynamics of all the controllers (3.8) can be described by a single discrete-time system, which will consist of block diagonal matrices due to the decoupled nature of the controllers:

$$\mathcal{C}_{\sigma_k}: \left\{ \begin{array}{rcl} \tilde{x}_{k+1} & = & \bar{A}_D\tilde{x}_k + \bar{B}_D\hat{u}_k + L_{\sigma_k}\Gamma^y_{\sigma_k}(\hat{y}_k - C_D\tilde{x}_k) \\ u_k & = & K_{\sigma_k}\tilde{x}_k, \end{array} \right. \tag{3.9}$$

where $\bar{A}_D := \mathrm{diag}(\bar{A}_{h_\star,1}, \bar{A}_{h_\star,2}, ..., \bar{A}_{h_\star,N})$, $\bar{B}_D$ and C_D defined similarly, and the observer gains

$$L_l = \mathrm{diag}(L_{l,1}, L_{l,2}, ..., L_{l,N}), \quad \text{for } l \in \{1, ..., \bar{N}\}, \tag{3.10a}$$

$$K_l = \mathrm{diag}(K_{l,1}, K_{l,2}, ..., K_{l,N}), \quad \text{for } l \in \{1, ..., \bar{N}\}. \tag{3.10b}$$

3.2.3 Closed-Loop Model

To derive an expression for the closed-loop dynamics, we will adopt the state vector

$$\bar{x}_k = [e_k^\top \quad x_k^\top \quad \hat{u}_{k-1}^\top \quad \hat{y}_{k-1}^\top]^\top \in \mathbb{R}^n,$$

where e_k denotes the state estimation error defined as $e_k := \tilde{x}_k - x_k$, $k \in \mathbb{N}$, and $n = 2n_x + n_u + n_y$. Combining (3.4), (3.5), and (3.9) results in the overall closed-loop system

$$\bar{x}_{k+1} = \tilde{A}_{\sigma_k,h_k}\bar{x}_k, \tag{3.11}$$

where

$$\tilde{A}_{l,h} = \begin{bmatrix} \Psi_{1,1}(l,h) & \Psi_{1,2}(l,h) \\ \Psi_{2,1}(l) & \Psi_{2,2}(l) \end{bmatrix}, \tag{3.12}$$

and

$$\Psi_{1,1}(l,h) := \begin{bmatrix} \bar{A}_D - L_l\Gamma_l^y C_D + (\bar{B}_D - \bar{B}_h)\Gamma_l^u K_l & (\bar{A}_D - \bar{A}_h) - L_l\Gamma_l^y(C_D - C) + (\bar{B}_D - \bar{B}_h)\Gamma_l^u K_l \\ \bar{B}_h\Gamma_l^u K_l & \bar{A}_h + \bar{B}_h\Gamma_l^u K_l \end{bmatrix},$$

$$\Psi_{1,2}(l,h) := \begin{bmatrix} (\bar{B}_D - \bar{B}_h)(I - \Gamma_l^u) & 0 \\ \bar{B}_h(I - \Gamma_l^u) & 0 \end{bmatrix}, \qquad \Psi_{2,1}(l) := \begin{bmatrix} \Gamma_l^u K_l & \Gamma_l^u K_l \\ 0 & \Gamma_l^y C \end{bmatrix},$$

$$\Psi_{2,2}(l) := \begin{bmatrix} (I - \Gamma_l^u) & 0 \\ 0 & (I - \Gamma_l^y) \end{bmatrix},$$

$l \in \{1, ..., \bar{N}\}$, and $h \in [h_{min}, h_{max}]$. In deriving (3.12) note that $\Gamma_{\sigma_k}^y(I - \Gamma_{\sigma_k}^y) = 0$ was used. The closed-loop system (3.11) is a discrete-time switched linear parameter-varying (SLPV) system where the switching, as given by σ_k, is due to the communication medium being shared and the parameter uncertainty is caused by the uncertainty in the transmission interval $h_k \in [h_{min}, h_{max}]$.

3.3 Polytopic Overapproximation

In the previous section, we obtained a decentralized NCS model in the form of a switched uncertain system. However, the form as in (3.11), (3.12) is not convenient to develop efficient techniques for stability analysis and controller synthesis due to the nonlinear dependence of $\tilde{A}_{\sigma_k,h_k}$ in (3.12) on the uncertain parameter h_k. To make the system amenable for analysis and synthesis, a procedure is proposed to overapproximate system (3.11), (3.12) by a polytopic system with norm-bounded additive uncertainty, i.e.,

$$\bar{x}_{k+1} = \sum_{m=1}^{M} \alpha_k^m \left(\mathcal{F}_{\sigma_k,m} + \mathcal{G}_m \Delta_k \mathcal{H}_{\sigma_k}\right) \bar{x}_k, \tag{3.13}$$

where $\mathcal{F}_{l,m} \in \mathbb{R}^{n\times n}$, $\mathcal{G}_m \in \mathbb{R}^{n\times 2n_x}$, $\mathcal{H}_l \in \mathbb{R}^{2n_x\times n}$, for $l \in \{1, \ldots, \bar{N}\}$ and $m \in \{1, \ldots, M\}$, with M the number of vertices of the polytope. The vector $\alpha_k = [\alpha_k^1 \ \ldots \ \alpha_k^M]^\top \in \mathbf{\Omega}$, for all $k \in \mathbb{N}$, is time-varying with

$$\mathbf{\Omega} = \left\{ \alpha \in \mathbb{R}^M \,\middle|\, \textstyle\sum_{m=1}^M \alpha^m = 1 \text{ and } \alpha^m \geq 0, \ \text{ for } \ m \in \{1, \ldots, M\} \right\} \tag{3.14}$$

and $\Delta_k \in \mathbf{\Delta}$, for all $k \in \mathbb{N}$, with the additive uncertainty set $\mathbf{\Delta} \subseteq \mathbb{R}^{2n_x\times 2n_x}$ given by

$$\mathbf{\Delta} = \big\{ \mathrm{diag}(\Delta^1, \ldots, \Delta^{2Q}) \mid \Delta^{q+jQ} \in \mathbb{R}^{n_{\lambda_q}\times n_{\lambda_q}}, \\ \|\Delta^{q+jQ}\| \leq 1, q \in \{1, \ldots, Q\}, j \in \{0, 1\} \big\}, \tag{3.15}$$

where $n_{\lambda_q} \times n_{\lambda_q}$, $q \in \{1, ..., Q\}$, are the dimensions of the q^{th} real Jordan block [68] of A and Q is the number of real Jordan blocks of A. See Procedure 3.1 for below for more details regarding this overapproximation. System (3.13) is an overapproximation of system (3.11) in the sense that for all $l \in \{1, \ldots, \bar{N}\}$, it holds that

$$\left\{\tilde{A}_{l,h} \mid h \in [h_{min}, h_{max}]\right\} \subseteq \left\{ \textstyle\sum_{m=1}^{M} \alpha^m \left(\mathcal{F}_{l,m} + \mathcal{G}_m \Delta \mathcal{H}_l\right) \mid \alpha \in \mathbf{\Omega}, \Delta \in \mathbf{\Delta} \right\}. \tag{3.16}$$

Due to this inclusion, stability of (3.13) with $\alpha_k \in \mathbf{\Omega}$ and $\Delta_k \in \mathbf{\Delta}, k \in \mathbb{N}$, implies stability of (3.11) for $h_k \in [h_{min}, h_{max}]$. Although many overapproximation techniques are available, see e.g. the survey [59], here we provide a gridding-based procedure based on [38] to overapproximate system (3.11), such that (3.16) holds. This choice is motivated by the favorable properties that this method possesses, see [59].

The overapproximation procedure we present here is a slight modification of the procedure presented in [38]. Namely, the procedure presented in [38] is for an NCS model that includes bounded time-varying delays, as well as bounded time-varying transmission intervals, while the procedure presented here is for an NCS model that only includes bounded time-varying transmission intervals. As such, the NCS model given here has fewer nonlinear uncertainty terms that require overapproximation. However, including bounded time-varying delays in the NCS model (as in [38]) is a straightforward extension, see Remark 3.6 below. Moreover, [38] uses a parameter $\epsilon_u > 0$, which is defined as the user-specified tightness of the overapproximation, to determine when to terminate the procedure. For reasons related to controller synthesis, we cannot use this criterion. Instead, we use a user-specified number of grid points to determine when to terminate the procedure. Both the procedure presented in this paper, as well as the one presented in [38], iteratively place grid points at the locations of the worst-case approximation error and, hence, in both procedures, the overapproximation becomes tighter when either the user-specified number of grid points increased or the user-specified tightness is decreased, respectively.

Procedure 3.1.

1. *Select a desired number of grid points $M_{des} \geq 2$.*
2. *Decompose the matrix A, as given in* (3.2)*, into its real Jordan form [68], i.e. $A := T\Lambda T^{-1}$, where T is an invertible matrix and*

$$\Lambda = \operatorname{diag}(\Lambda_1, \ldots, \Lambda_Q) \tag{3.17}$$

   with $\Lambda_q \in \mathbb{R}^{n_{\lambda_q} \times n_{\lambda_q}}$, $q \in \{1, \ldots, Q\}$, the q^{th} real Jordan block of A.
3. *Define the set of grid points $\mathbf{G} := \{\tilde{h}_1, \tilde{h}_2\}$, where $\tilde{h}_1 =: h_{min}$ and $\tilde{h}_2 := h_{max}$, and define $M := 2$.*

4. *Compute for each line segment* $S_\ell = [\tilde{h}_\ell, \tilde{h}_{\ell+1}]$, $\ell \in \{1, \dots, M-1\}$, *and for each real Jordan block* Λ_i, $q \in \{1, \dots, Q\}$, *the worst case approximation error, i.e.*

$$\tilde{\delta}^A_{q,\ell} = \max_{\tilde{\alpha}_1+\tilde{\alpha}_2=1,\ \tilde{\alpha}_1,\tilde{\alpha}_2\geq 0} \hat{\delta}^A_{q,\ell,\tilde{\alpha}_1,\tilde{\alpha}_2}, \tag{3.18a}$$

$$\tilde{\delta}^E_{q,\ell} = \max_{\tilde{\alpha}_1+\tilde{\alpha}_2=1,\ \tilde{\alpha}_1,\tilde{\alpha}_2\geq 0} \hat{\delta}^E_{q,\ell,\tilde{\alpha}_1,\tilde{\alpha}_2}, \tag{3.18b}$$

where

$$\hat{\delta}^A_{q,\ell,\tilde{\alpha}_1,\tilde{\alpha}_2} := \left\| e^{\Lambda_q(\tilde{\alpha}_1\tilde{h}_\ell+\tilde{\alpha}_2\tilde{h}_{\ell+1})} - \sum_{j=1}^{2} \tilde{\alpha}_j e^{\Lambda_q \tilde{h}_{\ell+j-1}} \right\|,$$

$$\hat{\delta}^E_{q,\ell,\tilde{\alpha}_1,\tilde{\alpha}_2} := \left\| \sum_{j=1}^{2} \tilde{\alpha}_j \int_{\tilde{h}_{\ell+j-1}}^{\tilde{\alpha}_1\tilde{h}_\ell+\tilde{\alpha}_2\tilde{h}_{\ell+1}} e^{\Lambda_q s} ds \right\|.$$

For a detailed explanation of the origin of the approximation error bounds, see [38].

5. *If the number of grid points equals the desired number of grid points* ($M = M_{des}$), *go to step 9, else go to step 6.*

6. *Place an additional grid point at the location of the worst case approximation error. The new grid point is defined as* $h_{new} := \tilde{\alpha}^\star_1 \tilde{h}_{\ell^\star} + \tilde{\alpha}^\star_2 \tilde{h}_{\ell^\star+1}$ *for a line segment* $\ell^\star$ *and constants* $\tilde{\alpha}^\star_1$, $\tilde{\alpha}^\star_2 \geq 0$ *where* $\tilde{\alpha}^\star_1 + \tilde{\alpha}^\star_2 = 1$ *determined as follows. If* $\max_{q,\ell} \tilde{\delta}^E_{q,\ell} \geq \max_{q,\ell} \tilde{\delta}^A_{q,\ell}$ *then* $\tilde{\alpha}^\star_1$, $\tilde{\alpha}^\star_2$ *and* $\ell^\star$ *are given by* $(q^\star, \ell^\star) = \arg\max \tilde{\delta}^E_{q,\ell}$ *and*

$$(\tilde{\alpha}^\star_1, \tilde{\alpha}^\star_2) \in \operatorname*{arg\,max}_{\tilde{\alpha}_1+\tilde{\alpha}_2=1,\ \tilde{\alpha}_1,\tilde{\alpha}_2\geq 0} \hat{\delta}^E_{q^\star,\ell^\star,\tilde{\alpha}_1,\tilde{\alpha}_2},$$

else $\tilde{\alpha}^\star_1$, $\tilde{\alpha}^\star_2$ *and* $\ell^\star$ *are given by* $(q^\star, \ell^\star) = \arg\max \tilde{\delta}^A_{q,\ell}$ *and*

$$(\tilde{\alpha}^\star_1, \tilde{\alpha}^\star_2) \in \operatorname*{arg\,max}_{\tilde{\alpha}_1+\tilde{\alpha}_2=1,\ \tilde{\alpha}_1,\tilde{\alpha}_2\geq 0} \hat{\delta}^A_{q^\star,\ell^\star,\tilde{\alpha}_1,\tilde{\alpha}_2}.$$

7. *Set* $M := M + 1$ *and* $\mathbf{G} := \mathbf{G} \cup \{h_{new}\}$.

8. *Order the grid points according to* $\mathbf{G} = \{\tilde{h}_1, ...\tilde{h}_M\}$ *such that* $h_{min} := \tilde{h}_1 < \tilde{h}_2 < ... < \tilde{h}_{M-1} < \tilde{h}_M := h_{max}$ *and go to step 4.*

9. *Map the obtained bounds* (3.18) *at each line segment* S_ℓ, $\ell \in \{1, \dots, M-1\}$, *for each Jordan block* Λ_q, $q \in \{1, \dots, Q\}$, *to their corresponding vertices* $m \in \{1, \dots, M\}$, *according to*

$$\delta^A_{q,m} = \max\{\tilde{\delta}^A_{q,m-1}, \tilde{\delta}^A_{q,m}\}, \quad \delta^E_{q,m} = \max\{\tilde{\delta}^E_{q,m-1}, \tilde{\delta}^E_{q,m}\},$$

where $\tilde{\delta}^A_{q,0} = \tilde{\delta}^A_{q,M} = \tilde{\delta}^E_{q,0} = \tilde{\delta}^E_{q,M} := 0$ *for all* $q \in \{1, \dots, Q\}$.

10. *Finally, define* $\mathcal{F}_{l,m} := \tilde{A}_{l,\tilde{h}_m}$, *for* $l \in \{1, ..., \bar{N}\}$, $m \in \{1, ..., M\}$, *and define*

$$\mathcal{H}_l := \begin{bmatrix} 0 & T^{-1} & 0 & 0 \\ T^{-1}B\Gamma_l^u K_l & T^{-1}B\Gamma_l^u K_l & T^{-1}B(I-\Gamma_l^u) & 0 \end{bmatrix}, \tag{3.20}$$

for $l \in \{1, ..., \bar{N}\}$, *and*

$$\mathcal{G}_m := \begin{bmatrix} -T & -T \\ T & T \\ 0 & 0 \\ 0 & 0 \end{bmatrix} U_m, \quad \text{for } m \in \{1, ..., M\}, \tag{3.21}$$

in which

$$U_m = \operatorname{diag}(\delta_{1,m}^A I_1, \ldots, \delta_{Q,m}^A I_Q, \delta_{1,m}^E I_1, \ldots, \delta_{Q,m}^E I_Q) \tag{3.22}$$

where I_q *is the* $n_{\lambda_q} \times n_{\lambda_q}$ *identity matrix and* B *is given in* (3.2).

Theorem 3.1. *Consider system* (3.11), *where* $h_k \in [h_{min}, h_{max}], k \in \mathbb{N}$, *and* $\tilde{A}_{l,h}$, $l \in \{1, ..., \bar{N}\}$, $h \in \mathbb{R}$, *is given as in* (3.12). *If system* (3.13) *is obtained by following Procedure 3.1,* (3.13) *is an overapproximation of* (3.11) *in the sense that* (3.16) *holds for all* $l \in \{1, ..., \bar{N}\}$.

Proof. The proof can be obtained along the lines of the proof of [38, Theorem III.2]. □

We care to stress that the most appealing aspect of this particular overapproximation technique is the fact that it introduces arbitrarily little conservatism when employed in Lyapunov-based stability analysis. More specifically, in [38], it was proven that if the original system (without any overapproximation), is uniformly globally exponentially stable (UGES) in the sense that a parameter-dependent quadratic Lyapunov function exists, the presented LMI-based stability check based on the overapproximation will guarantee UGES and will find a respective parameter-dependent quadratic Lyapunov function, given that the overapproximation consists of a collection of grid points which are sufficiently refined, i.e. the desired number of grid points in Procedure 3.1 sufficiently large (see [38, Theorem V.1]). Therefore, in this sense, no conservatism is introduced by making a convex overapproximation according to Procedure 3.1.

Remark 3.2. The stability analysis problem, i.e. determining whether the system (3.11), (3.12) with *given* controller gains K_l, L_l, $l \in \{1, ..., \bar{N}\}$, is uniformly globally exponentially stable (UGES) for a given scheduling protocol, as in (3.6), and given bounds on the transmission interval, i.e. $h_k \in [h_{min}, h_{max}]$ for all $k \in \mathbb{N}$, can be addressed by using the overapproximated model (3.13) combined with the proposed LMI conditions in [38]. The focus of the current chapter is on the more challenging problem of controller synthesis, see Section 3.4. ◁

Remark 3.3. Using a reasoning similar as in [93], it can be shown that UGES of the discrete-time model (3.11), (3.12) with a protocol satisfying (3.6) implies UGES of the sampled-data NCS (3.2), (3.3), (3.5), (3.9), with the same protocol and including the intersample behavior. ◁

3.4 Controller Synthesis

In the previous sections, we derived a model describing an LTI plant interconnected with a decentralized switched observer-based output-feedback controller by a communication network. In this section, we will present the main contribution of this chapter consisting of LMI-based conditions for *designing* the decentralized controller and observer gains K_l and L_l, respectively, in (3.9) by using the overapproximated model (3.13).

For reasons of transparency, we choose to divide the presentation of our solution into two sections. In Section 3.4.1, LMI conditions which synthesize stabilizing controllers are derived for the case when the subsystems are restricted to communicate in a serial fashion (i.e. only one subsystem is allowed to communicate at each transmission instant). Then, in Section 3.4.2, the foundation laid in Section 3.4.1 is built upon to derive LMI conditions which synthesize stabilizing controllers for the more general case when the subsystems can communicate (also) in parallel (i.e. multiple subsystems are allowed to communicate at each transmission instant).

3.4.1 Serial Subsystem Communication

In this section, we will adopt the network assumption presented below. Next to the transparency reason already given before, a second reason to treat the case corresponding to this assumption separately, is that it represents a relevant subclass of the synthesis problem. In Section 3.4.2, details are provided regarding how this assumption can be removed.

Assumption 3.1. *All sensors or actuators associated with a node must be members of the same subsystem, i.e. for each node* $l \in \{1, ..., \bar{N}\}$*, there exists a subsystem* $i \in \{1, ..., N\}$ *such that* $\bar{\mathbf{J}}_l^u \subseteq \mathbf{J}_i^u$ *and* $\bar{\mathbf{J}}_l^y \subseteq \mathbf{J}_i^y$.

Remark 3.4. Due to Assumption 3.1, only one subsystem can communicate (a part of) its corresponding signals at each transmission time. Indeed, when node $l \in \{1, ..., \bar{N}\}$ attains access to the network, only one (corresponding) subsystem $i \in \{1, ..., N\}$ can communicate over the network and, hence, one gain $K_{l,i}$ and one gain $L_{l,i}$, as in (3.10), influence the closed loop dynamics given by either (3.11) or (3.16) (due to the presence of $\Gamma_{\sigma_k}^y$ in (3.9) and the fact that $\hat{u}_k$, given in (3.5), is the input to the plant). As a consequence, some of the gains $K_{l,i}$ and $L_{l,i}$, which are defined for all $i \in \{1, ..., N\}$, have no influence when node

$l \in \{1, ..., \bar{N}\}$ attains access (in fact all of them that do not correspond to node l will have no influence). We care to stress that we explicitly account for this fact in the synthesis theorems in Section 3.4. Moreover, we choose to keep the more general definitions, as in (3.10), since we provide explicit details on how to remove Assumption 3.1 in Section 3.4.2 (meaning that possibly all of the gains $K_{l,i}$ and $L_{l,i}$, $i \in \{1, ..., N\}$, influence the closed-loop dynamics when node l communicates). ◁

Before we can use the overapproximated model (3.13) for synthesis, an essential step must be taken so that the model (3.13) can be rewritten in a form which is suitable for controller synthesis. The essential step in achieving LMI-based synthesis conditions is reformulating (3.12) such that the design variables are non-structured matrices, instead of the structured (block diagonal) matrices K_{σ_k} and L_{σ_k}, as in (3.10), respectively, that are currently present. To achieve this, we first introduce the set of state indices belonging to subsystems associated with node l as

$$\bar{\mathbf{J}}_l^x \subseteq \{1, ..., n_x\}.$$

Due to Assumption 3.1, only one subsystem i is associated with node l and, hence, $\bar{\mathbf{J}}_l^x$ is the set of the state indices corresponding to the subsystem i that is associated with node l (i.e. if i is the subsystem associated with node l then $\bar{\mathbf{J}}_l^x = \mathbf{J}_i^x$). With these sets defined, we introduce

$$\Upsilon_l^u := \begin{cases} (\Gamma_l^u)_{\bullet, \bar{\mathbf{J}}_l^u}, & \text{if } l \in \mathbf{L}_u, \\ 0, & \text{otherwise}, \end{cases} \tag{3.23a}$$

$$\Upsilon_l^y := \begin{cases} (\Gamma_l^y)_{\bullet, \bar{\mathbf{J}}_l^y}, & \text{if } l \in \mathbf{L}_y, \\ 0, & \text{otherwise}, \end{cases} \tag{3.23b}$$

$$\Upsilon_l^x := (I^x)_{\bullet, \bar{\mathbf{J}}_l^x}, \tag{3.23c}$$

for $l \in \{1, ..., \bar{N}\}$, where $I^x \in \mathbb{R}^{n_x \times n_x}$ is the identity matrix and

$$\mathbf{L}_u := \{l \in \{1, ..., \bar{N}\} \mid \bar{\mathbf{J}}_l^u \neq \emptyset\}, \tag{3.24a}$$

$$\mathbf{L}_y := \{l \in \{1, ..., \bar{N}\} \mid \bar{\mathbf{J}}_l^y \neq \emptyset\}, \tag{3.24b}$$

are the sets of node indices that contain at least one actuator or sensor, respectively. Note that Υ_l^u and Υ_l^y are simply matrices consisting of the non-zero columns of Γ_l^u and Γ_l^y, respectively. Finally, we define

$$\bar{K}_l := \begin{cases} (K_l)_{\bar{\mathbf{J}}_l^u, \bar{\mathbf{J}}_l^x}, & \text{if } l \in \mathbf{L}_u, \\ 0, & \text{otherwise}, \end{cases} \tag{3.25a}$$

$$\bar{L}_l := \begin{cases} (L_l)_{\bar{\mathbf{J}}_l^x, \bar{\mathbf{J}}_l^y}, & \text{if } \ l \in \mathbf{L}_y, \\ 0, & \text{otherwise}, \end{cases} \tag{3.25b}$$

for $l \in \{1, ..., \bar{N}\}$. Notice that $\bar{K}_l$ and $\bar{L}_l$ consist of all the non-restricted elements of $\Gamma_l^u K_l$ and $L_l \Gamma_l^y$, respectively. With these matrices defined and Assumption 3.1

adopted, we have that the following equations hold

$$\Gamma_l^u K_l = \Upsilon_l^u \bar{K}_l \Upsilon_l^{x\top}, \qquad L_l \Gamma_l^y = \Upsilon_l^x \bar{L}_l \Upsilon_l^{y\top}. \tag{3.26}$$

Now, (3.26) allows the closed-loop matrix $\tilde{A}_{\sigma_k,h_k}$ in (3.12) to be expressed in terms of the non-structured matrices $\bar{K}_{\sigma_k}$ and $\bar{L}_{\sigma_k}$ instead of the structured (block diagonal) matrices K_{σ_k} and L_{σ_k}. The benefit of this is that the synthesis problem for decentralized control, which naturally imposes 'structural' constraints, can now be formulated as a 'non-structured synthesis' problem when Assumption 3.1 is adopted. To help convey (3.23), (3.25) and (3.26), we will explicitly define these matrices for the example given in Section 3.5.

Recall that after employment of the overapproximation technique described in Section 3.3, we now have a system of the form (3.13), where the matrices $\mathcal{F}_{l,m} = \tilde{A}_{l,\tilde{h}_m}$ are given by (3.12) with $h \in \{\tilde{h}_1, ..., \tilde{h}_m\}$ and the matrices $\mathcal{H}_l$ and $\mathcal{G}_m$ are given in (3.20) and (3.21), respectively. Using (3.26), we can decompose $\mathcal{F}_{l,m}$ and $\mathcal{H}_l$ in the following way

$$\mathcal{F}_{l,m} = \mathcal{A}_{l,m} + \mathcal{B}_{l,m} \bar{K}_l \mathcal{E}_l - \mathcal{D}_l \bar{L}_l \mathcal{C}_l, \tag{3.27a}$$

$$\mathcal{H}_l = \mathcal{I}_l + \mathcal{J}_l \bar{K}_l \mathcal{E}_l, \tag{3.27b}$$

where

$$\mathcal{A}_{l,m} := \begin{bmatrix} \bar{A}_D & \bar{A}_D - \bar{A}_{\tilde{h}_m} & (\bar{B}_D - \bar{B}_{\tilde{h}_m})(I - \Gamma_l^u) & 0 \\ 0 & \bar{A}_{\tilde{h}_m} & \bar{B}_{\tilde{h}_m}(I - \Gamma_l^u) & 0 \\ 0 & 0 & I - \Gamma_l^u & 0 \\ 0 & \Gamma_l^y C & 0 & I - \Gamma_l^y \end{bmatrix}, \tag{3.28a}$$

$$\mathcal{B}_{l,m} := \begin{bmatrix} (\bar{B}_D - \bar{B}_{\tilde{h}_m})\Upsilon_l^u \\ \bar{B}_{\tilde{h}_m} \Upsilon_l^u \\ \Upsilon_l^u \\ 0 \end{bmatrix}, \quad \mathcal{E}_l := \begin{bmatrix} \Upsilon_l^{x\top} & \Upsilon_l^{x\top} & 0 & 0 \end{bmatrix}, \tag{3.28b}$$

$$\mathcal{D}_l := \begin{bmatrix} \Upsilon_l^x \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathcal{C}_l := \begin{bmatrix} \Upsilon_l^{y\top} C_D & \Upsilon_l^{y\top}(C_D - C) & 0 & 0 \end{bmatrix}, \tag{3.28c}$$

$$\mathcal{I}_l := \begin{bmatrix} 0 & T^{-1} & 0 & 0 \\ 0 & 0 & T^{-1} B (I - \Gamma_l^u) & 0 \end{bmatrix}, \quad \mathcal{J}_l := \begin{bmatrix} 0 \\ T^{-1} B \Upsilon_l^u \end{bmatrix}. \tag{3.28d}$$

Now we are ready to state our main result. Notice that (3.13) with (3.27) describes a discrete-time switched linear parameter-varying (SLPV) system with norm-bounded uncertainty. No results are available in the literature to synthesize the controller gains $\bar{K}_l$ and $\bar{L}_l$, at present. However, LMI-based synthesis conditions can be obtained by generalizing the results in [33,116] in three directions. In particular, the first extension is the accommodation of norm-bounded

uncertainty $\mathcal{G}_m \Delta \mathcal{H}_{\sigma_k}$ in (3.13), where $\Delta \in \mathbf{\Delta}$, and the second extension is that the switching sequence (3.6) that we consider is ordered (periodic in this case), whereas [33] considered the case of arbitrary switching. Finally, the third extension is generalizing the set of LMI-based conditions in [33] so that solutions for the *multi-gain* switched static output-feedback problem can be included. Although the required extensions of the ideas in [33] contribute toward our main result, we would like to emphasize that using (3.26) for the formulation of (3.27) is the foundation upon which our main result is built. Stabilizing controller and observer gains K_l and L_l for the NCS given by (3.11) with $h_k \in [h_{min}, h_{max}]$ and a protocol satisfying (3.6) can be synthesized according to the following theorem. In the formulation of the theorem, the matrix set

$$\mathbf{R} := \Big\{ \operatorname{diag}(r_1 I_1, \ldots, r_Q I_Q, r_{Q+1} I_1, \ldots, r_{2Q} I_Q) \in \mathbb{R}^{2n_x \times 2n_x} \mid r_{\tilde{q}} \in \mathbb{R}, \;\; r_{\tilde{q}} > 0, \; \tilde{q} \in \{1, 2, ..., 2Q\} \Big\} \tag{3.29}$$

is used, where, as in (3.22), I_q is the $n_{\lambda_q} \times n_{\lambda_q}$ identity matrix.

Theorem 3.2. *Consider the system* (3.11), (3.12) *with* $h_k \in [h_{min}, h_{max}]$, $k \in \mathbb{N}$, *and its overapproximation given by* (3.13), (3.21), (3.27). *Assume that Assumption 3.1 holds, the protocol satisfies* (3.6) *and any node* $l \in \{1, ..., \bar{N}\}$ *containing at least one sensor, i.e.* $\bar{\mathbf{J}}_l^y \neq \emptyset$, *consists of linearly independent sensors, i.e.* $(C)_{\bar{\mathbf{J}}_l^y, \bullet}$ *has full row rank. Suppose there exist symmetric matrices* P_j, *matrices* $R_{j,m} \in \mathbf{R}$, *with* $\mathbf{R}$ *as in* (3.29), *and matrices* G_l, $Z_{1,l}$, $Z_{2,l}$, $X_{1,l}$ *and* $X_{2,l}$ *where* $j \in \{1, ..., \tilde{N}\}$, $m \in \{1, ..., M\}$, $l \in \{1, ..., \bar{N}\}$ *such that*

$$\left[\begin{array}{cc|cc} G_{\sigma_j} + G_{\sigma_j}^\top - P_j & \Xi_1(j,m)^\top & 0 & \Xi_2(j)^\top \\ \star & P_{j+1} & \mathcal{G}_m R_{j,m} & 0 \\ \hline \star & \star & R_{j,m} & 0 \\ \star & \star & \star & R_{j,m} \end{array}\right] \succ 0, \tag{3.30}$$

for $j \in \{1, ..., \tilde{N}\}$, $m \in \{1, ..., M\}$, *and*

$$X_{1,l} \mathcal{E}_l = \mathcal{E}_l G_l, \quad \text{for } l \in \mathbf{L}_u, \tag{3.31a}$$

$$X_{2,l} \mathcal{C}_l = \mathcal{C}_l G_l, \quad \text{for } l \in \mathbf{L}_y, \tag{3.31b}$$

for which we define

$$\Xi_1(j,m) := \mathcal{A}_{\sigma_j,m} G_{\sigma_j} + \mathcal{B}_{\sigma_j,m} Z_{1,\sigma_j} \mathcal{E}_{\sigma_j} - \mathcal{D}_{\sigma_j} Z_{2,\sigma_j} \mathcal{C}_{\sigma_j},$$
$$\Xi_2(j) := \mathcal{I}_{\sigma_j} G_{\sigma_j} + \mathcal{J}_{\sigma_j} Z_{1,\sigma_j} \mathcal{E}_{\sigma_j},$$

for $j \in \{1, ..., \tilde{N}\}$, $m \in \{1, ..., M\}$, *with* $P_{\tilde{N}+1} := P_1$ *and the sets* $\mathbf{L}_u$ *and* $\mathbf{L}_y$ *are defined in* (3.24), *respectively. Then the controller gains* K_l, *defined by* (3.25), (3.26) *and* $\bar{K}_l = Z_{1,l} X_{1,l}^{-1}$, $l \in \mathbf{L}_u$, *and the observer gains, defined by* (3.25), (3.26) *and* $\bar{L}_l = Z_{2,l} X_{2,l}^{-1}$, $l \in \mathbf{L}_y$, *render the system* (3.11), (3.12), *with* $h_k \in [h_{min}, h_{max}]$, $k \in \mathbb{N}$, *and the mentioned periodic protocol, UGES.*

Proof. Before going to the construction of a Lyapunov function to prove the theorem, we first establish some technical facts that we need in the sequel:

- Due to (3.30), $P_j \succ 0$ for $j \in \{1, ..., \tilde{N}\}$.
- Feasibility of (3.30) implies that $G_{\sigma_j} + G_{\sigma_j}^\top - P_j \succ 0$, and thus G_l is invertible for all $l \in \{1, ..., \bar{N}\}$. Indeed, suppose that $G_{\sigma_j}\bar{x} = 0$ for some $\bar{x}$, then $0 = \bar{x}^\top (G_{\sigma_j} + G_{\sigma_j}^\top)\bar{x} \succeq \bar{x}^\top P_j \bar{x}$. Since $P_j \succ 0$, this implies that $\bar{x} = 0$ and thus G_l, $l \in \{1, ..., \bar{N}\}$, is invertible.
- Using the fact that $\mathcal{C}_l$ has full row rank when $(C)_{\bar{\mathbf{J}}_l^y, \bullet} = {\Upsilon_l^y}^\top C$ has full row rank and $\mathcal{E}_l$ is full row rank by definition, then it follows from (3.31) and invertibility of G_l, $l \in \{1, ..., \bar{N}\}$, that $X_{1,l}$, $l \in \mathbf{L}_u$, must have full rank and $X_{2,l}$, $l \in \mathbf{L}_y$, must have full rank and thus be invertible. Hence, the controller gains $\bar{K}_l = Z_{1,l} X_{1,l}^{-1}$, $l \in \mathbf{L}_u$, and observer gains $\bar{L}_l = Z_{2,l} X_{2,l}^{-1}$, $l \in \mathbf{L}_y$, are well defined.

Now we are ready to prove that the controller gains $\bar{K}_l = Z_{1,l} V_{1,l}^{-1}$, $l \in \mathbf{L}_u$, and observer gains $\bar{L}_l = Z_{2,l} V_{2,l}^{-1}$, $l \in \mathbf{L}_y$, with (3.25) and (3.26) stabilize (3.11), (3.12) with $h_k \in [h_{min}, h_{max}]$ and a given protocol satisfying (3.6) by proving that (3.30) and (3.31) guarantee the existence of a Lyapunov function proving uniform global exponential stability (UGES) of (3.13), (3.21), (3.27) with $\alpha_k \in \mathbf{\Omega}$, $\Delta_k \in \mathbf{\Delta}$ and the same protocol satisfying (3.6). This is a direct consequence as (3.13) is an overapproximation of (3.11), (3.12) in the sense that (3.16) holds.

Let us consider the following Lyapunov function candidate

$$V_k(\bar{x}_k) = \bar{x}_k^\top P_j^{-1} \bar{x}_k, \tag{3.32}$$

where $j = k - r\tilde{N}$ for some $r \in \mathbb{N}$ such that $j \in \{1, ..., \tilde{N}\}$. Clearly, there exist $c_1, c_2 > 0$ such that $c_1 ||\bar{x}||^2 \leq V_k(\bar{x}) \leq c_2 ||\bar{x}||^2$ for all $k \in \mathbb{N}$ and $\bar{x} \in \mathbb{R}^n$ due to the positive definiteness of $P_1^{-1}, ..., P_{\tilde{N}}^{-1}$. Due to the fact that σ_k is periodic, see (3.6), we only have to show that the Lyapunov function candidate decreases along solutions of (3.13), (3.21), (3.27) for $k = \{1, ..., \tilde{N}\}$. UGES of (3.13), (3.21), (3.27) is established if this Lyapunov function candidate satisfies

$$\begin{aligned} P_j^{-1} - \sum_{m_1=1}^{M} \alpha^{m_1} (\mathcal{F}_{\sigma_j, m_1} + \mathcal{G}_{m_1} \Delta \mathcal{H}_{\sigma_j})^\top \\ P_{j+1}^{-1} \sum_{m_2=1}^{M} \alpha^{m_2} (\mathcal{F}_{\sigma_j, m_2} + \mathcal{G}_{m_2} \Delta \mathcal{H}_{\sigma_j}) \succ 0 \end{aligned} \tag{3.33}$$

for all $j \in \{1, ..., \tilde{N}\}$, $\Delta \in \mathbf{\Delta}$ and $\alpha \in \mathbf{\Omega}$ where $P_{\tilde{N}+1} = P_1$ as this would guarantee the existence of an $\epsilon > 0$ such that $\Delta V_k(\bar{x}_k) := V_{k+1}(\bar{x}_{k+1}) - V_k(\bar{x}_k) \leq -\epsilon \|\bar{x}_k\|^2$ for all $\bar{x}_k \in \mathbb{R}^n$ and all $k \in \mathbb{N}$.

Now we will prove that satisfaction of (3.33) for all $\Delta \in \mathbf{\Delta}$ and $\alpha \in \mathbf{\Omega}$ is implied by satisfaction of (3.30) and (3.31) with $\bar{K}_l = Z_{1,l}X_{1,l}^{-1}$, $l \in \mathbf{L}_u$ and $\bar{L}_l = Z_{2,l}X_{2,l}^{-1}$, $l \in \mathbf{L}_y$. By a Schur complement we can observe that the condition in (3.33) is equivalent to satisfying $\sum_{m=1}^{M} \alpha^m Q_{j,m} \succ 0$, where

$$Q_{j,m} := \begin{bmatrix} P_j^{-1} & (\mathcal{F}_{\sigma_j,m} + \mathcal{G}_m \Delta \mathcal{H}_{\sigma_j})^\top \\ (\mathcal{F}_{\sigma_j,m} + \mathcal{G}_m \Delta \mathcal{H}_{\sigma_j}) & P_{j+1} \end{bmatrix},$$

$\alpha \in \mathbf{\Omega}$ and $\Delta \in \mathbf{\Delta}$ for all $j = \{1, ..., \tilde{N}\}$. A necessary and sufficient condition for the satisfaction of $\sum_{m=1}^{M} \alpha^m Q_{j,m} \succ 0$ for all $\alpha \in \mathbf{\Omega}$ and for all $\Delta \in \mathbf{\Delta}$ is to require that $Q_{j,m} \succ 0$ for all $\Delta \in \mathbf{\Delta}$ and for all $j \in \{1, ..., \tilde{N}\},\ m \in \{1, ..., M\}$. Now observe that for all $\Delta_j \in \mathbf{\Delta}$, it holds that $\mathcal{H}_{\sigma_j}^\top (R_{j,m}^{-1} - \Delta^\top R_{j,m}^{-1} \Delta)\mathcal{H}_{\sigma_j} \succeq 0$, for all $R_{j,m}^{-1} \in \mathbf{R}$, $j \in \{1, \ldots, \tilde{N}\}$ and $m \in \{1, \ldots, M\}$ by the definitions of $\mathbf{\Delta}$ in (3.15) and $\mathbf{R}$ in (3.29). Hence, $Q_{j,m} \succ 0$ is satisfied if

$$\begin{bmatrix} P_j^{-1} & (\mathcal{F}_{\sigma_j,m} + \mathcal{G}_m \Delta \mathcal{H}_{\sigma_j})^\top \\ (\mathcal{F}_{\sigma_j,m} + \mathcal{G}_m \Delta \mathcal{H}_{\sigma_j}) & P_{j+1} \end{bmatrix} \succ \begin{bmatrix} \mathcal{H}_{\sigma_j}^\top (R_{j,m}^{-1} - \Delta^\top R_{j,m}^{-1} \Delta)\mathcal{H}_{\sigma_j} & 0 \\ 0 & 0 \end{bmatrix},$$

or equivalently that $S_{j,m}^\top \bar{Q}_{j,m} S_{j,m} \succ 0$, where

$$\bar{Q}_{j,m} := \begin{bmatrix} G_{\sigma_j}^\top P_j^{-1} G_{\sigma_j} & (\mathcal{F}_{\sigma_j,m} G_{\sigma_j})^\top & 0 & (\mathcal{H}_{\sigma_j} G_{\sigma_j})^\top \\ \star & P_{j+1} & \mathcal{G}_m R_{j,m} & 0 \\ \star & \star & R_{j,m} & 0 \\ \star & \star & \star & R_{j,m} \end{bmatrix},$$

and

$$S_{j,m} := \begin{bmatrix} G_{\sigma_j}^{-1} & 0 \\ 0 & I \\ R_{j,m}^{-1} \Delta \mathcal{H}_{\sigma_j} & 0 \\ -R_{j,m}^{-1} \mathcal{H}_{\sigma_j} & 0 \end{bmatrix}.$$

The matrix inequality $S_{j,m}^\top \bar{Q}_{j,m} S_{j,m} \succ 0$ is satisfied if $\bar{Q}_{j,m} \succ 0$ since $S_{j,m}$ is full column-rank. Using the fact that it holds that $G_{\sigma_j}^\top P_j^{-1} G_{\sigma_j} \succeq G_{\sigma_j} + G_{\sigma_j}^\top - P_j$, the satisfaction of $\bar{Q}_{j,m} \succ 0$ is implied by the satisfaction of

$$\left[\begin{array}{cc|cc} G_{\sigma_j} + G_{\sigma_j}^\top - P_j & (\mathcal{F}_{\sigma_j,m} G_{\sigma_j})^\top & 0 & (\mathcal{H}_{\sigma_j} G_{\sigma_j})^\top \\ \star & P_{j+1} & \mathcal{G}_m R_{j,m} & 0 \\ \hline \star & \star & R_{j,m} & 0 \\ \star & \star & \star & R_{j,m} \end{array}\right] \succ 0. \tag{3.34}$$

Note that $G_{\sigma_j}^\top P_j^{-1} G_{\sigma_j} \succeq G_{\sigma_j} + G_{\sigma_j}^\top - P_j$ follows from the fact that if $P_j^{-1} \succ 0$ then $(G_{\sigma_j} - P_j)^\top P_j^{-1} (G_{\sigma_j} - P_j) \succeq 0$.

Finally, combining $Z_{2,\sigma_j}\mathcal{C}_{\sigma_j} = \bar{L}_{\sigma_j}\mathcal{C}_{\sigma_j}G_{\sigma_j}$, which is derived from (3.31b) and $\bar{L}_{\sigma_j} = Z_{2,\sigma_j}X_{2,\sigma_j}^{-1}$, and $Z_{1,\sigma_j}\mathcal{E}_{\sigma_j} = \bar{K}_{\sigma_j}\mathcal{E}_{\sigma_j}G_{\sigma_j}$, which is derived from (3.31a) and $\bar{K}_{\sigma_j} = Z_{1,\sigma_j}X_{1,\sigma_j}^{-1}$, with (3.27), we can substitute

$$\begin{aligned}\mathcal{F}_{\sigma_j,m}G_{\sigma_j} &= \mathcal{A}_{\sigma_j,m}G_{\sigma_j} + \mathcal{B}_{\sigma_j,m}\bar{K}_{\sigma_j}\mathcal{E}_{\sigma_j}G_{\sigma_j} - \mathcal{D}_{\sigma_j}\bar{L}_{\sigma_j}\mathcal{C}_{\sigma_j}G_{\sigma_j}\\ &= \mathcal{A}_{\sigma_j,m}G_{\sigma_j} + \mathcal{B}_{\sigma_j,m}Z_{1,\sigma_j}\mathcal{E}_{\sigma_j} - \mathcal{D}_{\sigma_j}Z_{2,\sigma_j}\mathcal{C}_{\sigma_j},\\ \mathcal{H}_{\sigma_j}G_{\sigma_j} &= \mathcal{I}_{\sigma_j}G_{\sigma_j} + \mathcal{J}_{\sigma_j}\bar{K}_{\sigma_j}\mathcal{E}_{\sigma_j}G_{\sigma_j}\\ &= \mathcal{I}_{\sigma_j}G_{\sigma_j} + \mathcal{J}_{\sigma_j}Z_{1,\sigma_j}\mathcal{E}_{\sigma_j},\end{aligned}$$

into (3.34), which yields (3.30). The above substitution of $Z_{2,l}\mathcal{C}_l = \bar{L}_l\mathcal{C}_lG_l$, thus using (3.31b), is only needed when $l \in \mathbf{L}_y$ since, by definition, $\Upsilon_l^y = 0$ when $l \notin \mathbf{L}_y$ (and thus $\mathcal{C}_l = 0$). Similarly, substitution of $Z_{1,l}\mathcal{E}_l = \bar{K}_l\mathcal{E}_lG_l$, thus using (3.31a), is only needed when $l \in \mathbf{L}_u$ since $\Upsilon_l^u = 0$ when $l \notin \mathbf{L}_u$ (and thus $\mathcal{J}_l = 0$ and $\mathcal{B}_{l,m} = 0$ for all $m \in \{1, ..., M\}$).

We have shown that satisfaction of (3.30) and (3.31) yield $\bar{K}_l$ and $\bar{L}_l$ which satisfy (3.33) for all $j \in \{1, ..., \tilde{N}\}$, $\alpha \in \mathbf{\Omega}$ and $\Delta \in \mathbf{\Delta}$. Hence, using standard Lyapunov arguments, UGES of (3.13), (3.21), (3.27) with $\alpha_k \in \mathbf{\Omega}$, $\Delta_k \in \mathbf{\Delta}$ and the given protocol satisfying (3.6) is guaranteed and yields also UGES of (3.11), (3.12) with $h_k \in [h_{min}, h_{max}]$ and the same periodic protocol. □

Remark 3.5. The requirement that any node $l \in \{1, ..., \bar{N}\}$ containing at least one sensor, consists of linearly independent sensors, i.e. $(C)_{\bar{\mathbf{J}}_l^y,\bullet}$ has full row rank, is a rather mild condition. Indeed, the natural situation of C having full row rank is a sufficient condition for this requirement. ◁

3.4.2 Parallel Subsystem Communication

In this section, we will generalize the reasoning in Section 3.4.1 to allow the subsystems to communicate in parallel. Specifically, we will explain why Assumption 3.1 is included and how it is possible to remove Assumption 3.1 from Theorem 3.2. If Assumption 3.1 does not hold, sensors and/or actuators from two (or more) subsystems are grouped into one node and, hence, communicate at the same transmission instant. The consequence of two subsystems communicating is that $\bar{K}_l$ and $\bar{L}_l$ as defined in (3.25) will remain structured, as these gains will then contain elements that must be equal to zero. Due to these design variables containing structure, solutions to Theorem 3.2 using (3.26) will not be valid. In order to remove Assumption 3.1, (3.26) needs to be generalized to

$$\Gamma_l^u K_l = \sum_{i=1}^{N} \Upsilon_{l,i}^u \bar{K}_{l,i} \Upsilon_{l,i}^{x\top}, \quad L_l\Gamma_l^y = \sum_{i=1}^{N} \Upsilon_{l,i}^x \bar{L}_{l,i} \Upsilon_{l,i}^{y\top}, \tag{3.35}$$

where (3.23) is then generalized to

$$\Upsilon^u_{l,i} := \begin{cases} (\Gamma^u_l)_{\bullet,\bar{\mathbf{J}}^u_l \cap \mathbf{J}^u_i}, & \text{if } \ l \in \mathbf{L}_{u,i}, \\ 0, & \text{otherwise,} \end{cases}$$
$$\Upsilon^y_{l,i} := \begin{cases} (\Gamma^y_l)_{\bullet,\bar{\mathbf{J}}^y_l \cap \mathbf{J}^y_i}, & \text{if } \ l \in \mathbf{L}_{y,i}, \\ 0, & \text{otherwise,} \end{cases}$$
$$\Upsilon^x_{l,i} := \begin{cases} (I^x)_{\bullet,\bar{\mathbf{J}}^x_l \cap \mathbf{J}^x_i}, & \text{if } \ \bar{\mathbf{J}}^x_l \cap \mathbf{J}^x_i \neq \emptyset, \\ 0, & \text{otherwise,} \end{cases}$$

where (3.24) is generalized to

$$\mathbf{L}_{u,i} := \{l \in \{1, ..., \bar{N}\} \mid \bar{\mathbf{J}}^u_l \cap \mathbf{J}^u_i \neq \emptyset\}, \tag{3.36a}$$
$$\mathbf{L}_{y,i} := \{l \in \{1, ..., \bar{N}\} \mid \bar{\mathbf{J}}^y_l \cap \mathbf{J}^y_i \neq \emptyset\}, \tag{3.36b}$$

and, finally, (3.25) is generalized to

$$\bar{K}_{l,i} := \begin{cases} (K_l)_{\bar{\mathbf{J}}^u_l \cap \mathbf{J}^u_i, \bar{\mathbf{J}}^x_l \cap \mathbf{J}^x_i}, & \text{if } \ l \in \mathbf{L}_{u,i}, \\ 0, & \text{otherwise,} \end{cases} \tag{3.37a}$$
$$\bar{L}_{l,i} := \begin{cases} (L_l)_{\bar{\mathbf{J}}^x_l \cap \mathbf{J}^x_i, \bar{\mathbf{J}}^y_l \cap \mathbf{J}^y_i}, & \text{if } l \in \mathbf{L}_{y,i}, \\ 0, & \text{otherwise.} \end{cases} \tag{3.37b}$$

Notice that, using (3.35), if multiple subsystems communicate simultaneously then each non-zero gain $\bar{K}_{l,i}$ and $\bar{L}_{l,i}$ is non-structured. In the case that Assumption 3.1 is adopted, for each $l \in \{1, ..., \bar{N}\}$, there exists only one $i \in \{1, ..., N\}$ where $\Upsilon^x_{l,i} \neq 0$, and thus, (3.35) simplifies to (3.26). With (3.35), we have that (3.27) becomes

$$\mathcal{F}_{l,m} = \mathcal{A}_{l,m} + \sum_{i=1}^{N} (\mathcal{B}_{l,m,i} \bar{K}_{l,i} \mathcal{E}_{l,i} - \mathcal{D}_{l,i} \bar{L}_{l,i} \mathcal{C}_{l,i}), \tag{3.38a}$$

$$\mathcal{H}_l = \mathcal{I}_l + \sum_{i=1}^{N} \mathcal{J}_{l,i} \bar{K}_{l,i} \mathcal{E}_{l,i}, \tag{3.38b}$$

where $\mathcal{B}_{l,m,i}, \mathcal{E}_{l,i}, \mathcal{D}_{l,i}$, $\mathcal{C}_{l,i}$ and $\mathcal{J}_{l,i}$ are of the form $\mathcal{B}_{l,m}, \mathcal{E}_l, \mathcal{D}_l$, $\mathcal{C}_l$ and $\mathcal{J}_l$ in (3.28) with $\Upsilon^u_{l,i}, \Upsilon^y_{l,i}$ and $\Upsilon^x_{l,i}$ substituted for $\Upsilon^u_l, \Upsilon^y_l$ and Υ^x_l, respectively. These extensions lead to the following theorem, which is a generalization of Theorem 3.2.

Theorem 3.3. *Consider the system* (3.11), (3.12) *with* $h_k \in [h_{min}, h_{max}]$, $k \in \mathbb{N}$, *and its overapproximation given by* (3.13), (3.21), (3.38). *Assume that the protocol satisfies* (3.6) *and any node* $l \in \{1, ..., \bar{N}\}$ *containing at least one sensor from subsystem* i, *i.e.* $\bar{\mathbf{J}}^y_l \cap \mathbf{J}^y_i \neq \emptyset$, *consists of linearly independent subsystem sensors, i.e.* $(C)_{\bar{\mathbf{J}}^y_l \cap \mathbf{J}^y_i, \bullet}$ *has full row rank. Suppose there exist symmetric*

matrices P_j*, matrices* $R_{j,m} \in \mathbf{R}$*, with* $\mathbf{R}$ *as in* (3.29)*, and matrices* G_l*,* $Z_{1,l,i}$*,* $Z_{2,l,i}$*,* $X_{1,l,i}$ *and* $X_{2,l,i}$ *where* $i \in \{1,...,N\}$*,* $j \in \{1,...,\tilde{N}\}$*,* $m \in \{1,...,M\}$*,* $l \in \{1,...,\bar{N}\}$ *such that* (3.30) *holds for* $j \in \{1,...,\tilde{N}\}$*,* $m \in \{1,...,M\}$*, and*

$$X_{1,l,i}\mathcal{E}_{l,i} = \mathcal{E}_{l,i}G_l, \quad \text{for } l \in \mathbf{L}_{u,i},\ i \in \{1,...,N\}$$
$$X_{2,l,i}\mathcal{C}_{l,i} = \mathcal{C}_{l,i}G_l, \quad \text{for } l \in \mathbf{L}_{y,i},\ i \in \{1,...,N\}$$

for which we define

$$\Xi_1(j,m) := \mathcal{A}_{\sigma_j,m}G_{\sigma_j} + \sum_{i=1}^{N}(\mathcal{B}_{\sigma_j,m,i}Z_{1,\sigma_j,i}\mathcal{E}_{\sigma_j,i} - \mathcal{D}_{\sigma_j,i}Z_{2,\sigma_j,i}\mathcal{C}_{\sigma_j,i}),$$
$$\Xi_2(j) := \mathcal{I}_{\sigma_j}G_{\sigma_j} + \sum_{i=1}^{N}\mathcal{J}_{\sigma_j,i}Z_{1,\sigma_j,i}\mathcal{E}_{\sigma_j,i},$$

for $j \in \{1,...,\tilde{N}\}$*,* $m \in \{1,...,M\}$*, with* $P_{\tilde{N}+1} := P_1$ *and the sets* $\mathbf{L}_{u,i}$ *and* $\mathbf{L}_{y,i}$*,* $i \in \{1,...,N\}$*, are defined in* (3.36)*, respectively. Then the controller gains* K_l*, defined by* (3.35)*,* (3.37) *and* $\bar{K}_{l,i} = Z_{1,l,i}X_{1,l,i}^{-1}$*,* $l \in \mathbf{L}_{u,i}$*,* $i \in \{1,...,N\}$*, and the observer gains, defined by* (3.35)*,* (3.37) *and* $\bar{L}_{l,i} = Z_{2,l,i}X_{2,l,i}^{-1}$*,* $l \in \mathbf{L}_{y,i}$*,* $i \in \{1,...,N\}$*, render the system* (3.11)*, with* $h_k \in [h_{min}, h_{max}]$*,* $k \in \mathbb{N}$*, and the mentioned periodic protocol, UGES.*

Proof. The proof follows directly from Theorem 3.2. □

Remark 3.6. The NCS model presented here can be extended to include time-varying communication delays $\tau_k \in [\tau_{max}, \tau_{max}]$, where $\tau_k < h_k$ for all $k \in \mathbb{N}$, using the results in [38], in a straightforward manner. Such an extension only requires redefining $\bar{B}_{h_k}$ to be $\bar{B}_{h_k,\tau_k} = \int_{\tau_k}^{h_k} e^{A(h_k-s)}dsB$ and adding an additional term $W_{h_k,\tau_k}\hat{u}_{k-1} = \int_0^{\tau_k} e^{A(h_k-s)}dsB\hat{u}_{k-1}$ to x_{k+1} in (3.4). As a direct consequence, the closed-loop system matrix (3.12) will depend on τ_k. This delay-induced uncertainty can be incorporated into an overapproximated system of the form (3.13), where the additive uncertainty set $\mathbf{\Delta}$ then becomes part of $\mathbb{R}^{3n_x \times 3n_x}$ instead of $\mathbb{R}^{2n_x \times 2n_x}$. The decomposition of this overapproximated system into the form (3.13) with (3.27) can still be achieved and, hence, Theorem 3.3 can still be applied. ◁

3.5 Example

In this section, we illustrate the presented theory using a well-known benchmark example [17, 34, 38, 58, 94, 122] in the NCS literature consisting of a linearized model of an unstable batch reactor. This benchmark example has been used primarily to compare conservatism in stability analysis techniques, where the

dynamic output-based stabilizing controller is assumed to be given. In [34], dynamic output-feedback controllers were synthesized for this problem in the presence of a shared communication medium, but with a constant transmission interval. These are the first results which synthesizes dynamic output-based stabilizing controllers for this problem with both a shared communication medium and time-varying transmission intervals. Moreover, we synthesize controllers while imposing constraints regarding controller decentralization. First we will synthesize stabilizing decentralized controllers for a single batch reactor in Section 3.5.1 and then, in Section 3.5.2, synthesize stabilizing controllers for multiple batch reactors. The single batch reactor will be considered primarily for reasons of comparison to previous work, and the multiple batch reactor case will be considered to explore the (computational) limitations of the presented synthesis technique.

3.5.1 Single Batch Reactor

The system matrices for the linearized batch reactor are given in [122]. This system is not in an ideal form to be expressed as a collection of disjoint subsystems as in (3.1). So we use a linear state transformation $z = \bar{T}x$, where

$$\bar{T} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

and we reverse the order of the output vector y to arrive at the following system matrices for the system in the form (3.2):

$$\left[\begin{array}{c|c} A & B \\ \hline C & \end{array}\right] = \left[\begin{array}{cc:cc|c} -4.290 & 0.675 & -0.581 & -0.581 & 5.679 & 0 \\ 4.273 & -0.761 & 0.048 & -1.295 & 1.136 & 0 \\ -0.208 & 1.039 & 2.399 & 3.681 & 0 & -3.146 \\ 0 & 0 & -1.019 & -9.016 & 0 & 3.146 \\ \hline 1 & 0 & 0 & 0 & & \\ 0 & 0 & 1 & 0 & & \end{array}\right]. \tag{3.39}$$

The two disjoint subsystems of (3.39) are denoted by the dashed lines. We will use the system matrices in (3.39) as the plant model (3.2) for the remainder of this example. We will compare four different controller structures, denoted C1, C2, C3 and C4.

C1 - The first controller (C1) is a centralized controller ($N = 1$) of the form (3.9) where the communication medium is not shared, meaning all sensors and actuators are in one node ($\bar{N} = 1$) and $\Gamma_1^u = \Gamma_1^y = I$. This is the simplest setting for which Theorem 3.2 applies.

C2 - The second controller (C2) is a decentralized controller ($N = 2$) of the form (3.9) where the decentralized structure is indicated in (3.39) by the dashed lines. The communication medium is not shared, meaning all sensors and

actuators are in one node ($\bar{N}=1$) and $\Gamma_1^u=\Gamma_1^y=I$. Since the communication medium is not shared and the controller is decentralized, the subsystems will communicate in parallel and Theorem 3.3 must be used.

C3 - The third controller (C3) is a decentralized controller ($N=2$) of the form (3.9) where the decentralized structure is indicated in (3.39) with the dashed lines. In addition, the communication medium is shared. We specify that each sensor and actuator is placed into a separate node. Hence, there are $\bar{N}=4$ nodes, where $\Gamma_1^u=\mathrm{diag}(1,0)$, $\Gamma_2^u=\mathrm{diag}(0,1)$, $\Gamma_3^u=\Gamma_4^u=\Gamma_1^y=\Gamma_2^y=\mathrm{diag}(0,0)$, $\Gamma_3^y=\mathrm{diag}(1,0)$ and $\Gamma_4^y=\mathrm{diag}(0,1)$. We specify the protocol to be the well-known Round Robin protocol given by (3.6) with $\sigma_l=l$, $l\in\{1,...,4\}$ and $\tilde{N}=4$. With this decentralized structure and communication protocol, the subsystems communicate in a serial fashion and Theorem 3.2 applies.

To help clarify (3.26) we now explicitly provide the matrices Υ_l^u, Υ_l^y, Υ_l^x, $\bar{K}_l$ and $\bar{L}_l$, $l\in\{1,...,4\}$ associated with this controller. If we define the elements of K_l and L_l as

$$K_l:=\left[\begin{array}{c|c}\begin{matrix}K_{l,1}\\0\end{matrix} & \begin{matrix}K_{l,2}\\0\end{matrix}\\ \hline \begin{matrix}0\\K_{l,3}\end{matrix} & \begin{matrix}0\\K_{l,4}\end{matrix}\end{array}\right],$$

$$L_l^\top:=\left[\begin{array}{c|c}\begin{matrix}L_{l,1}\\0\end{matrix} & \begin{matrix}L_{l,2}\\0\end{matrix}\\ \hline \begin{matrix}0\\L_{l,3}\end{matrix} & \begin{matrix}0\\L_{l,4}\end{matrix}\end{array}\right],$$

then we have that (3.26) translates to

$$\Gamma_1^uK_1=\Upsilon_1^u\bar{K}_1\Upsilon_1^{x\top}=\begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}K_{1,1} & K_{1,2}\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&0\end{bmatrix},$$

$$\Gamma_2^uK_2=\Upsilon_2^u\bar{K}_2\Upsilon_2^{x\top}=\begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}K_{2,3} & K_{2,4}\end{bmatrix}\begin{bmatrix}0&0&1&0\\0&0&0&1\end{bmatrix},$$

$$\Gamma_3^uK_3=\Gamma_4^uK_4=0,\ L_1\Gamma_1^y=L_2\Gamma_2^y=0,$$

$$L_3\Gamma_3^y=\Upsilon_3^x\bar{L}_3\Upsilon_3^{y\top}=\begin{bmatrix}1&0\\0&1\\0&0\\0&0\end{bmatrix}\begin{bmatrix}L_{3,1}\\L_{3,2}\end{bmatrix}\begin{bmatrix}1&0\end{bmatrix},$$

$$L_4\Gamma_4^y=\Upsilon_4^x\bar{L}_4\Upsilon_4^{y\top}=\begin{bmatrix}0&0\\0&0\\1&0\\0&1\end{bmatrix}\begin{bmatrix}L_{4,3}\\L_{4,4}\end{bmatrix}\begin{bmatrix}0&1\end{bmatrix}.$$

C4 - The fourth controller (C4) is an exact discretization of the dynamic controller considered in [17, 38, 58, 94, 122] which, when discretized, becomes of the form $\tilde{x}_{k+1}=A_c\tilde{x}_k+B_c\hat{y}_k$, $u_k=C_c\tilde{x}_k+D_c\hat{y}_{k-1}$, where

$$\left[\begin{array}{c|c}A_c & B_c\\ \hline C_c & D_c\end{array}\right]=\left[\begin{array}{cc|cc}1&0&h_\star&0\\0&1&0&h_\star\\ \hline -2&0&-2&0\\0&8&0&5\end{array}\right],$$

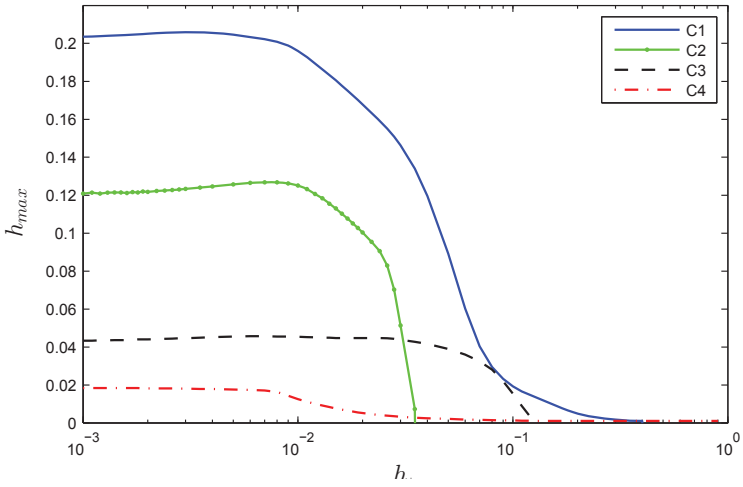


Figure 3.2: Maximum h_{max} (with $h_{min} = 10^{-3}$) for which (i) stabilizing controller gains for the batch reactor system could be synthesized for C1 and C3 using Theorem 3.2 and C2 using Theorem 3.3 and (ii) stability could be guaranteed for C4.

$h_\star$ is the nominal sampling interval used for controller discretization and the decentralized structure is indicated with dashed lines. This discrete-time controller was also studied in [34]. We consider the communication medium to be shared and impose the same nodes and Round Robin protocol as specified for controller C3.

For each of the controllers C1, C2 and C3 we took different values of $h_\star$ and used the YALMIP interface [78] with the SeDuMi solver [118] to verify the conditions of Theorem 3.2 or Theorem 3.3 in order to find stabilizing gains K_l and L_l which maximize h_{max} such that the NCS (3.11) is stable for $[h_{min}, h_{max}] = [10^{-3}, h_{max}]$, $k \in \mathbb{N}$, i.e. for a fixed lower bound on the transmission interval. In the NCS literature, this problem setting is also known as finding the maximum allowable transmission interval (MATI) that still guarantees stability [17, 38, 58, 94, 122]. Unlike the aforementioned references which consider the controller as given, we now have the advantage of using controller synthesis to push the MATI to an even higher value. For C4, the closed-loop model and stability analysis technique given in [38] (see Remark 3.2) were used to verify stability in order to maximize the uncertainty range $[h_{min}, h_{max}] = [10^{-3}, h_{max}]$, $k \in \mathbb{N}$. For C1, C2, C3 and C4 we considered an overapproximation of the closed-loop dynamics using 10 gridpoints.

The result of applying Theorem 3.2 to C1 and C3 and applying Theorem 3.3 to C2 is plotted in Fig. 3.2. The regions for which closed-loop stability can

be guaranteed for controller structures C1, C2, C3 and C4 lie below the lines corresponding to C1, C2, C3 and C4, respectively. Furthermore, the regions lying below the lines corresponding to C1, C2 and C3 represent the set of stabilizing controllers that can be found by iteratively applying Theorem 3.2 or Theorem 3.3. One can see that, as expected, the case of a centralized controller and a communication medium which is not shared (C1) achieves the largest robustness margins and yields the largest set of stabilizing controllers. Imposing decentralized structural constraints (C2) and both decentralized structural constraints and a shared communication medium (C3) results in lower robustness margins and smaller sets of controllers. Lastly, analyzing stability of the 'conventional' batch reactor controller (C4) yields the smallest region. The stability analysis technique used to analyze C4 was shown in [38] to greatly reduce conservatism compared to robustness margins proven in previous work. However, every point $(h_\star, h_{max})$, which lies between the lines corresponding to C3 and C4 in Fig. 3.2, represents a decentralized observer-based controller (3.9) that has improved closed-loop robustness compared to the existing controller C4. Hence, the presented technique, which synthesizes decentralized dynamic controllers for C3, results in finding an entire set of controllers that have significantly improved robustness margins compared to the given decentralized controller C4.

As a final remark, the amount of time taken to solve the LMI feasibility problem given in Theorem 3.2 or Theorem 3.3 was, on average, 5 seconds for C1 and C2 and 20 seconds for C3 using a laptop containing a 2.5GHz Core2 Duo CPU and 3GB of RAM, which illustrates the computational feasibility of the presented approach for small-scale problems.

3.5.2 Multiple Batch Reactors

In this section, we will apply the synthesis techniques to the case where $\nu \in \mathbb{N}$ batch reactors are considered. Thus, the system matrices $\hat{A}$, $\hat{B}$ and $\hat{C}$ considered in this section are block diagonal, where the number of blocks is equal to ν and the blocks themselves are equal to the A, B and C matrices in (3.39), respectively. Hence, we have that $\hat{A} = \text{diag}(A, A, ..., A)$, $\hat{B} = \text{diag}(B, B, ..., B)$ and $\hat{C} = \text{diag}(C, C, ..., C)$. The scenario we aim to study is a factory setting where multiple batch reactors are using wireless communication to transmit their sensor values. We can perform a similar analysis as in the previous section, which compares the resulting robustness for different controller configurations. We will again consider an overapproximation of the closed-loop dynamics using $M = 10$ grid points, as in the previous section.

For this multi-batch-reactor situation, we will again consider the controller structures C1 and C2 introduced in the previous section. In this section, C2 is a decentralized controller ($N = 2\nu$) of the form (3.9) where the decentralized structure *for each batch reactor* is indicated in (3.39) with the dashed lines. Although synthesizing robustly stabilizing controllers C1 or C2 for a single batch

reactor *does* guarantee robust stability when applied to multiple batch reactors (due to a lack of network and subsystem coupling), the main reason for considering these two controller structures is to explore the computational limitations of the developed synthesis technique when the closed-loop dimension is increased. In addition to C1 and C2, we also want to investigate the resulting robustness for a decentralized controller structure that does introduce network coupling, denoted C5, described below.

C5 - The fifth controller (C5) is a decentralized controller ($N = 2\nu$) of the form (3.9) where the decentralized structure *for each batch reactor* is indicated in (3.39) with the dashed lines. In addition, the communication medium is shared. We specify that each sensor is placed into a separate node and all actuators are updated at each transmission instant. Hence, there are $\bar{N} = 2\nu$ nodes, where $\Gamma_l^u = I$ for all $l \in \{1, ..., 2\nu\}$ and $(\Gamma_l^y)_{(r,r)} = 1$ when $l = r$ and is zero otherwise for all $l \in \{1, ..., 2\nu\}$. We specify the protocol to be the well-known Round Robin protocol given by (3.6) with $\sigma_l = l$, $l \in \{1, ..., 2\nu\}$ and $\tilde{N} = 2\nu$. With this decentralized structure and communication protocol, the subsystems' actuators communicate in a parallel fashion and Theorem 3.3 applies.

The controller structure C5 models the practical situation where each (decentralized) controller is co-located at each actuator. Although the batch reactors themselves are not coupled, the presence of a shared communication network couples the individual batch reactor's dynamics. Unlike C1 and C2, a controller C5 that robustly stabilizes a single batch reactor is *not* guaranteed to be stabilizing when applied to multiple batch reactors. Hence, stabilizing multiple batch reactors by using this practical (wireless) controller structure requires a synthesis technique that includes both decentralized and shared networked aspects, such as the one provided in Theorem 3.3.

The result of applying the synthesis theorems to this setting is shown in Fig. 3.3. Due to the lack of a shared communication medium, the amount of robustness that can be guaranteed employing controllers corresponding to C1 and C2 is equal to that in Fig. 3.2 for any number of batch reactors. This is, of course, expected since the controller structures C1 and C2 do not couple the batch reactors in any way. However, for C5, which considers a shared communication medium, we can see that the amount of robustness that can be guaranteed decreases with an increasing number of batch reactors as more sensors are required to share the medium. For the case of $\nu = 3$ batch reactors, the proposed synthesis technique can be used to synthesize a decentralized controller that robustly stabilizes the closed-loop NCS for $h_{max} = 0.011$. Due to the memory limitations of the computer used for computation (mentioned before), stabilizing controllers could be synthesized for a maximum of $\nu = 4$ batch reactors for C1 and C2, whereas stabilizing controllers could be synthesized for a maximum number of $\nu = 3$ batch reactors for C5. This indicates the limitations of this technique when implemented in current commercially available computer hardware.

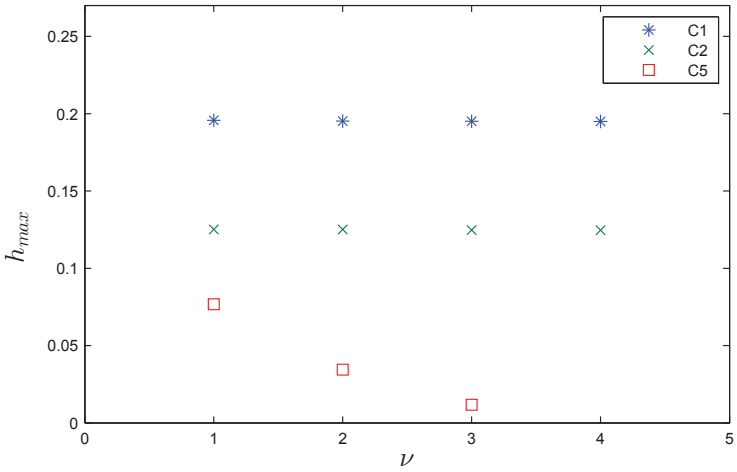


Figure 3.3: Maximum h_{max} (with $h_{min} = 10^{-3}$ and $h_\star = 0.010$) for which (i) stabilizing controller gains for the batch reactor system could be synthesized for C1 using Theorem 3.2 and C2 and C5 using Theorem 3.3.

To provide a better (solver/hardware independent) indication of how the computational complexity (i.e. memory/computational time required) scales with the state dimension, we will provide an analytical expression which specifies the number of free variables which must be determined to synthesize controller C5 as a function of the number of batch reactors (and grid points). This expression is

$$n_{vars} = n_P + n_R + n_G + n_X + n_Z \tag{3.40}$$

where n_P, n_R, n_G, n_X and n_Z indicate the number free variables in the P, R, G, X and Z matrices of Theorem 3.3, respectively. For the controller structure C5, it can be determined that

$$\begin{gathered} n_P = 144\nu^3 + 12\nu^2, \qquad n_R = 16M\nu^2, \qquad n_G = 288\nu^3, \\ n_X = 8\nu^2 + 4\nu, \qquad n_Z = 16\nu^2 + 2\nu, \end{gathered} \tag{3.41}$$

where $M \in \mathbb{N}$ is the number of grid points (and ν is the number of batch reactors). This representation illustrates the computational penalty incurred from each component, and gives insight into how the computational complexity would be reduced if certain elements were removed or modified. For example, performing robust stability analysis (as mentioned in Remark 3.2) only requires the computation of the P and R matrices and, therefore, $n_{vars} = n_P + n_R$.

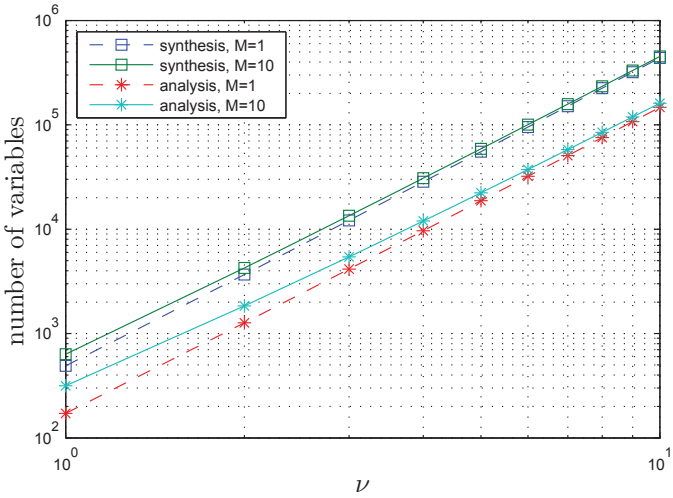


Figure 3.4: The number of variables which must be solved as a function of the number of batch reactors ν. Both the cases of controller synthesis and stability analysis are plotted with different number of grid points M.

Therefore, we can conclude that for this switched system framework, the computational complexity has polynomial growth (of third order in this case) in terms of the number of batch reactors considered. Interestingly, the decentralized synthesis technique presented in this paper has the same order of complexity as the simpler stability analysis problem.

In Fig. 3.4, the number of free variables is plotted as a function of the number of batch reactors, as specified in (3.40) with (3.41). This plot considers the number of variables required to both synthesize controllers C5 and determining robust stability analysis of a given controller C5 (based on [38]). We can observe how many more additional variables the synthesis technique presented in this paper requires than the stability analysis technique. Recalling that the maximum number of batch reactors able to synthesized for C5 was $\nu = 3$ (which corresponds to a closed-loop dimension of size 36), we can see that $\approx 10^4$ variables need to be determined by the LMI solver, which, from Fig. 3.4, implies that robust stability analysis of C5 can be assessed for at least $\nu = 4$ batch reactors (which corresponds to a closed-loop dimension of size 48). We also observe that the addition of more grid points does not introduce a large computational penalty in terms of additional variables as M enters linearly in (3.40), (3.41). Finally, Fig. 3.4 provides an indication of the amount of memory (and computational time) required to solve larger problems, and suggests where the limitations of one-shot LMI-based techniques currently are.

3.6 Conclusion

In this chapter, we have presented LMI-based synthesis conditions for designing decentralized observer-based control laws in the presence of a shared communication medium, which are robust with respect to time-varying transmission intervals and time-varying delays. This result was obtained by expressing the observer-based controller design problem as a multi-gain switched static output-feedback problem (with additive uncertainty), for which, the gains can be efficiently solved by LMI-based feasibility conditions. These LMI-based synthesis conditions, if satisfied, provide stabilizing gains for both the decentralized problem setting and the NCS problem setting in isolation, as well as the unification of these two problem settings. Using a benchmark example in the NCS literature, it was shown that this synthesis technique was able to find an entire set of controllers that significantly improved the closed-loop robustness compared to that of a dynamical controller, extensively studied in the literature. However, the computational complexity of the proposed approach limits this one-shot technique to synthesizing (decentralized) controllers for small and mid-size state-space dimensions. This limitation is primarily due to the fact that, although offering low levels of conservatism and efficient verification for small-scale problems, the number of variables that must be solved using a (switched) quadratic Lyapunov function candidate grows polynomially with respect to the state dimension. Therefore, this advocates that future techniques should not only focus on providing low levels of conservatism but also focus on having low levels of computational complexity. Accomplishing lower levels of computational complexity, improved solvers, and distributed solving of LMIs would enhance the possible application of the proposed methodology for large-scale systems. In any event, the framework laid down in this paper forms one of the first systematic methodologies for the synthesis of stabilizing controllers that incorporate both decentralized and (shared) networked features.

Chapter 4

Networked Control Systems Toolbox: Robust Stability Analysis Made Easy

"When you do things right, people won't be sure you've done anything at all."

- Futurama, "Godfellas"

4.1 Introduction

In recent years, there have been many theoretical developments in the area of stability analysis for networked control systems (NCSs), however, computational tools which implement the theory are lacking. For this reason, we have developed a prototype MATLAB toolbox with the primary goal of increasing the number of people that can interact with and use the stability theory for NCSs in a simple manner. In fact, there are multiple other reasons for creating an NCS toolbox: (i) to provide a user-friendly way to apply the existing theoretical developments to NCS problems of choice; (ii) to remove the burden of implementing some of the more complex algorithms employed in the theory; (iii) to gain feedback from the community to improve the quality and overall usefulness of both the toolbox and the theory. To appeal to the widest range of people, we have designed the toolbox such that it is useful for both the basic user (general control engineers) and the advanced user (members of the NCS community).

To appeal to the basic user, we have completely automated the stability analysis procedure such that the user is only required to input a plant model, a controller model, and bounds on the network uncertainties (described in Section 4.2) and can directly start analyzing robust stability of the corresponding closed-loop NCS. Moreover, we have recognized that NCS models can contain a considerable amount of variables, which can be intimidating to someone who

is not familiar with the NCS nomenclature. For example, in addition to the matrices required to define the plant and controller dynamics, variables characterizing all the network-induced effects listed in Section 4.2 must also be defined. Therefore, we created a graphical user interface (GUI), see Fig. 4.3, such that the variables are displayed and can be edited in a very intuitive manner. An additional stumbling block that is encountered when learning about NCS theory is related to the fact that there exist several mathematical techniques used for analysis. To provide easy access to the usage of the results in two mathematical frameworks (based on continuous-time hybrid models, as considered in Chapter 2, and discrete-time switched systems, as considered in Chapter 3) that have been developed for robust stability analysis, an additional analysis GUI was created, see Fig. 4.4.

To appeal to the advanced user, we enabled the use of (command-line) functionality outside the GUI to aid in analyzing more specific problem settings. First, researchers can add functionality for stability analysis to the toolbox by writing their own programs which can visualize entire stability regions for their problem of interest. Second, to encourage community participation in the future development of the toolbox, we made it possible for the researcher to incorporate their own discrete-time models (provided that the network-induced uncertainties enter the model in the appropriate way, described below). In the discrete-time framework, a polytopic overapproximation of the closed-loop NCS model is necessary before linear matrix inequalities can be used to determine if stability can be guaranteed. Acquiring a polytopic overapproximation (possibly with norm-bounded uncertainty) is the most tedious aspect of implementing a discrete-time approach to stability analysis of NCSs. The toolbox in this chapter automates this overapproximation procedure for a general class of models using a choice of different techniques and allows the NCS community to use the resulting overapproximation for analysis of customized stability or performance properties.

The MATLAB toolbox presented in this chapter has automated the theoretical developments in [29, 38, 58, 79]. In [29, 38, 79] a discrete-time framework was used and in [58] a continuous-time hybrid framework (built upon the work in [94]) was used to analyze stability properties of NCSs. In the discrete-time framework it has been shown that the amount of conservatism when analyzing stability can be minimized, however this framework is limited to the analysis of only linear plants and (switched) linear controllers. On the other hand, the hybrid formulation has the advantage of being able to analyze general nonlinear plants and controllers and study $\mathcal{L}_p$-gain type of performance criterion. So it is beneficial to consider both frameworks. However, for the sake of brevity, in this chapter, we focus on the toolbox implementation of the discrete-time framework even though the analysis tools in the hybrid framework have been implemented for linear plants and controllers as well. By using this toolbox, the user is able to make multi-disciplinary design tradeoffs between control properties such as stability and performance, and network-related properties such as

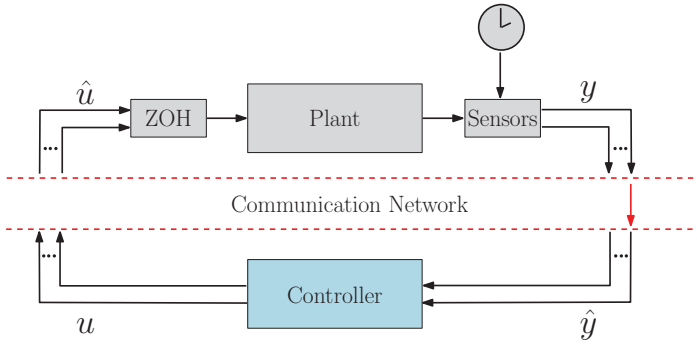


Figure 4.1: Block Schematic of a networked control system.

delays, scheduling, bandwidth limitations etc., in a user-friendly manner. Finally, we would like to emphasize here that there are many other results in the NCS literature which either use the discrete-time or hybrid framework that are currently not implemented in the the toolbox. We hope that the prototype toolbox presented here serves as a platform to which the NCS community can contribute.

The outline of this chapter is as follows. In Section 4.2, a description of the NCS model as implemented in the toolbox will be given; then in Section 4.3, the software structure will be presented. In Section 4.4, we describe how the basic user can quickly assess robust stability of an NCS model. In Section 4.5, we describe how the advanced user can plot stability regions and how custom models can be incorporated. In Section 4.6, we give an example of how the prototype toolbox can be used to investigate the conservatism introduced via one of the overapproximation techniques and demonstrate how stability regions can be plotted. Finally, conclusions and future developments are presented in Section 4.7.

4.2 Description of the NCS

The general schematic of an NCS is depicted in Fig. 4.1. It consists of a linear time-invariant (LTI) continuous-time plant and an LTI linear controller, which are interconnected through a communication network that induces

- uncertain time-varying transmission intervals h in the range $[h_{min}, h_{max}]$;
- uncertain time-varying network delays τ in the range $[\tau_{min}, \min\{h, \tau_{max}\}]$;

- uncertain dropouts sequences, where the number of successive dropouts is bounded by δ_{max};
- quantization;
- a shared communication medium, which prevents all sensor and actuator signals from being transmitted simultaneously.

Due to the communication medium being shared, the sensors and actuators are grouped into N nodes. At each transmission instant, one node obtains access to the network and transmits its corresponding values (implied by the red arrow shown in Fig. 4.1). Note that if there is only one node which contains all sensors and actuators, then the 'classical' (sampled-data) control setup, where all sensor and actuator signals are transmitted at each transmission time, is recovered. In case of multiple nodes, a mechanism is needed that orchestrates which node is given access, when the network is available. This mechanism is called a scheduling protocol. In this toolbox, we provide analysis techniques for two well-known protocols, namely, the Round Robin (RR) protocol and the Try-Once-Discard (TOD) protocol, see [122].

Incorporating the communication network (including the protocol) between the plant and controller, leads to the following operation aspects of the NCS in Fig. 4.1. First, the sensor acts in a time-driven fashion (i.e. sends data at each transmission instant) and both the controller and actuator act in an event-driven fashion (i.e. they respond instantaneously to newly arrived data). Second, the dropouts are modeled as prolongations of the sampling interval, meaning that, if a packet is considered 'dropped' then a new packet is transmitted at the next transmission time with new data. Third, the discrete-time control commands are converted to a continuous-time control signal by using a zero-order-hold (ZOH) function, see Fig. 4.1. Finally, the delays are assumed to be smaller than the transmission intervals. Note that retransmissions of packets can be modeled as prolongations of the delays. More information regarding the mathematical modeling and assumptions can be found in [29, 38, 58, 79].

4.3 Software Structure

The overall structure of the MATLAB software based on the discrete-time NCS framework is given in Fig. 4.2. What the toolbox currently offers is the possibility to analyze three different controller structures via three different overapproximation techniques and assess stability of two standard communication protocols. By carefully considering and implementing the software structure, horizontal expansion (e.g. including different controllers/models, different overapproximation techniques and different stability analysis conditions) is straightforward since the vertical links are defined in a generalized manner. Adopting this structure, we pave the road for the addition of custom models and create the possibility

for using the resulting overapproximations in customized conditions for analysis (e.g. analysis of different protocols). In this way, we hope to encourage the NCS community to use and contribute to the toolbox as well.

The structure of the toolbox reflects the goal of appealing to both the basic user and the advanced user. The basic user can simply interact with the first and last layer of the toolbox through the GUIs. However, for advanced users, the set of standard models provided with this toolbox might not include the exact model or stability analysis technique they are interested in studying. We provide the possibility for the advanced user to incorporate their own model and/or use their own stability or performance techniques, as will be discussed in Section 4.5.2.

4.4 Basic Functionality

The NCS Toolbox contains a number of features which make it easy to efficiently verify whether an LTI plant and an LTI controller interconnected with a network are robust to the aforementioned network imperfections. This is extremely convenient to someone who designed a linear controller and wants to quickly verify these robustness properties. We provide easy model management, overapproximation automation and automated stability verification.

4.4.1 Model Management

An NCS class object (which describes the model covered in Section 4.2) consists of an LTI plant, an LTI controller and network variables (bounds on time-varying sampling intervals, bounds on time-varying delays, a bound on the maximum number of successive dropouts, the type of quantizer, the node definitions and the protocol). Creating an NCS object is easily done by using the NCS Editor (GUI) shown in Fig. 4.3.

To display the NCS Editor, simply type `ncsEditor` in the MATLAB command line and the GUI will appear. Here the NCS properties can be defined and an NCS object can be exported to the MATLAB workspace by clicking 'File > Export'. Modifications to an NCS object can be made by clicking on 'File > Import'. We will now briefly discuss the specific elements of the NCS object.

Plant

In this toolbox the plant, as shown in Fig. 4.1, is an LTI continuous-time system expressed as

$$\begin{aligned} \dot{x}(t) &= Ax(t) + B\hat{u}(t), \\ y(t) &= Cx(t). \end{aligned}$$

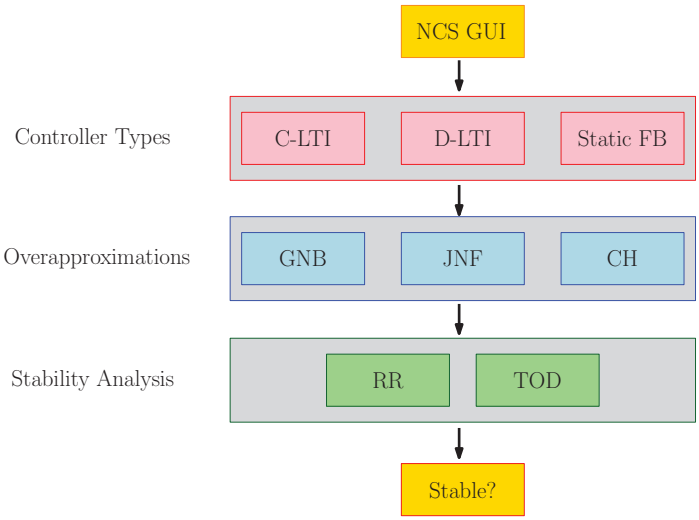


Figure 4.2: The software structure for the discrete-time NCS framework.

The matrices A, B and C are the first input parameters to define an NCS object. The signals $\hat{u}$ (and $\hat{y}$) result from transmitting y and u, respectively, through the network, as depicted in Fig. 4.1. In the related input boxes, see Fig. 4.3, either matrices can be input directly or variables defined in the MATLAB workspace can be used.

Controller

Next, the controller, as shown in Fig. 4.1, needs to be specified. The controller type can be defined by clicking on a choice from the drop down dialog box, see Fig. 4.3. There are three possible controller types currently available in this toolbox. The first is 'Static Feedback' where the feedback law is of the form $u(t) = K\hat{y}(t)$. The second controller type is a 'C-LTI Dynamic Feedback' control law, which is given by the following continuous-time linear dynamic control law:

$$\begin{aligned} \dot{x}^c(t) &= A^c x^c(t) + B^c \hat{y}(t), \\ u(t) &= C^c x^c(t) + D^c \hat{y}(t). \end{aligned}$$

The third controller type is a 'D-LTI Dynamic Feedback' control law, which is given by the following discrete-time linear dynamic control law:

$$\begin{aligned} x^c_{k+1} &= A^c x^c_k + B^c \hat{y}_k, \\ u(t_k) &= C^c x^c_k + D^c \hat{y}(t_k), \end{aligned}$$

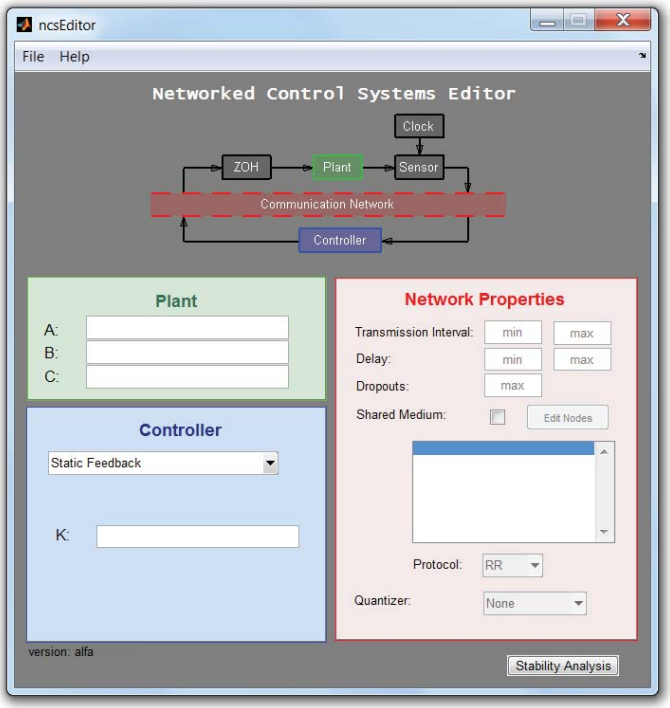


Figure 4.3: Graphical user interface to define an NCS object.

where t_k is the k^{th} transmission time. The matrices K or A_c, B_c, C_c and D_c are entered into the corresponding text boxes, see Fig. 4.3. In these text boxes, either matrices can be input directly or variables defined in the workspace can be used. Being able to input variables from the workspace is convenient if the controller was designed using a different tool (e.g. based on loop-shaping or H_∞-based tools).

Network

Lastly, the network effects need to be defined. The bounds on the time-varying transmission intervals and time-varying delays are entered into the corresponding input box, see Fig. 4.3. If the transmission intervals or delays are considered constant then the 'min' value can be set equal to the 'max' value. If delays are not considered, then both the 'min' value and the 'max' value can be set equal to zero. Next, a bound on successive packet dropouts needs to be specified. If dropouts do not occur, this value can be set to zero.

Next, there is a check box to indicate whether or not the communication medium is shared (restricting all actuators and sensors from communicating at

the same transmission time). Once this box is checked, the button 'Define Nodes' is enabled. Another GUI will appear where the nodes can be defined by 'adding' inputs and outputs to each node. Nodes can be added or removed and moved up and down to change the ordering. Next, in the NCS Editor, the protocol type needs to be chosen as 'RR' for Round Robin or 'TOD' for Try-Once-Discard.

Finally, a type of quantizer can be chosen from a drop down menu as 'none', 'uniform' or 'logarithmic'. For further information about quantizer modeling and analysis, see [79].

Once all the fields are filled in, all the NCS parameters can be exported to the workspace in a single NCS object and saved for later use.

4.4.2 Automated Overapproximation

The discrete-time closed-loop NCS models given in [29,38,79] can all be expressed in the general form

$$\bar{x}_{k+1} = \left(A_{\sigma_k} + B_{\sigma_k} \begin{bmatrix} \hat{A}_{h_k} & 0 & 0 \\ 0 & \hat{E}_{h_k} & 0 \\ 0 & 0 & \hat{E}_{h_k-\tau_k} \end{bmatrix} C_{\sigma_k} \right) \bar{x}_k + \left(E_{\sigma_k} + D_{\sigma_k} \begin{bmatrix} \hat{A}_{h_k} & 0 & 0 \\ 0 & \hat{E}_{h_k} & 0 \\ 0 & 0 & \hat{E}_{h_k-\tau_k} \end{bmatrix} J_{\sigma_k} \right) \omega_k \tag{4.1}$$

where $k \in \mathbb{N}$ is a counter related to the number of transmissions, $\sigma_k \in \{1, ..., N\}$ is the node which receives network access at transmission time t_k, $h_k \in [h_{min}, h_{max}]$ is the sampling interval at the k^{th} transmission time, $\tau_k \in [\tau_{min}, \tau_{max}]$ is the delay at the k^{th} transmission time, $\omega_k \in \mathbb{R}^{n_\omega}$ is a disturbance on the closed-loop system and $\hat{A}_\rho = e^{\tilde{A}\rho}$ and $\hat{E}_\rho = \int_0^\rho e^{\tilde{A}s} ds$ for some matrix $\tilde{A}$. Due to the exponential form in which the uncertainty parameters h_k and τ_k appear in (4.1), i.e. in the matrices $\hat{A}_{h_k} \in \mathbf{\Gamma}_1$, $\hat{E}_{h_k} \in \mathbf{\Gamma}_2$, and $\hat{E}_{h_k-\tau_k} \in \mathbf{\Gamma}_3$, where

$$\begin{aligned} \mathbf{\Gamma}_1 &:= \left\{ e^{\tilde{A}h} \mid h \in [h_{min}, h_{max}] \right\}, \\ \mathbf{\Gamma}_2 &:= \left\{ \textstyle\int_0^h e^{\tilde{A}s} ds \mid h \in [h_{min}, h_{max}] \right\}, \\ \mathbf{\Gamma}_3 &:= \left\{ \textstyle\int_0^{h-\tau} e^{\tilde{A}s} ds \mid h \in [h_{min}, h_{max}], \tau \in [\tau_{min}, \tau_{max}] \right\}, \end{aligned}$$

the discrete-time model (4.1) is not directly suitable to construct LMI conditions for stability verification. To make the model amendable for LMI-based stability analysis, we aim to overapproximate the set of matrices $\mathbf{\Gamma}_i$, $i = 1, 2, 3$, as

$$\mathbf{\Gamma}_i \subseteq \left\{ \sum_{l=1}^{L} \alpha_l \bar{F}_{i,l} + \bar{G}_i \Delta_i \bar{H}_i \mid \alpha = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_L \end{bmatrix} \in \mathcal{A},\ \Delta_i \in \mathbf{\Delta} \right\},$$

where $\bar{F}_{i,l},\ \bar{G}_i,\ \bar{H}_i,\ l = 1, ..., L,\ i = 1, 2, 3$, are suitably constructed matrices, see, e.g. [59], with L the number of vertices in the polytopic overapproximation. In addition, $\mathbf{\Delta}$ is a specific set of structured matrices (e.g. with a norm bound $||\Delta|| := \sqrt{\lambda_{max}(\Delta^\top \Delta)} \leq 1$) and

$$\mathcal{A} = \left\{ \alpha \in \mathbb{R}^L \mid \alpha_l \geq 0,\ l = 1, ..., L \text{ and} \sum_{l=1}^{L} \alpha_l = 1 \right\}.$$

In the toolbox, we provide techniques to overapproximate the matrix sets $\mathbf{\Gamma}_i$ leading to the transformation of the closed-loop system (4.1) into

$$\bar{x}_{k+1} = \left(\sum_{l=1}^{L} \alpha_k^l \bar{A}_{\sigma_k,l} + \bar{B}_{\sigma_k} \bar{\Delta}_k \bar{C}_{\sigma_k} \right) \bar{x}_k + \left(\sum_{l=1}^{L} \alpha_k^l \bar{E}_{\sigma_k,l} + \bar{D}_{\sigma_k} \bar{\Delta}_k \bar{J}_{\sigma_k} \right) \omega_k \tag{4.2}$$

for $l \in \{1, ..., L\}$ and $\sigma_k \in \{1, ..., N\}$ with $\alpha_k^l \in \mathcal{A}$ and $\bar{\Delta}_k \in \mathbf{\Delta}$, $k \in \mathbb{N}$. Three overapproximation techniques are automated in this toolbox: an approach based on gridding and norm bounding (GNB), see, e.g., [38], an approach based on the Jordan normal form (JNF), see, e.g., [29] and an approach based on the Cayley-Hamilton theorem (CH), see, e.g., [50]. For a theoretical comparison between these three methods, the reader is referred to [59].

Although each of these three techniques are mathematically interesting, a fairly strong familiarity with the notation is required in order to implement (and actually use) these techniques in software. So for the control engineer who would like to determine if their specific closed-loop system is robust to network-induced effects, it will cost him or her a significant amount of time and effort to understand and implement these techniques. The NCS toolbox allows the control engineer to quickly verify if his control setup possesses robustness properties without having to know the all the (mathematical) details about a polytopic overapproximation besides the basic idea and the fact that they may introduce some conservatism.

4.4.3 Automated Stability Verification

To analyze stability of the NCS, linear matrix inequality (LMI) conditions are verified on the polytopic overapproximation described in the previous section. The sufficient LMI conditions (derived in the supporting literature [38, 59, 79]) are verified[1] using the YALMIP interface, [78], with the SeDuMi solver, [118].

[1] The primal residuals of the matrices provided by the LMI solver (which are equivalent to the eigenvalues) are checked for strict positivity to determine if the solution is valid. If the LMI solver encounters numerical problems, only a warning is displayed. In this way, the user is aware of the numerical problems and can make a decision on whether to accept the solution or not.

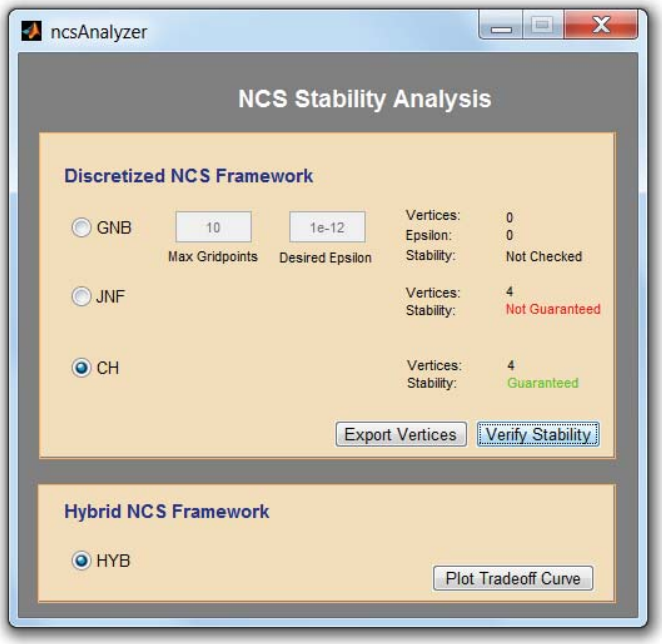


Figure 4.4: Graphical user interface to determine stability.

Stability of an NCS configuration can be assessed directly from the NCS Editor by clicking on the button 'Analyze Stability' in the lower left corner of the NCS Editor shown in Fig. 4.3 which opens the NCS Analyzer (GUI) shown in Fig. 4.4.

The NCS Analyzer was designed such that a user who is not familiar with the theory can quickly glance at the layout and intuitively understand that there are two main frameworks and the options available within each framework. In this way, a basic user can easily understand the overview of modeling and stability techniques of an NCS without having to first understand any of the mathematical details. Verifying closed-loop stability of the NCS can be done by clicking on the 'Verify Stability' button. After the stability verification is complete, the result is displayed on the rightmost column (see Fig. 4.4). The result includes information regarding the number of vertices in the overapproximation as well as whether the closed-loop NCS could be determined robustly stable. By making it easy to test stability with various options, the user can understand that some methods fail at proving stability while others succeed due to different levels of conservatism introduced by the different methods. Moreover, the user can directly compare the computational effort required by the different approaches when applied to different NCSs of their choice. This way, the basic user can experience first-

hand the capabilities (and limitations) of the robust stability analysis tools in the supporting literature [29, 38, 58, 79].

4.5 Advanced Functionality

Of course, having a GUI to interact with the NCS theory is useful to people who want to quickly verify stability properties of the provided NCS setups; however, it is also possible to plot entire regions for which robust stability can be guaranteed by iteratively calling the function to assess robust stability. By providing a simple function to assess stability, the advanced user can write their own programs to plot robust stability regions for different parameters of interest. In this section, we will provide MATLAB code which demonstrates how to plot stability regions when h_{max} and τ_{max} are the parameters of interest. Furthermore, we recognize that the NCS configurations included in the toolbox might not suit the needs of a particular NCS problem. This is why we made it possible for advanced users to incorporate their own models and use their own stability or performance conditions.

4.5.1 Plotting Robust Stability Regions

With the capability to assess stability of a single set of conditions, it is also possible to vary certain parameters to discover how robust the closed-loop system is for such parameter variations and to gain an understanding of where certain design tradeoffs lie. Assessing stability is achieved by executing the function

```
>> ncs.isNcsStable(ovrAprxType,quantVars,eu,M)
```

where `ovrAprxType` is either 'GNB', 'JNF' or 'CH' to indicate the type of overapproximation, `quantVars` is a structure containing data regarding quantization (if applicable), and `eu` and `M` are optional parameters used for the GNB algorithm. By iteratively using the above command, the NCS toolbox presented here can be used to produce tradeoff curves between networked properties (e.g. delays) and control properties (e.g. $\mathcal{L}_2$ gain) such as the ones given in previously published papers (e.g. in [38, 58, 59, 79]). In Section 4.6, we will provide simple example code which demonstrates how to plot stability regions.

4.5.2 Automated Overapproximation for Custom Models

To encourage community development, the NCS toolbox allows users to incorporate custom models for robust stability to be assessed using the discrete-time NCS framework. For example, one might be interested in an NCS setup where a different type of hold mechanism is used (besides the zero order hold) or where a multi-hop network needs to be modeled or where specific inputs and outputs

are directly wired to the controller and others are connected via a network. Creating a discrete-time model (of the form (4.1)) in these situations is not a very time-consuming task. However, the implementation of an overapproximation technique is generally a very time consuming task, which is why we have automated the overapproximation procedure for closed-loop models that can be expressed in the form (4.1). By providing the matrices A_σ, B_σ, C_σ, D_σ, E_σ, J_σ, $\sigma \in \{1, ..., N\}$ and $\tilde{A}$, the user can subsequently employ the automated procedure to create a polytopic model (4.2) for $l \in \{1, ..., L\}$ and $\sigma \in \{1, ..., N\}$ by using either the GNB, JNF or CH overapproximation technique. To create an overapproximation (4.2) of a model (4.1), the user just needs to execute the command

```
>>  ovrAprx = genPolyOvrAprx(mdlVars,ovrAprxType,eu,M)
```

where `mdlVars` is a variable containing the matrices needed to express a custom NCS model in the form (4.1), `ovrAprxType` is either 'GNB', 'JNF' or 'CH', and `eu` and `M` are optional parameters used for the GNB algorithm. The output of this function is a variable called `ovrAprx`, which contains the matrices in (4.2) and other information about the overapproximation technique. The data in `ovrAprx` is suitable to use for any type of performance or stability analysis conditions based on LMIs, such as those implemented in the toolbox which were specified in the supporting literature [29, 38, 79].

4.6 Example

In this example, we will analyze the batch reactor system, studied in [38, 94, 122] and others, to demonstrate how stability regions can be visualized, as described in Section 4.5.1. In particular, we will investigate the conservatism introduced by using the GNB technique. The GNB algorithm iteratively tightens the overapproximation by iteratively adding grid points to the overapproximated model at the location of the worst-case approximation error, until either the user-specified maximum number of grid points are reached or the user-specified desired tightness of the overapproximation is achieved, see [38]. The resulting tightness of the overapproximation that is obtained, denoted ϵ, is a norm related to the overapproximation error. This is the fundamental idea behind the GNB algorithm, and this is all the user needs to know before he/she can start investigating the technique through numerical examples.

In order to investigate the conservatism, we will use the following simple MATLAB script:

```
for taumax=0:0.005:0.03
    for hmax=0.01:0.01:0.07
        if hmax >= taumax
            ncs.tau = [0 taumax];
            ncs.h = [1e-3 hmax];
            stable =
                ncs.isNcsStable('GNB',[],eu,M);
            if stable==1
                plot(ncs.h(2),ncs.tau(2),'b.');
            else
                plot(ncs.h(2),ncs.tau(2),'rx');
            end
        end
    end
end
```

In this script, `ncs` is an NCS object containing all the parameters specified in [38], which was created using the NCS Editor shown in Fig. 4.3. This script tests combinations of maximal transmission intervals and delays, (`hmax`,`taumax`), by modifying the parameters `ncs.h` and `ncs.tau` and then calling the function `isNcsStable`. If the NCS is determined stable, then we specify to plot a blue dot, otherwise if stability cannot be guaranteed, we specify to plot a red 'x' (see, e.g. Fig. 4.5). The input parameters to `isNcsStable` are the `ncs` variable, the string `'GNB'` (to specify the overapproximation technique), `[]` to indicate that a quantizer is not used, and two additional parameters that need to be specified: the maximum number of grid points allowed, denoted `m`, and the desired approximation tightness, denoted `eu`. We will investigate the GNB technique by varying these parameters and observing the resulting conservatism introduced.

For the first case, denoted GNB1, we take `M`=50 and `eu`=2. For this example, the maximum of 50 grid points is never reached, so the overapproximation tightness $\epsilon \leq 2$ is guaranteed.

To compare with a second case, denoted GNB2, we set `M`=10 and keep `eu`=2. With these parameters, the overapproximation technique is limited to using a maximum of 10 grid points or less, depending on if $\epsilon \leq 2$ can be obtained within 10 grid points. This situation can be convenient if the user would prefer a reduction in computational time at the cost of a possible increase in conservatism. These results, along with the stability boundry plotted in [38], are shown in Fig. 4.5.

From Fig. 4.5(a), we can see that using GNB1, we have successfully approximated the results in [38] since all the blue dots lie to the left of the dashed line and all the red x's lie to the right of the dashed line. However, limiting the maximum number of grid points to 10 introduces conservatism, as for some points in Fig. 4.5(b) the approach (GNB2) is not able to prove stability, were these points were proven stable by the GNB1 approach. In any case, the stabil-

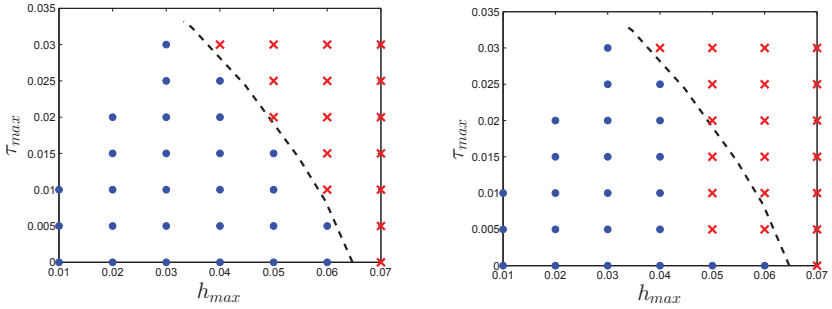


Figure 4.5: Batch reactor stability plot. (a) is associated with GNB1 and (b) is associated with GNB2.

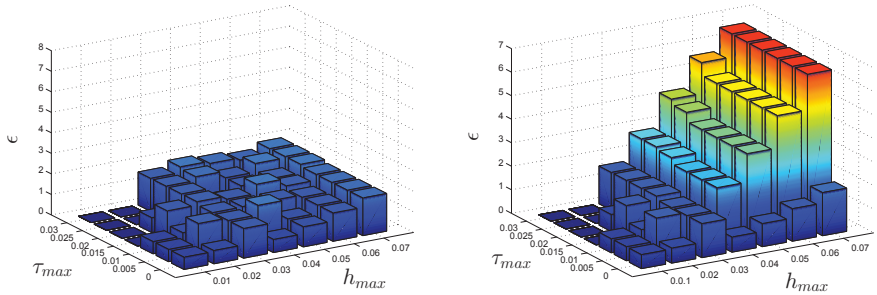
ity regions indicated in Fig. 4.5 provide a quantitative indication of the robust stability region when two parameters of interest, namely h_{max} and τ_{max}, are set to different values. In a similar manner, this toolbox enables the advanced user to easily create other robust stability regions for different parameters.

To further investigate the GNB technique and the resulting conservatism which was observed in Fig. 4.5(b), the toolbox enables the user to access the parameter ϵ for each point (`hmax`, `taumax`) by using the second output from the `isNcsStable` function. The second output is a structure which contains useful information about the overapproximation used for verifying stability. So, the algorithm above can be modified such that the `isNcsStable` outputs `[stable, ovrAprx]` instead of just `stable` and the variable `ovrAprx` contains a property `ovrAprx.MaxEps` which is the ϵ obtained from the GNB algorithm. Plotting this value for each (`hmax`, `taumax`) results in the figures given in Fig. 4.6. As can be seen from Fig. 4.6, GNB1 is able to guarantee $\epsilon \leq 2$ while the ϵ associated with GNB2 is higher due to the 10 grid point limit specified.

This example shows that by using the functions provided in the toolbox within simple iterative procedures, the control community can quickly begin experimenting with these techniques to gain insights which are not readily available from reading the literature.

4.7 Conclusions and Future Development

In this chapter, we introduced a prototype of a toolbox that automates stability analysis of NCSs based on the theoretical contributions in [29, 38, 58, 79]. The toolbox was designed such that the general control engineer can immediately start assessing robust stability of LTI plants and controllers interconnected via a network. Moreover, the software was coded in such a way that the NCS com-



(a) (b)

Figure 4.6: Batch reactor overapproximation tightness, ϵ, plot. (a) shows ϵ for each (h_{max}, τ_{max}) associated with GNB1 and (b) shows ϵ for each (h_{max}, τ_{max}) associated with GNB2.

munity can easily apply the overapproximation theory (used within a discrete-time approach for NCS) on any closed-loop model which is able to be written in the general form (4.1). Finally, the resulting overapproximation can be used in custom conditions for performance or stability analysis, further promoting community development.

This first NCS toolbox prototype focuses only on analysis; however, in future releases, we plan to include automation of the controller synthesis techniques given in [15, 28] and the stochastic results given in [37].

Chapter 5

Experimental Exploration of Wireless Control

"As far as the laws of mathematics refer to reality, they are not certain; and as far as they are certain, they do not refer to reality."

- Albert Einstein

5.1 Introduction

In this chapter, a wireless control system that stabilizes an experimental inverted pendulum setup will be constructed and analyzed. The controller will be computing control commands based on the received measurements from a wireless sensor network (WSN) consisting of two sensor nodes. By logging the network data during the closed-loop experiments, we will see that this fairly simple networked control system (NCS) exhibits very rich behavior in terms of the network-induced effects. The mechanisms which produce this rich behavior are a result of the many years of work on the development of reliable radio communication hardware and the standards to which the hardware adheres. To begin explaining these mechanisms, let us first describe the operation of WSNs in somewhat moderate detail.

WSNs are allowed to operate in the unlicensed 2.4GHz band, in which there are currently three major standards available: IEEE 802.11 for wireless local area networks (WLAN/WiFi), IEEE 802.15.1 for wireless personal area networks (WPAN/Bluetooth) and IEEE 802.15.4 for low-rate wireless personal area networks (LR-WPAN). Due to its low data rate, low power consumption and low cost, IEEE 802.15.4 is a very suitable candidate for battery-powered WSNs. The IEEE 802.15.4 standard specifies the physical layer and the media access control (MAC) layer for LR-WPANs. The ZigBee, ISA100.11a, WirelessHART, and

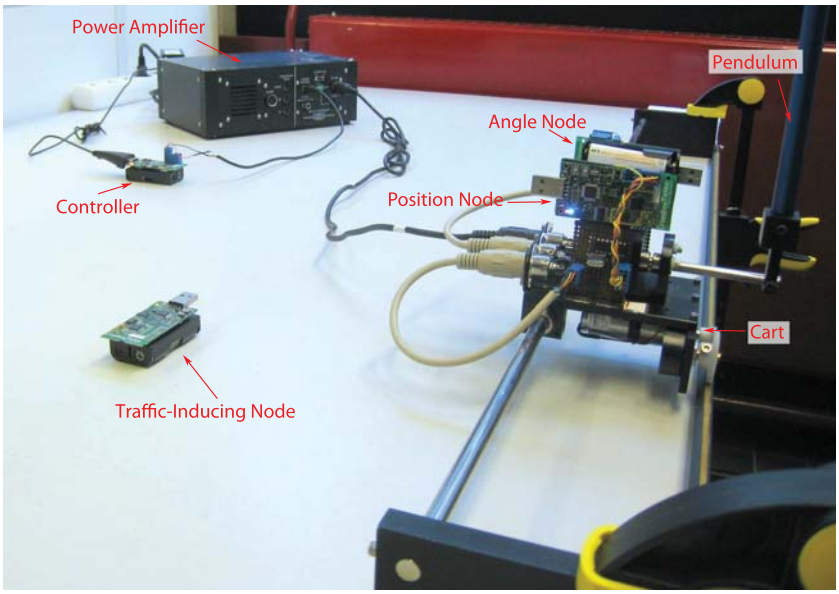


Figure 5.1: Photograph of the wireless control experimental setup.

MiWi specifications are different architectures (commonly referred to as 'stacks') all built upon the IEEE 802.15.4 standard in an effort to create a general networking architecture flexible enough to handle a variety of different applications. In fact, WSNs have been successfully used in a variety of monitoring applications such as forest fire detection [39,70], air pollution monitoring [81], patient health monitoring [27] and structural monitoring [124], just to name a few. Due to the wide variety of applications, open-source operating systems such as TinyOS [75] and the WaspMote API have been developed by computer science communities to enable developers to easily interact with these devices and to contribute to the constantly growing software library.

Despite the success of WSNs in monitoring applications, using wireless communication for real-time feedback control is still in its infancy. The success of WSNs ability to operate satisfactorily in monitoring applications despite the uncertainty in the wireless environment can be attributed to the dynamics of monitoring applications being slow in general where as the dynamics of control are faster (which is especially true for motion control systems). Hence, monitoring applications do not suffer as much from data-rate (i.e. timing) related issues whereas typical control problems do. Some recent experiments on industrial applications using wireless communication for real-time feedback include the development of cooperative adaptive cruise control (CACC) for vehicles [96,97,103] and the development of a ventilation regulation system for industrial mines [36]. Industrial applications are invaluable since they demonstrate the practical useful-

ness of the wireless technology for real-time control and the existing (academic) analysis and synthesis techniques for NCSs. In fact, there are many more theoretical results in the NCS field which would benefit from experimental validation but, using an industrial application for validation is costly. Therefore, it would be beneficial to construct experimental setups that can be accessed easily and allow for inexpensive experimentation, e.g. as done in [8, 61]. Inspired by the work done in [61], in this chapter we will develop a wireless control experimental setup as in [61], but with different implementation aspects and validating different (analysis and design) techniques for NCS.

The experimental setup considered here investigates the use of wireless sensor communication for motion control. The application of WSNs to motion control is of particular interest to companies whose machines experience significant disturbances (due to vibrations of the cables used for communication) and frequent repair (due to cables breaking). Hence, removing cables in certainly beneficial in such motion control applications. However, wireless communication also introduces negative effects. This chapter will investigate these effects when controlling a well-known and exemplary academic motion system, consisting of the pendulum/cart system in an experimental setting, see Fig. 5.1. When using wired communication, the pendulum/cart system is primarily used to investigate classical control theory since an accurate linear state-space model (valid within the desired range of operation) can be obtained from a linearizing a first-principles nonlinear dynamic model [41]. A typical model of this setup consists of four states, two outputs and a single control input. Hence, this setup requires output feedback controller synthesis techniques to design controllers with desired performance. The pendulum/cart system allows the consideration of two possible configurations: the pendulum inverted and the pendulum gantry. The pendulum gantry has a stable equilibrium, which is therefore more suitable for investigating performance properties related to damping out unwanted oscillations. Still, the pendulum gantry can also be used to investigate stability boundaries under network-induced effects as it has been shown in e.g. [29] that a controller designed without network considerations can destabilize the closed-loop, when network-effects are present. However, for stability considerations, the inverted pendulum is more interesting as it has an (open-loop) unstable equilibrium. For this reason, we focus in this chapter on the inverted pendulum configuration of the pendulum/cart system.

Let us now discuss in more detail the experimental setup as shown in the photo in Fig. 5.1. In this photo, the sensor data is being wirelessly transmitted using TelosB devices [105] to a controller node which is directly connected to the actuator. Since there are two sensors, namely the cart position sensor and the pendulum angle sensor, that need to share the same communication channel, we have implemented communication logic according to the Round Robin (RR) communication protocol. In addition to a shared communication medium, it will be shown in this chapter that time-varying transmission intervals, time-

varying delays, packet dropouts and quantization of the sensor readings are present and relevant in the context of this control problem. So indeed, all five network effects mentioned in the introduction to this thesis are present and their significance/impact will be evaluated in practice.

The benefit of considering an academic experimental setup, as the pendulum/cart system, is the ability to be configured both in a 'traditional' wired setting and wireless setting. Opposed to the more experimental wireless setting, the wired setting has the benefit of being able to interface with user-friendly computer aided tools such as MATLAB Simulink. By running the same controller in the wired and wireless setting, we can see first hand the distortion of the measurement and control signals caused by the wireless network. In addition to such an analysis of the closed-loop system, the network itself will be analyzed such that a characterization of the communication network, in terms of the bounds on the transmission intervals and transmission delays, can be determined. With these bounds, the theoretical tools developed in the previous chapters can be applied and validated. The analysis performed using these tools will provide robustness regions for different performance specifications, which aid in *tuning* the controller to achieve more closed-loop robustness with respect to the network-induced effects. We will see that the modular structure of the toolbox developed in Chapter 4 is indeed useful to analyze this NCS experimental setup, as it does not fit into any of the 'standard' NCS setups which are studied in the supporting literature, see, e.g. [38].

Even though wireless control of the inverted pendulum was also achieved in [61], there are various interesting differences in the setup and analysis considered in this thesis. First, the primary difference is that work done in [61] focused mainly on designing the IEEE 802.15.4 'stack' to achieve a certain network Quality of Service (QoS) which is suitable for control applications. The communication network was essentially setup such that the network-induced effects were removed to the largest extent possible. Although this is of course interesting, in this chapter, we consider the network-induced effects as created by using the standard 'stack' provided by the open-source TinyOS [75] operating system. We focus our attention on evaluating these given network-induced effects in order to assess the robustness of the NCS using the existing and developed theoretical tools. Second, here we tune and compare different Linear Quadratic Gaussian (LQG) controllers with each other, both in the wired and the wireless setting, whereas in [61], a single time-varying LQG controller was designed without any comparison. Third, in [61], the so-called 'beacon-enabled' communication was (further) developed and used for communication, whereas in this chapter, we choose to use 'non-beacon' enabled communication which is the standard operating mode for TelosB devices. A comparison of beacon and non-beacon enabled communication will be given in Section 5.2.3. Finally, as being mostly an aesthetic note, we were able to reduce the number of wires by powering the sensor nodes using a 3V battery pack, see Fig. 5.1, whereas the design spec-

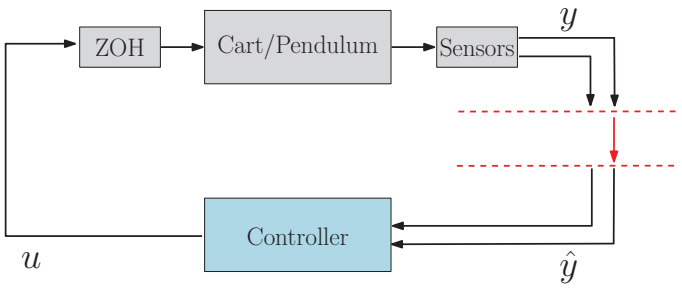


Figure 5.2: Schematic of the NCS.

ified in [61] required a 5V power supply. To the best of the authors' knowledge, the results presented in this chapter form, next to [61], the only other work which experimentally examines an inverted pendulum/cart setup with wireless control using IEEE 802.15.4. Other wireless control experimental investigations on the *rotary* inverted pendulum have been performed using Bluetooth, see, e.g. [40], and WiFi, see, e.g. [104].

The objectives of the experiments conducted in this chapter are to (i) investigate whether the theoretical tools developed and implemented in the toolbox are able to aid in the analysis and design of stabilizing controllers in practice, (ii) gain additional insight into the behavior of a real-life wireless NCS and (iii) possibly identify open problems requiring new or improved analysis/design techniques. With these objectives in mind, the rest of this chapter is organized as follows. In Section 5.2 a description of the experimental setup will be given. This includes a brief description of the pendulum/cart system, descriptions of the (TelosB) wireless transmitters and the communication logic. In Section 5.3, the wireless communication network will be analyzed and characterized in terms of the five general categories of the network-induced effects, as mentioned in the introduction to this thesis. Once the network is properties have been characterized experimentally, a procedure will be given in Section 5.4 on how to tune a LQG controller, which is originally designed for the wired setting, in order to increase robustness when implemented in the wireless setting, and, an experimental comparison will be provided using two different controllers. Next, in Section 5.5, the observer-based synthesis tools developed in Chapter 3 will be applied to the pendulum/cart system. Finally, in Section 5.6 conclusions and recommendations for future research will be given.

5.2 Description of the Experimental Setup

The experimental setup consists of a pendulum/cart system and TelosB motes used for wireless transmission, as shown in Fig. 5.1. The corresponding NCS,

schematically depicted in Fig. 5.2, consists of two sensors, one that measures the cart's position and one that measures the pendulum's angle, that wirelessly transmit their data separately (implying a shared communication medium) to a controller, which is directly wired to the actuator. Each of the sensors and the controller are implemented on separate TelosB motes, as shown in Fig. 5.1.

Remark 5.1. Note that unlike the previous chapters, where both sensor measurements and control commands were transmitted through the shared network, here we assume that only the sensor measurements are transmitted through the network. The reason for this choice is threefold. The first reason is that this 'special case' already exhibits rich enough network behavior to investigate many interesting NCS aspects. The second reason is that in wireless control networks, there must be some computational capabilities at the actuators for message reception/tranmission, which, can be exploited for control. Hence, it is practically relevant to investigate this scenario. Finally, we will show that the prototype toolbox, discussed in Chapter 4, is flexible enough to analyze this 'special-case' network configuration, which requires using the custom model functionality included in the toolbox. ◁

Next, we will describe the pendulum/cart system, introduce the TelosB motes and specify the communication logic the motes will obey.

5.2.1 Pendulum/Cart System

The physical setup we consider is a freely swinging pendulum attached to a cart which can be driven directly, as schematically shown in Fig. 5.3. This system has two degrees of freedom, resulting in a model with four states. The cart-pendulum setup has two optical encoders that measure the position of the cart and the angle of the pendulum. The actuator is a DC motor located on the cart which can drive the cart along one translational degree of freedom. The first-principles based dynamic model of this system (also used in [61]) was derived in [109]. For our purposes, here we will only provide the linearized model given by

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ y &= Cx, \end{aligned} \tag{5.1}$$

where $x = [x_c \;\; \theta \;\; \dot{x}_c \;\; \dot{\theta}]^\top$ with x_c the position of the cart (m), $\dot{x}_c$ is the cart velocity (m/s), θ is the angle of the pendulum (rad), $\dot{\theta}$ is the angular velocity of the pendulum (rad/s) and u is the input force (N) applied by the DC motor. The matrices in (5.1) are given by

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 2.2755 & -6.8848 & -0.0089 \\ 0 & 29.0261 & -15.9037 & -0.0935 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 1.6011 \\ 3.6985 \end{bmatrix},$$

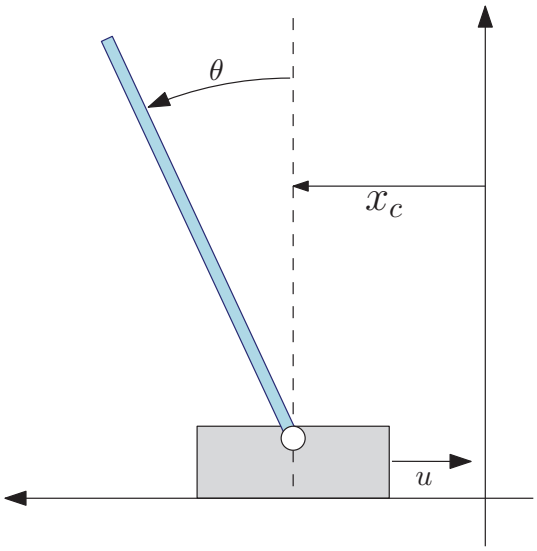


Figure 5.3: Schematic of the pendulum/cart system.

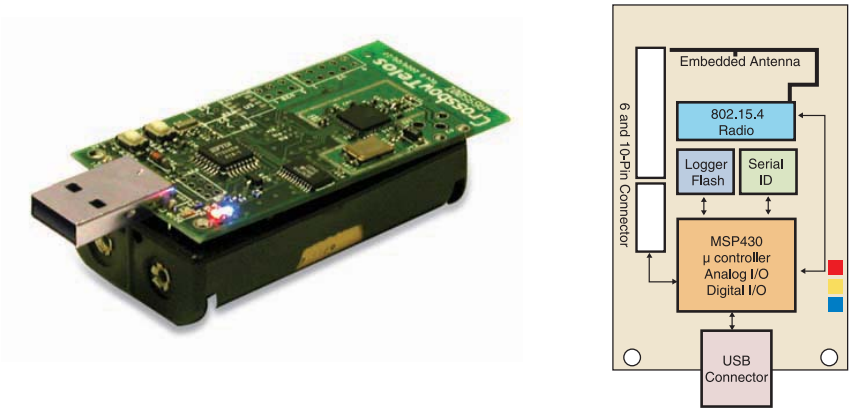
$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$

resulting in the set of eigenvalues of the matrix A above given by $\{0,\ 4.9929,\ -8.1257,\ -3.8455\}$, indicating that the equilibrium of this system is open-loop unstable.

The objective is to synthesize a controller that stabilizes the origin of this system (which corresponds to the upright positions of the pendulum). However, the measurements y, consisting of x_c and θ will be transmitted wirelessly, thereby introducing network-induced imperfections.

5.2.2 TelosB Motes

The wireless devices which connect the two optical encoders to the controller are TelosB motes [105], shown in Fig. 5.4. The TelosB motes are low-powered embedded devices that have been developed for quick prototyping purposes. The TelosB motes communicate wirelessly via a CC2420 radio chip which communicates at 250kbps in the 2.4GHz band and is IEEE 802.15.4 compliant. For computation, the devices have an 8MHz TI MSP430 microcontroller with 48kB ROM and 10kB RAM. Finally, the devices are equipped with 1MB of flash memory for data logging. The controller node will use the flash memory to store the received messages, as well as controller information to be used for network and



(a) TelosB device. (b) TelosB schematic.

Figure 5.4: TelosB device.

control signal analysis.

The devices can be conveniently programmed and accessed via a standard USB connection. The devices are programmed with the open source embedded operating system called TinyOS [75]. The TinyOS code is written in the nesC programming language [48], which is a variant of C. The available software libraries are based on a flexible three layer hardware abstraction architecture (HAA) [53], where the programming level, i.e. the depth of hardware interaction, can be increased as required by the application. TinyOS provides interfaces and components for common hardware abstractions such as packet communication, routing, sensing, actuation and storage.

These devices are ideal for a wireless control experimental setup for a number of reasons. First, they are small, battery-powered, wireless devices which can be easily installed and maintained. Second, their development community is quite large and still very active, whose forums provide an invaluable resource for debugging. Third, the development environment for these embedded devices is very intuitive for someone who is familiar with C programming, which results in quick and easy code adaptation. Finally, the devices are versatile enough to interact with a number of other sensors and devices.

Remark 5.2. The TelosB motes do not have the computational capacity to accurately count the pulses generated by the optical encoder. This results in sensor drift, e.g. the measured value received by the controller contains a time-varying offset from the true value. Any offset in the measurement of the pendulum's angle will prevent the controller from successfully stabilizing the inverted pen-

dulum due to the track limit for the cart. Therefore, a hardware modification is needed to successfully stabilize the inverted pendulum. In this experiment, we augmented the TelosB motes to interact with the LS7366R (a 32-bit quadrature counter) chip and used the TinyOS library functions resulting from the work done in [61]. This hardware modification is shown just below the position node in Fig. 5.1. ◁

5.2.3 Communication Logic

For this experimental setup, we must implement a scheduling protocol since we have each optical encoder reading being transmitted to the controller node separately. In this manner, the radio transmission of the nearby nodes do not interfere with each other. We will implement the Round Robin (RR) protocol which operates via the channel access method known as time division multiple access (TDMA). In this fashion, the RR protocol requires that nodes communicate based on a fixed sequence that periodically repeats itself. Since the node's clocks are not synchronized, a scheduling policy based on the node's internal clock will not guarantee a RR-like operation. There are two alternatives: (i) each node decides when to communicate based on when its predecessor node communicated or (ii) there is a coordinator node which transmits evenly spaced beacons in time, indicating when the individual nodes should transmit. Either of these strategies result in the desired RR behavior.

The choice between these two communication strategies largely depends on the hardware used for wireless communication. Since we are using TelosB devices, the wireless communication is done via the CC2420 radio chip, which imposes the communication protocol to be IEEE 802.15.4 compliant. The IEEE 802.15.4 standard specifies channel access algorithms, data verification and packet acknowledgement to ensure reliable packet delivery. An IEEE 802.15.4 network has two modes of operation: a beacon-enabled mode or a non-beacon mode. We will now provide a brief overview of these two modes; for a thorough overview, the reader is referred to [71].

The first alternative, namely the strategy where each nodes decides when to communicate based on the predecessor node, can be implemented in the non-beacon operation mode. In this mode, a node that wants to transmit uses carrier sense multiple access with 'unslotted' collision avoidance (CSMA/CA) to decide on the exact moment when to transmit the message. Basically, the node senses the medium (in an effort to estimate the network traffic) to determine if the medium is free and then decides to transmit if it is free, or, it otherwise waits a random back-off time until attempting to transmit again. The CSMA/CA mechanism is crucial for reliable packet delivery since the 2.4GHz band is crowded with WiFi, cordless telephone and bluetooth traffic (even microwave ovens emit radiation at that frequency). If each node decides to communicate based on when its predecessor node communicated, as suggested above, then the nodes

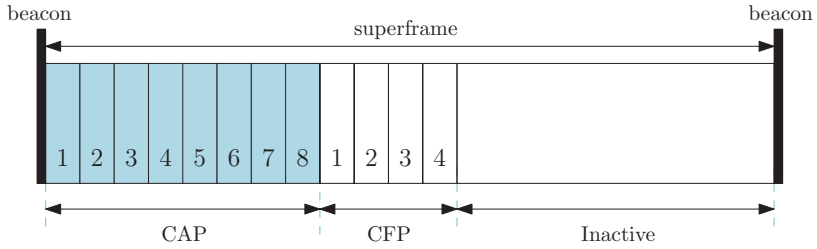


Figure 5.5: An example of the superframe structure when employing beacon-enabled communication. In this example, the contention free period (CFP) has four guaranteed time slots (GTS).

can communicate without competing with each other, but they will still use the CSMA/CA mechanism due to the crowded (uncertain) nature when using the 2.4GHz carrier frequency for communication.

The second alternative, namely the strategy which uses a coordinator node, can be implemented in the beacon-enabled mode. In this mode, a coordinating node is assigned which transmits beacons that indicate equally spaced transmission frames, known as 'superframes'. The equally spaced superframes are divided into a contention access period (CAP), where 'slotted' CSMA/CA is used, and an optional contention free period (CFP), where no CSMA/CA is used, which contains guaranteed time slots (GTS) for specific nodes, as illustrated in Fig. 5.5. The CFP can be used for various purposes depending on the application. For example, it can be used to grant access to nodes which have been unsuccessful at transmitting in the CAP for an extended period of time.

Remark 5.3. One of the coordinator node's main functions is to establish a global reference clock for the rest of the network, thereby synchronizing the clock of every node in the network. This synchronization is very advantageous since it aids in reducing the variation of the transmission intervals throughout the network. When a coordinator node is not present, clock synchronization methods could still be implemented (by sending timing information in each packet) to still reduce the variation of the transmission intervals. However, in general, the clock synchronization problem is quite challenging, especially when unknown delays are present, see, e.g. [42]. ◁

Since the beacon-enabled architectures available for TinyOS devices are still somewhat experimental due to the CC2420 limitations [31], we choose to use the standard TinyOS CC2420 radio stack (non-beacon mode) and simply have each node decide when to communicate based on when its predecessor node communicated. This communication strategy does not suffer from the time critical behavior required by beacon-enabled communication. Work on the implementation

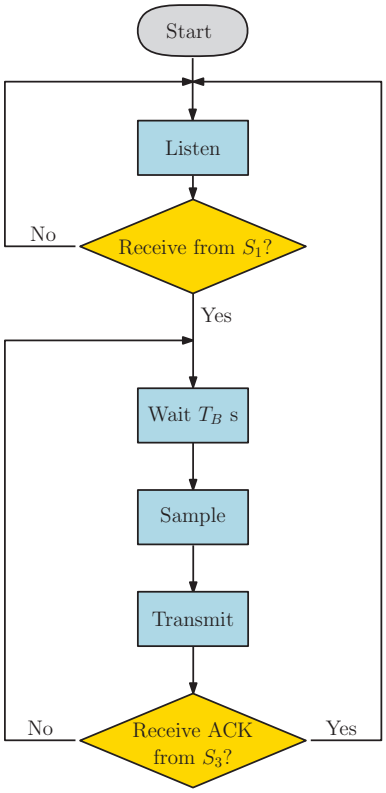


Figure 5.6: Communication logic for node S_2. Node S_1 is the predecessor node which is before node S_2 in the transmission sequence. Node S_3 is the successor node scheduled to transmit after node S_2 in the transmission sequence.

of beacon-enabled communication for TinyOS devices has been done with the TKN15.4 architecture [55] (extended in [61]) and the open-ZB architecture [30]. These beacon-enabled communication architectures for TinyOS still have a few (minor) known issues and are currently under development.

More specifically, the sensor nodes decide when to communicate based on the logic shown in Fig. 5.6. Since the nodes are in a constant state of listening by default, this logic was coded within the `receive` event handler of each node. We programmed the sensor nodes to transmit unicast messages directed to the successor node (scheduled to transmit next) while the controller was programmed to act as a 'snooping' receiver, meaning that it receives all messages transmitted from the sensor nodes. To explain the reasoning for using this logic, consider the following:

- The nodes decide when to transmit based on when they hear that the predecessor node in the fixed transmission sequence has transmitted. Since the individual nodes decide when to communicate, there is no need for a supervisory scheduling/coordinating node in this scenario.

- The use of acknowledgements (ACK in Fig. 5.6) is to increase the robustness toward the transmission sequence ending. Indeed, without acknowledgements the chosen communication logic might result in the ending of all transmissions. Consider the case when a node scheduled to transmit next does not hear that its predecessor node has transmitted, even though it actually did. Then, the node that is scheduled to transmit does not know that it is its turn to transmit and no new transmission would take place anymore, thereby breaking the feedback loop. In the proposed communication logic, each node keeps re-sampling and transmitting new packets until an acknowledgement from the successor node is received. A consequence of using acknowledgements is the introduction of an additional transmission delay. In the proposed communication logic, this additional transmission delay is negligible since the acknowledgement procedure is executed within the MAC layer when unicast messages are used.

- The use of 'Wait T_B s' is to ensure that the controller has enough time to compute the control command and implement the control action before the next packet is received. In this way, the controller will not miss any messages and the 'small delay' assumption (i.e. the delay being smaller than the transmission interval) is guaranteed. The value T_B can be adjusted based on the computational capacity of the controller node and the computational complexity of the control algorithm.

- Notice that acknowledgements are only to be received from the successor node in the transmission sequence. Hence, if the controller is not considered a node (e.g. it does not wirelessly transmit its data), it simply receives the data being transmitted and does not send an acknowledgement (i.e. it does not influence the retransmission of dropped packets).

Remark 5.4. An alternative strategy to packet retransmission is to simply retransmit the packet immediately (without waiting T_B seconds) after an acknowledgement is not received. A potential problem with this scenario is that the controller might not receive the messages which are retransmitted. Consider the case when a packet is successfully received by the controller but not by the other sensor. In this situation, the transmitting node does not receive an acknowledgement from the predecessor node and a retransmission will occur immediately. However, the controller will miss this retransmission since it will be busy computing a control action using the previous message which was successfully received. $\lhd$

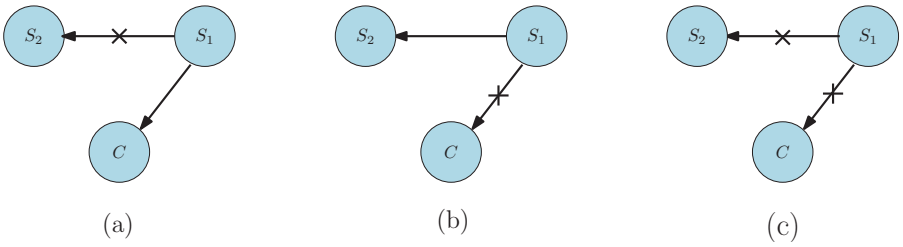


Figure 5.7: Failure modes that can occur when the transmitting sensor node, denoted S_1, is transmitting to the successor sensor node, denoted S_2 and a controller, denoted C. In (a), there is a failure only between S_1 and S_2. In (b), there is a failure only between S_1 and C. In (c), there is a failure between S_1 and S_2 and a failure between S_1 and C.

Remark 5.5. As in the previous chapters, we assume (in Section 5.4.1) that the controller acts in an event-based fashion, in the sense that when it receives a new measurement packet, it immediately updates its state and computes a new control action. Using the proposed communication logic, this implies that the controller updates (which occur at approximately every T_B seconds) are based on only partial measurement information. Another strategy is to have the sensor nodes all transmit (immediately after each other), and, have the controller update its state and control command once all sensor measurements are received. Such a strategy would enable the controller to compute a control command based on more complete measurement information. We intentionally chose the communication logic described above so that we can investigate the (negative) consequences of having the controller update its state and control command based only on partial measurement information, as it is commonly assumed in the NCS literature, see, e.g. [17, 34, 38, 58, 94, 122]. ◁

In this experimental setup, there are two communication links where packets can drop. First of all, a drop can occur between a sensor and the controller, and, second, between the currently transmitting sensor node and the successor sensor node. These transmission 'links' can fail separately or they both fail simultaneously, as depicted in Fig. 5.7. To describe the effect that dropouts have on the timing of the data received by the controller, we will first introduce some relevant terminology. The controller's global reception interval, denoted Δr_k, $k \in \mathbb{N}$, is the time difference between two consecutively received messages from *any* node. The controller's reception interval of only the i^{th} node, denoted Δr_k^i, $k \in \mathbb{N}$, is the time difference between two consecutively received messages from node i. Additionally, we will denote $\Delta \bar{r}_k$ and $\Delta \bar{r}_k^i$ as the average global reception interval and average reception interval of the i^{th} node, respectively.

Each type of failure has a different impact on the controller's global reception

interval[1] and the reception interval of the individual nodes, as explained below:

- only sensor to sensor drop: If a packet drops between sensor node S_1 and the receiving (successor) node S_2, as depicted in Fig. 5.7(a), S_1 will then re-sample and transmit a new message. Since the controller received the previous message, the controller will receive two consecutive messages from S_1 (i.e. $\Delta r_k^1 \approx \frac{1}{2}\Delta \bar{r}_k^1$), and, consequentially, $\Delta r_k^2 \approx \frac{3}{2}\Delta \bar{r}_k^2$, but $\Delta r_k \approx \Delta \bar{r}_k$, as illustrated in Fig. 5.8(b).

- only sensor to controller drop: If a packet drops between a sensor node S_1 and the controller, as depicted in Fig. 5.7(b), S_1 will not transmit a new message and the next message C receives is again from S_2. Hence, the reception interval of S_2 will not be affected (i.e. $\Delta r_k^2 \approx \Delta \bar{r}_k^2$), but $\Delta r_k^1 \approx 2\Delta \bar{r}_k^1$ and $\Delta r_k \approx 2\Delta \bar{r}_k$, as illustrated in Fig. 5.8(c).

- both sensor to sensor and sensor to controller drops: If a packet is dropped both between sensor nodes S_1 and S_2 and sensor node S_1 and the controller C, as depicted in Fig. 5.7(c), then S_1 will re-sample and transmit a new message. In this case, it is as if the dropped message was never transmitted so $\Delta r_k^1 \approx \frac{3}{2}\Delta \bar{r}_k^1$, $\Delta r_k^2 \approx \frac{3}{2}\Delta \bar{r}_k^2$ and $\Delta r_k \approx 2\Delta \bar{r}_k$, as illustrated in Fig. 5.8(d).

- acknowledgement (ACK) drop: It is possible that a sent acknowledgement is not received by the other node. In this case, both nodes will attempt to transmit simultaneously. However, the CSMA/CA mechanism will cause one node to send first, and possibly then successfully receive an acknowledgment. So, although the behavior cannot be predicted deterministically, the transmission sequence will not end.

The analysis above, which considered how dropouts impact the reception intervals, can also be carried out to understand the impact on the periodic transmission sequence, denoted σ_k, $k \in \mathbb{N}$. A packet drop between two sensor nodes, depicted in Fig. 5.7(a), or a packet drop between a sensor node and the controller node, depicted in Fig. 5.7(b), will result in a perturbation of the periodic transmission sequence (i.e. $\sigma_k = \sigma_{k+1}$ for some k), as illustrated in Fig. 5.8(b) and Fig. 5.8(c), respectively. These situations cause the controller node to receive two consecutive messages from the same node without receiving a message from the other node. Although a packet drop by both the successor sensor node and the controller, depicted in Fig. 5.7(c), will have an effect on the reception interval, the transmission sequence will not be affected, as illustrated in Fig. 5.8(d).

[1] Although here we describe how dropouts perturb the controller's reception interval(s), we are, in fact, interested in how dropouts perturb the transmission interval h_k, $k \in \mathbb{N}$, as defined in Chapter 2, 3 and 4. Later, in Remark 5.6, we specify how the transmission interval h_k is defined such that it is related to the global reception interval Δr_k, $k \in \mathbb{N}$.

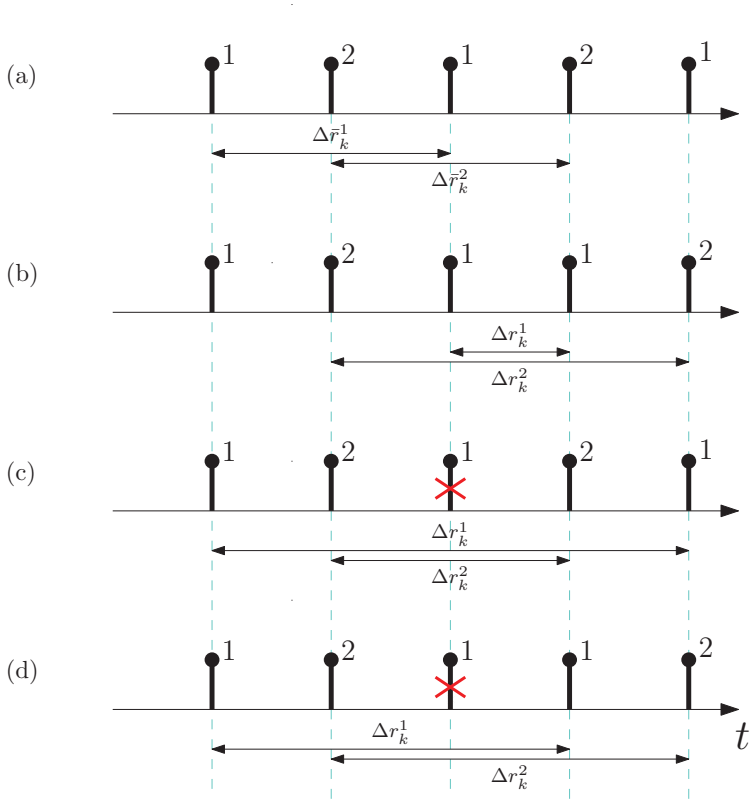


Figure 5.8: An illustration of how different dropout modes, shown in Fig. 5.7, perturb the controller's reception interval and the transmission sequence. The number next to the upward stem indicates which node is received by the controller and an 'x' indicates that the controller dropped that packet. The situations depicted are when (a) no packets are dropped, (b) only sensor node 2 drops a packet sent from sensor node 1, (c) only the controller drops a packet sent from sensor node 1 (d) both the sensor node 2 and the controller drop a packet sent from sensor node 1.

5.3 Network Characterization

To apply the analysis tools described in the previous chapters, we would like to be able to obtain the characteristics of the behavior of the packet dropouts, the transmission delays and the transmission intervals. In this section, we will discuss the presence of the five network-induced effects (listed in the introduction of this thesis) in this experimental setup. For our purposes, the characteristics of the network will be determined experimentally, however, there exist several performance analysis techniques which can also be employed to obtain more (detailed) characteristics, see, e.g. [19].

Before we discuss the network-induced effects, we will first mention the computational delay of the controller node. From measuring the difference between the time when a message is received by the controller node (when the `receive` event is signaled) and the time when the digital to analog (DAC) output of the controller node is updated, we concluded that the controller takes a constant $\tau_C = 0.020$s to compute a control command. Based on this value, we can take $T_B = 0.024$s to ensure that the controller will have computed and updated the control signal before a new packet is transmitted from a sensor to the controller. Next, we will explain how the nodes are programmed to measure the network-induced effects.

The first network-induced effect we would like to measure is the packet dropout behavior. As depicted in Fig. 5.7, each packet transmitted by a sensor node has three failure modes. For our analysis, we would like to not only measure when a dropout occurred but also be able to determine which message was lost and exactly which dropout failure mode occurred. To achieve this, we programmed the sensor nodes to transmit a (monotonically increasing) integer message identifier (also known as a packet sequence number) and an integer node identifier (node ID) along with the encoder reading to the controller. In this way, the controller can log which message numbers were received from which nodes. With this information, we can extract two sequences of message numbers (one from each sensor node), which we denote by MSG_k^i, $k \in \mathbb{N}$, $i \in \{1, 2\}$, and the transmission sequence, which we denote by σ_k, $k \in \mathbb{N}$. This provides enough information to determine the three failure types illustrated in Fig. 5.7. The situation depicted in Fig.5.7(a) occurs when there is only a perturbation in the transmission sequence (without a perturbation in the message number sequence of sensor node S_1). The situation depicted in Fig.5.7(b) occurs when there is a perturbation in both the transmission sequence and the message sequence of a sensor node S_1. The situation depicted in Fig.5.7(c) occurs when there is only a perturbation in the message number sequence of a sensor node S_1 (without a perturbation in the transmission sequence). These conclusions can be drawn with the help of Fig. 5.8 by assigning a (monotonically increasing) message identifier to the messages received from each node.

Along with packet dropouts, we also aim to measure the transmission inter-

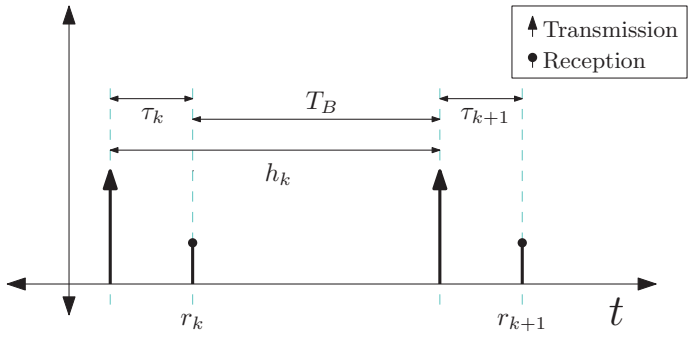


Figure 5.9: Timing diagram.

vals and the delays. The challenge here is that the transmission intervals cannot be *directly* measured due to the facts that (i) the two sensor nodes decide when to transmit independently, (ii) there is no clock synchronization between the devices and (iii) the controller only knows when messages are received, not transmitted. However, the transmission intervals can be reconstructed from the measurements of other network timing parameters. In Fig. 5.9, we have provided a diagram to introduce the relevant network timing parameters. In this figure, $k \in \mathbb{N}$ is the transmission number, $\tau_k \in \mathbb{R}_{\geq 0}$ is the transmission delay, $h_k \in \mathbb{R}_{\geq 0}$ is the transmission interval, $r_k \in \mathbb{R}_{\geq 0}$ is the reception time and $T_B = 0.024$s is the back-off time parameter to ensure the controller has enough time to compute the control command before the next message is transmitted by a sensor. We can conclude from Fig. 5.9 that we can extract h_k from τ_k, τ_{k+1}, r_k and r_{k+1} by using the relationship

$$h_k + \tau_{k+1} = \Delta r_k + \tau_k, \tag{5.2}$$

where, as introduced in Section 5.2.3, $\Delta r_k := r_{k+1} - r_k$ is the global reception interval. Fortunately, the global reception times r_k, $k \in \mathbb{N}$, and the transmission delays τ_k, $k \in \mathbb{N}$, can be measured locally. The global reception times r_k, $k \in \mathbb{N}$, can be measured locally at the controller by programming the device to take a time stamp when the `receive` event is signaled. Moreover, the transmission delay τ_k, $k \in \mathbb{N}$, can be measured locally at each of the sensor nodes by programming the devices to transmit the time difference between when the `send` command is called and the `sendDone` event is signaled. Interestingly, we will see that this time difference is primarily caused by the mechanisms predefined by the TinyOS radio stack to ensure reliable packet delivery. Thus, both the sequence of transmission intervals and the sequence of transmission delays can be determined.

Remark 5.6. Since h_k is derived from the controller's global reception interval Δr_k (indicated in (5.2)), using this equation implies that if a single packet was dropped at the controller, then it is the same as if it was not transmitted. So, although the dropped packet may have been transmitted, the transmission

interval, as derived from (5.2), will be longer than it technically should be. In fact, the transmission interval will be, on average, equal to the global reception interval, see, e.g. Fig. 5.12. However, we are interested in studying the network effects from the controller's perspective, so, a dropped packet is the same as if it was never transmitted. In this way, dropout modeling which assumes a prolongation of the transmission interval can be evaluated. ◁

Remark 5.7. Although the nodes are programmed such that T_B is a constant value, the hardware of the nodes does not guarantee T_B to be exactly equal to 24ms. Hence, by using (5.2) instead of $h_k = \tau_k + T_B$ (as can also be concluded from Fig. 5.9), we avoid the pitfall of assuming that T_B is constant in the analysis while in practice it is not. In fact, included in Fig. 5.10 is a plot of T_B where its value at each transmission time was determined from the measurements τ_k, τ_{k+1}, r_k and r_{k+1}. This clearly indicates that T_B is (slightly) time-varying and it is indeed better to use (5.2) to determine h_k. ◁

With the data collection methods implemented in the nodes as described above, we are now ready to present and analyze the resulting network effects. In the following analysis, the network-induced effects were measured in two scenarios, both of which were measured when the controller was stabilizing the pendulum. The first scenario is when only the two sensor nodes are communicating. The network measurements corresponding to this scenario are given in Fig. 5.10 and Fig. 5.11. The second scenario is when the two sensor nodes are communicating along with two traffic-inducing nodes. These traffic-inducing nodes are additional TelosB motes which were programmed to create network traffic by specifying that they communicate at 20ms intervals. The additional TelosB motes were also communicating in a non-beacon fashion using CSMA/CA. These experiments were performed to emulate a situation where multiple IEEE 802.15.4 devices are sharing a common communication medium. The network measurements corresponding to this scenario are given in Fig. 5.12 and Fig. 5.13.

Transmission Delays: Transmission delays, without the presence of the traffic-inducing nodes, can be seen in the τ_k-plot of Fig. 5.10. Here it can be concluded that $\tau_k \in [0.005, 0.016]$. The transmission delays are varying primarily due to the CSMA/CA mechanism. When two traffic-inducing nodes are present, then, according to Fig 5.12, the range of transmission delays increases slightly to $\tau_k \in [0.005, 0.018]$.

Transmission Intervals: From the h_k-plot in Fig. 5.10, without the presence of the traffic-inducing nodes, it can be seen that $h_k \in [0.029, 0.041]$. The relationship $h_k = \tau_k + T_B$ (derived from Fig. 5.9) implies that the variation of the transmission delay, in fact, induces the variation of the transmission interval since T_B is approximately constant. This relationship can be confirmed by looking at a (h, τ) histogram plot given in Fig. 5.11 and noting that all of the measured data points are clustered around the line $h = \tau + T_B$. Hence, the variation of the transmission intervals can be attributed to the chosen communication logic, i.e. having the nodes transmit based on the time that the predecessor node

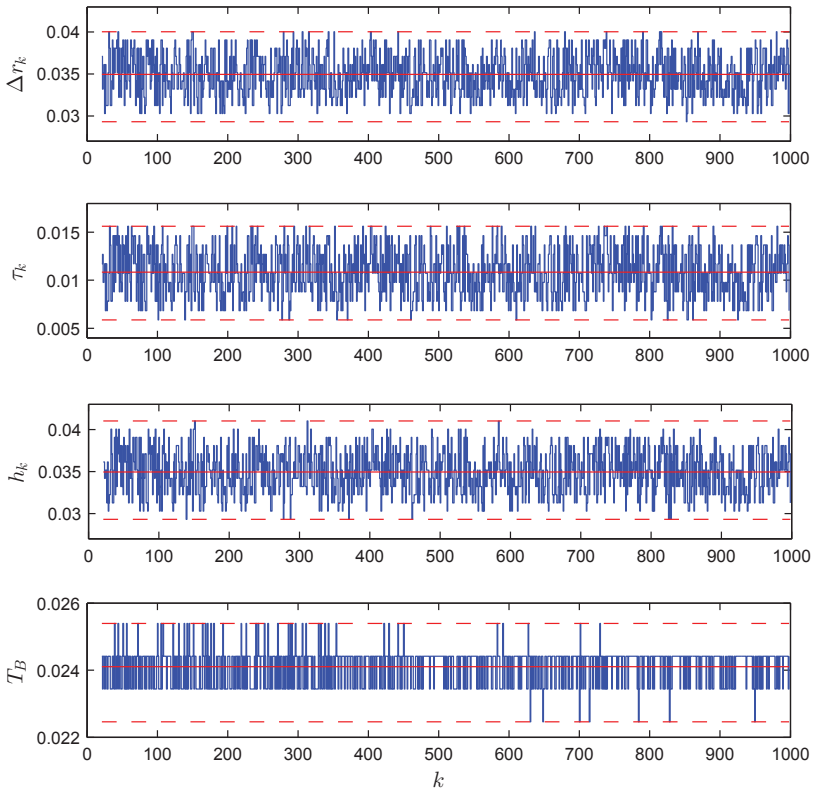


Figure 5.10: Network timing measurements without dropouts present. From top to bottom: reception intervals, transmission delays, transmission intervals and the back-off parameter.

transmitted (due to the absence of a global clock). In the case when two traffic-inducing nodes are present, then, according to Fig 5.12, $h_k \in [0.029, 0.120]$, indicating a maximum of two consecutive dropouts occurred at least once.

Packet Dropouts: As can be concluded from Fig. 5.10, the CSMA/CA mechanism prevents packet dropouts from occurring during the recording of that data set. The result of having traffic-inducing nodes transmitting is plotted in Fig 5.12. We can see packet dropouts occurring in Fig. 5.12 by noticing the spikes in the transmission interval h_k and reception interval Δr_k, compared with Fig. 5.10. Although these traffic-inducing nodes are causing more communication traffic, the CSMA/CA mechanism, which causes the variation in both the

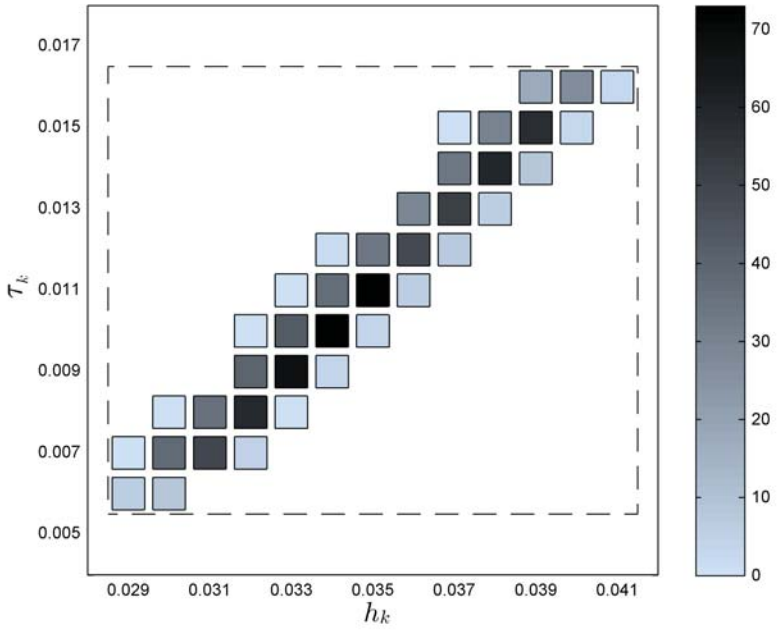


Figure 5.11: Histogram of transmission intervals and transmission delays without dropouts present.

transmission intervals and the transmission delay, is able to reduce the number of dropouts occurring.

Fig. 5.12 gives some indication when dropouts occur. However, in order to determine which node was not able to successfully deliver its message and which of the two communication 'links' failed, we need to consider Fig. 5.13, which is an analysis of the same recorded data as in Fig. 5.12. On the bottom plot of Fig. 5.13, we plot the perturbation of the message sequences, defined by

$$\Delta \mathrm{MSG}_k^i := \mathrm{MSG}_{k+1}^i - \mathrm{MSG}_k^i - 1, \qquad i \in \{1, 2\},$$

and the perturbation of the transmission sequence, defined by

$$\Delta\sigma_k = \begin{cases} -\frac{1}{2}, & \text{if } \sigma_k = \sigma_{k+1}, \\ 0, & \text{otherwise.} \end{cases}$$

Hence, we can see an upward spike if there is a perturbation in the message sequences of the angle node or position node (i.e. when messages transmitted by the angle node or position node were not received by the controller node). The height of the spike indicates the magnitude of the perturbation (i.e. the

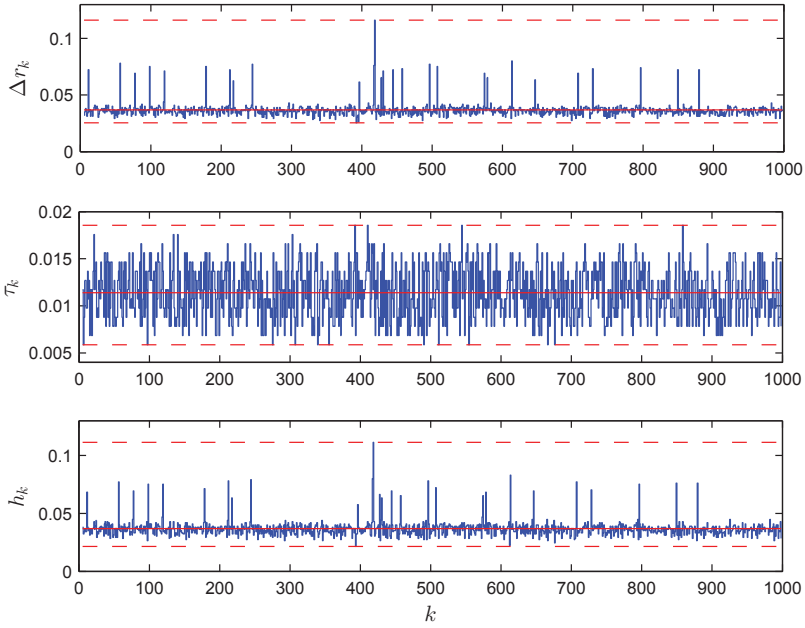


Figure 5.12: Network timing measurements with two traffic-inducing nodes present. From top to bottom: reception intervals, transmission delays and transmission intervals.

number of consecutively missed messages). Also, a downward spike indicates when there was a perturbation in the periodic transmission sequence σ_k. Since there are only two nodes, the perturbation in the transmission sequence can only have a magnitude of at most one. As explained in the beginning of this section, by using these two pieces of data, we can determine exactly where the drop occurred. For example, the first five dropouts all have a perturbation of the angle node's message sequence and experienced a disturbance on the transmission sequence. This behavior, similar to that of Fig. 5.8(c), occurs if a packet dropout was experienced between the angle node and the controller, i.e. as depicted in Fig 5.7(b). Similarly, perturbation of the positions node's message sequence and a disturbance on the transmission sequence indicates that a packet dropout between the position node and the controller occurred. On the other hand, at around $k \approx 850$, there is a perturbation on the angle node's message sequence, but no disturbance on the transmission sequence. This behavior, similar to that of Fig. 5.8(d), occurs when a packet sent by the angle node was dropped by both the position sensor node and the controller, i.e. as depicted in Fig 5.7(c).

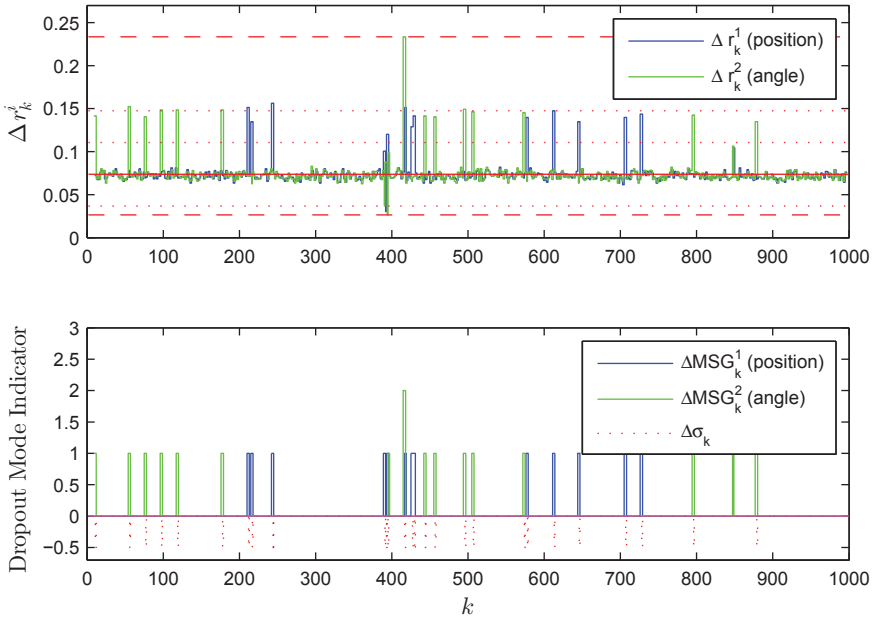


Figure 5.13: Network timing measurements with two traffic-inducing nodes present. The reception intervals of individual nodes Δr_k^i are shown on the top and the perturbations on the node's message sequences ΔMSG_k^i, along with perturbations to the transmission sequence $\Delta\sigma_k$, is shown on the bottom.

Perhaps the most interesting aspect of Fig. 5.13 is a distinct reduction in the individual node's reception interval Δr_k^i-plot just before $k \approx 400$. Actually, around this time, as indicated by the thickness of the line, multiple consecutive dropouts occur, but the dropouts we are interested in are the ones that caused a reduction in the individual node's reception interval. A reduction in the reception interval of an individual node is indicative of a packet drop between the sensor nodes (as in Fig. 5.7(a)) since the node which just transmitted will then transmit again (after T_B seconds). However, in the global reception interval Δr_k-plot and in the transmission interval h_k-plot in Fig. 5.12, no such reduction is seen since the next packet (transmitted by the same node) is received within the 'standard' interval. This type of dropout only perturbs the transmission sequence, not the transmission interval. Hence, modeling dropouts as a prolongation of the sampling intervals or delays, as done in [38, 58, 95], cannot include this type of dropout.

Quantization: Quantization is a result of the optical encoder accuracy (the

number of pulses generated per rotation) and the mode of quadrature counting. In this experimental setup, the encoder accuracy is given by 0.0015 radian/count for the pendulum's optical encoder and 2.275×10^{-5} meter/count for the cart's optical encoder. As such, the only variable left free is the mode of quadrature counting. We used the common 4X mode, which increments the position counter based on every pulse edge (both rising and falling) sent from the optical encoder, and, provides the highest resolution possible. In fact, this 4X mode renders the effect due to quantization negligible. However, it is also possible to use the 1X and 2X modes, which only increments the position counter based on a subset of the pulse edges sent from the optical encoders. This results in a lower resolution and therefore induces a larger quantization effect.

Shared Communication: Since each of the sensor nodes are transmitting separately, the communication medium must be shared. As mentioned before, we are aiming to implement the RR protocol. From Fig. 5.10, we can conclude that the packets are being received, on an average, every 35ms by the controller, and, since the node's identifier is contained in the message, we can confirm that they are being received in a RR fashion.

As can be concluded, all five network-induced effects are present in this simple experimental NCS. The effects which are considered in this thesis, time-varying transmission intervals, time-varying delays and a shared communication medium, are, moreover, *inherently* present in the NCS, i.e. their influence is unavoidable given this network configuration. Packet dropouts, however, rarely occur due to the CSMA/CA mechanism unless traffic-generating nodes are present. It has been experimentally verified that the most dominant cause of packet dropouts for LR-WPANs is due to the so-called 'co-existence' of LR-WPANs with other devices which utilize the 2.4GHz spectrum (such as Bluetooth, WiFi and other LR-WPANs), see, e.g. [80, 102]. Interestingly, the CSMA/CA mechanism induces the variation of the transmission delays (and transmission intervals) in order to reduce the number number of packet dropouts. Thereby, it increases the influence of one (or in this case two) network-induced effect(s) to reduce that of an other. Finally, the influence of quantization appears to be negligible. In the next section, we will observe first hand the consequences of using this wireless network for sensor to controller communication in a feedback motion control loop.

5.4 LQG Controller Tuning & Closed-Loop Analysis

In this section, we will design an LQG controller which will be implemented on a telosB device and used for wireless control experimentation. Since the telosB devices are low-powered, the hardware resources available for computing control commands is limited. In this respect, discrete-time controllers are preferred

over continuous-time controllers since the implementation of a continuous-time control law in digital hardware requires a nearly continuous evaluation of a discretized approximation, i.e. running a discrete-time controller at a very high sampling frequency (>1kHz). The controller that will be implemented in a TelosB device will be in the form of a discrete-time observer-based controller, given by

$$\begin{array}{rcl} \tilde{x}_{k+1} & = & \bar{A}\tilde{x}_k + \bar{B}u_k + L(\hat{y}_k - C\tilde{x}_k), \\ u_k & = & K\tilde{x}_k, \end{array} \tag{5.3}$$

where $k \in \mathbb{N}$ indicates the transmission number, $\tilde{x}_k$ is the controller's estimation of the plant state and, as in Chapter 3 and Chapter 4, due to the presence of the shared communication medium,

$$\hat{y}_k = \Gamma^y_{\sigma_k} y_k + (I - \Gamma^y_{\sigma_k})\hat{y}_{k-1}. \tag{5.4}$$

The matrices $\bar{A}$ and $\bar{B}$ are taken as $\bar{A} = e^{Ah_\star}$ and $\bar{B} = \int_0^{h_\star} e^{As} ds B$ for some constant $h_\star$. We choose the 'nominal' transmission interval $h_\star = 0.035$ as 35ms is the average value of the reception times, indicated in Fig. 5.10. We consider this controller to operate in an event-based fashion, in the sense that when it receives a new measurement packet, it immediately updates its state and computes a new control action.

To determine the numerical values in (5.3), we first will design an LQG controller for the wired setting (i.e. $\hat{y}_k = y_k$, $h_k = h_\star$ and $\tau_k = 0$). The corresponding performance will be experimentally evaluated and considered as a reference performance used for comparison to the situation in which sensor-to-controller communication is performed wirelessly. Then, we will propose a design procedure to tune (and, in fact, reduce) the controller aggressiveness to allow for more robustness to the network-induced effects using the NCS toolbox presented in Chapter 4. Lastly, we will perform closed-loop experiments and analyze the performance of the LQG controller in both the wired and the wireless setting.

5.4.1 Wired LQG Controller Design

In this section we will be tuning the gains K and L of the controller in (5.3) based on LQG principles such that a desired closed-loop behavior is produced when the control system is wired (i.e. $\hat{y}_k = y_k$, $h_k = h_\star$ and $\tau_k = 0$). In this chapter, desired behavior means that the controller is able to stabilize the angle of the pendulum and the position of the cart within a desired set of (narrow) error margins. After a controller with desired performance is designed in the wired setting, we will tune the controller in the wireless setting to determine if/when robustness towards network-effects is guaranteed and to which extent. To design a 'traditional' wired LQG controller, it is well known that the separation principle holds [11,62], i.e. the design of the optimal estimator and optimal state-feedback controller can be done separately.

The design of optimal output injection gain, L in (5.3), is accomplished by solving the linear quadratic estimator (LQE) problem, which assumes that a disturbance, $d \in \mathbb{R}$, and noise, $n \in \mathbb{R}^2$, enter the plant dynamics (5.1) affinely, i.e.

$$\begin{aligned} \dot{x} &= Ax + Bu + Ed, \\ y &= Cx + n, \end{aligned} \tag{5.5}$$

and, informally speaking, assumes that the disturbance and the noise are uncorrelated zero-mean Gaussian stochastic processes with covariance matrices

$$\mathbb{E}[dd^\top] = R_n, \quad \mathbb{E}[nn^\top] = \beta^{-1} Q_n, \quad R_n, Q_n \succ 0. \tag{5.6}$$

The matrices R_n, Q_n and the constant $\beta > 0$ are weights to be chosen by the designer to achieve desired closed-loop behavior (possibly resembling the actual covariance matrices). For this setup, the main component in d is the disturbance which is entering the DC motor. Therefore, we specify $E = B$. Since our controller (5.3) is in discrete-time, the LQE problem we are aiming to solve is to find the matrix L in (5.3) which minimizes the *discrete-time* cost function $\lim_{k\to\infty} \mathbb{E}[||x_k - \tilde{x}_k||^2]$ where this cost function is determined by discretizing the continuous-time plant model (5.5) (using $h_\star = 0.035$) and the disturbance/noise characteristics, for which an analytical solutions exist. The MATLAB function `kalmd` performs this exact discretization and produces the matrix L by solving the resulting discrete-time algebraic Riccati equation (DARE). For a formal derivation and details on the discretization, the reader is referred to [10, p82-83]. With some trial and error, choosing $R_n = 10^5$, $Q_n = I$ and $\beta = 1$ results in what will be referred to as desired observer performance. The parameter β can be increased or decreased to make the observer more or less aggressive, respectively. This type of observer is also commonly referred to as a (discrete-time) Kalman filter.

The design of the optimal state feedback gain, K in (5.3), is accomplished by solving the linear quadratic regulator (LQR) problem, which is finding a control law $u_k = Kx_k$ such that the cost function

$$J(x_0, \mathbf{u}) = \sum_{k=0}^{\infty} x_k^\top Q x_k + \gamma^{-1} u_k^\top R u_k \tag{5.7}$$

is minimized where $\mathbf{u} = (u_0, u_1, ...)$. Similar to the LQE case, the matrices $Q \succ 0$, $R \succ 0$ and the constant $\gamma > 0$ are weights to be chosen by the designer to achieve desired closed-loop behavior. Using the discrete-time plant model $x_{k+1} = \bar{A}x_k + \bar{B}u_k$, the solution to this problem (determining K) can be constructed analytically by formulating the problem as a DARE. The solution can be (and was) efficiently computed using the MATLAB command `dlqr`. With some trial and error, choosing the matrices $Q = \mathrm{diag}(10^4, 10^4, 1, 1)$, $R = 1$ and $\gamma = 1$ results in what will be referred to as desired controller performance. Similar to the LQE case, the parameter γ can be increased or decreased to make the controller more aggressive or less aggressive, respectively.

Now that a controller that produces desired closed-loop behavior in the wired setting has been designed (i.e. when $h_k = 0.035$, $\tau_k = 0$ and $\hat{y}_k = y_k$), we will now elaborate on what is meant by 'desired behavior'. Fig. 5.18 shows the experimental data collected when $\gamma = \beta = 1$. We can conclude that the controller is able to keep the pendulum upright, however, there is an apparent steady-state oscillation on the cart position, the pendulum angle and the input signal. This oscillation is suspected to be primarily caused by friction-related nonlinearities. Indeed, the cart will not respond to an input voltage which produces a force less than that of the opposing static friction force(s), so the controller will keep increasing the force requested until this friction force is overcome and the cart moves to catch the pendulum, thereby causing the steady-state oscillation. Hence, an aggressive controller is desirable since it will ramp up its input more rapidly and be able to keep the pendulum within a very small range. If we decrease the controller aggressiveness by taking $\gamma = 0.04$ and $\beta = 1$, then we have the steady-state response given in Fig. 5.19. As can be concluded, the controller is still able to keep the pendulum upright, however, the steady-state oscillations have larger amplitudes. Finally, decreasing the aggressiveness even further by taking $\gamma = 0.01$ and $\beta = 1$, then we have the steady-state response given in Fig. 5.20 where the steady-state oscillations have even larger amplitudes. Given that the length of the track is 0.99m, the controller with $\gamma = 0.01$, which oscillates with an amplitude of ≈ 0.30m, does not have much room left to react to (network-induced) disturbances. In fact, it was experimentally verified that $\gamma = 0.002$ is the lowest value such that the controller is able to keep the pendulum upright within the track limits. We consider the response given by $\gamma = \beta = 1$ desirable since it can stabilize the pendulum within the narrowest margins (while not producing too much noise on the DC motor input). As γ is decreased then the controller becomes less aggressive and, consequentially, the margins within which the pendulum can be stabilized by the controller increase, which we consider as worse performance. Hence, the controller aggressiveness γ will be used also as our performance characteristic as it is directly related to the desired performance[2][3].

Although the LQG design has the benefit of optimizing the controller parameters to achieve a desired performance, implementing this type of controller over a wireless network violates some of the design assumptions. First, the type of closed-loop disturbances caused by varying transmission intervals and varying

[2]Note that the performance characteristic γ is defined in the wired setting. Although the performance characteristic does not have the same precise interpretation in the wireless setting, we experimentally observe in Section 5.4.4 that decreasing the controller aggressiveness γ in the wireless setting will also cause the cart and pendulum oscillations to increase in a similar manner to that of the wired setting. Therefore, with a slight abuse of notation, we will use γ as the performance characteristic also in the wireless setting.

[3]Although that the value of β also affects the controller's performance, throughout this chapter β is taken as $\beta = 1$ as decreasing β does not have a significant influence on increasing the robustness to network-induced effects, see Remark 5.9.

delays were not taken into account in the design. Second, the controller design assumes that all measurement data is received at each time transmission time t_k, e.g. y_k is fully available. However, with the wireless network present, the sensors must share the wireless medium and, hence, simultaneous reception of both measurements cannot occur. In particular, only $\hat{y}_k$, the networked version of y_k, is available. Hence, implementing such a control law implies (in the manner discussed before) that the observer will update its state, $\tilde{x}_k$, and compute u_k each time an individual measurement is received, as opposed to receiving all measurements. Despite these design violations, we still have two parameters, γ and β, which we can tune to refine the controller in an effort to increase robustness for these network-induced perturbations. The tuning of these two parameters will later be investigated to achieve a desired robustness in the face of network-induced uncertainties.

Remark 5.8. Since the control law is formulated in discrete time and the small-delay case is guaranteed, the transmission delay from either sensor to the controller does not affect the dynamics of the controller. Indeed, the controller applies the same update rule no matter if the sensor value arrives early in the transmission interval or late. However, the transmission delay does influence when the control command u_k is implemented at the actuator. The control command is implemented at the actuator after both the transmission delay and the computation delay. Hence, we can 'lump' these two delays as done in [28,29,128] and assume the overall delay $\bar{\tau}_k \in [\tau_C + \tau_{min}, \tau_C + \tau_{max}]$, where $\tau_C \in \mathbb{R}_{\geq 0}$ is the computational delay and $\tau_k \in [\tau_{min}, \tau_{max}]$ is the transmission delay. ◁

5.4.2 Closed-Loop Modeling

In an effort to find suitable controller parameters β and γ, in (5.6) and (5.7), respectively, for the controller (5.3) in the wireless setting, we will evaluate the stability regions associated with this NCS under different parameter settings for β and γ by using the NCS toolbox presented in Chapter 4. However, the first important observation is that the NCS setup here does not exactly match that of the 'standard' NCS setups included in the toolbox, i.e. are not explicitly studied in the supporting literature (on which the toolbox developments were based). Still, it is possible (and in fact quite easy) to analyze this NCS using the toolbox, provided that the closed-loop NCS model can be expressed in the general form proposed in (4.1). To create a closed-loop model in the form (4.1), we need the following derivations. First of all, we exactly discretize the plant's

dynamics (5.1), leading to

$$\begin{aligned} x_{k+1} &= e^{Ah_k}x_k + \int_0^{\bar{\tau}_k} e^{A(h_k-s)}dsBu_{k-1} + \int_{\bar{\tau}_k}^{h_k} e^{A(h_k-s)}dsBu_k \\ &= e^{Ah_k}x_k + \int_{h_k-\bar{\tau}_k}^{h_k} e^{As}dsBu_{k-1} + \int_0^{h_k-\bar{\tau}_k} e^{As}dsBu_k \\ &= \hat{A}_{h_k}x_k + (\hat{E}_{h_k}B - \hat{E}_{h_k-\bar{\tau}_k}B)u_{k-1} + \hat{E}_{h_k-\bar{\tau}_k}Bu_k, \end{aligned}$$

where, as in Chapter 4, $\hat{A}_\rho = e^{\tilde{A}\rho}$ and $\hat{E}_\rho = \int_0^\rho e^{\tilde{A}s}ds$ and, for this experimental setup $\tilde{A} = A$. Combining the above equation with (5.4) and the discrete-time controller (5.3), results in the closed-loop system model

$$\bar{x}_{k+1} = \tilde{A}_{h_k,\bar{\tau}_k,\sigma_k}\bar{x}_k \tag{5.8}$$

where $\bar{x}_k = [x_k^\top \ \tilde{x}_k^\top \ u_{k-1} \ \hat{y}_{k-1}^\top]^\top$ and

$$\tilde{A}_{h_k,\bar{\tau}_k,\sigma_k} = \begin{bmatrix} \hat{A}_{h_k} & \hat{E}_{h_k-\bar{\tau}_k}BK & \hat{E}_{h_k}B - \hat{E}_{h_k-\bar{\tau}_k}B & 0 \\ L\Gamma^y_{\sigma_k}C & \bar{A} + \bar{B}K - LC & 0 & L(I-\Gamma^y_{\sigma_k}) \\ 0 & K & 0 & 0 \\ \Gamma^y_{\sigma_k}C & 0 & 0 & (I-\Gamma^y_{\sigma_k}) \end{bmatrix}.$$

The RR protocol implemented imposes that

$$\sigma_k = \begin{cases} 1, & \text{for } k \text{ odd} \\ 2, & \text{for } k \text{ even} \end{cases} \tag{5.9}$$

and $\Gamma^y_1 = \mathrm{diag}(1,0)$ and $\Gamma^y_2 = \mathrm{diag}(0,1)$. Now it follows that the closed-loop system (5.8) can be expressed in the form (4.1) by taking

$$A_{\sigma_k} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ L\Gamma^y_{\sigma_k}C & \bar{A} + \bar{B}K - LC & 0 & L(I-\Gamma^y_{\sigma_k}) \\ 0 & K & 0 & 0 \\ \Gamma^y_{\sigma_k}C & 0 & 0 & (I-\Gamma^y_{\sigma_k}) \end{bmatrix},$$

$$B_{\sigma_k} = \begin{bmatrix} I & I & I \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad C_{\sigma_k} = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & 0 & B & 0 \\ 0 & BK & -B & 0 \end{bmatrix},$$

$D_{\sigma_k} = 0$, $E_{\sigma_k} = 0$ and $J_{\sigma_k} = 0$. Hence, we can now use the toolbox and apply the RR analysis tools by using the functions `genPolyOverAprx.m` and `analyzePolyOvrAprx.m` as mention in Section 4.5.2.

5.4.3 LQG Controller Tuning

In this section, we will use the closed-loop NCS model (5.8) along with the NCS toolbox presented in Chapter 4 to tune the LQG controller parameters β and γ specified in Section 5.4.1 for robustness in the wireless setting. First we will provide a procedure to determine coarse ranges these parameters must lie within to satisfy a necessary condition for robust stability. Then we will provide a procedure based on a sufficient test for robust stability to fine tune the values by quantifying the amount of robustness that is able to be guaranteed for different values of β and γ. For this analysis, we will not consider packet dropouts and quantization.

Coarse Tuning of the Weighting Parameters β and γ

First, to coarsely tune the LQG parameters γ and β, we will use (5.8) to determine the regions in the $(h, \bar{\tau})$-space that are stable for *constant* $h_k = h$ and *constant* $\bar{\tau}_k = \bar{\tau}$, $k \in \mathbb{N}$. In this way, we can quickly determine the ranges where γ and β should lie in order to to satisfy a necessary condition for robust stability of the NCS when h and $\bar{\tau}$ are both varying in time. This evaluation can be obtained quickly since it amounts to an eigenvalue test, as for constant h and $\bar{\tau}$, a periodically time-varying closed-loop system is obtained. Fig. 5.14 displays the regions of constant h and $\bar{\tau}$ that correspond to a closed-loop stable system for different values of γ. The region defined by the dashed polygon, which we will denote by Ω_1, represents the region where the combinations of h_k and $\bar{\tau}_k = \tau_k + \tau_C$ were measured, according to Fig. 5.11. If Ω_1 is surrounded by a shaded stability region, then we know at least for each constant $(h, \bar{\tau})$-combination, the closed-loop system is stable for the corresponding value of γ. From Fig. 5.14, we conclude that choosing $\gamma = 1$ will result in having an overall small stability region and, moreover, every point in Ω_1 is unstable for all constant combinations of h and $\bar{\tau}$. Decreasing the controller aggressiveness by choosing $\gamma = 0.01$ will result in having every constant combination of h and $\bar{\tau}$, which is within Ω_1, stable. Recalling that $\gamma = 1$ was controller aggressiveness value which produced our desired behavior, we can see that decreasing γ results in a larger stability region (most probably) at the cost of less desirable performance. We can conclude from Fig. 5.14 that $\gamma < 0.1$ will result in at least a subregion of Ω_1 which is stable for constant values of h and $\bar{\tau}$. Hence, although choosing $\gamma = 1$ resulted in desired performance for the wired case, $\gamma \approx 0.01$ should be chosen to satisfy (at least) a necessary condition for robust stability in the wireless case.

Remark 5.9. The LQG controller parameter β in (5.6) can also be tuned to affect the shape of the stability region. It was observed that, for our particular choice of weighting matrices, changing β does not have a significant influence on the overall size of the region, and, we therefore chose to keep $\beta = 1$ throughout this chapter. However, in general, the stability regions resulting from tuning both parameters should be explored. ◁

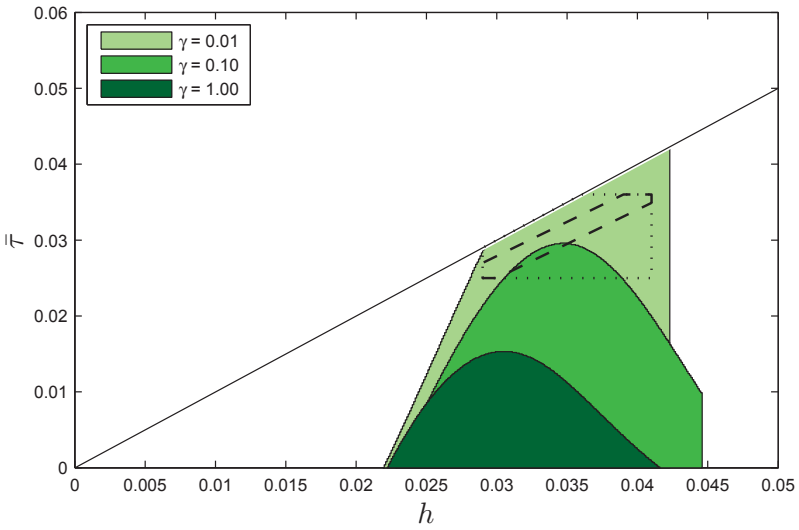


Figure 5.14: Stability regions given by NCS toolbox with RR protocol and h_k and $\bar{\tau}_k$ constant. The observer tuning parameter is set to $\beta = 1$.

Fine Tuning of the Weighting Parameters β and γ

Now that we have a coarse range determined for γ, we can fine tune the LQG parameters by analyzing the NCS with *time-varying* h and *time-varying* $\bar{\tau}$. For this analysis we will rely on the sufficient LMI conditions as obtained in [38] (which are implemented in the toolbox) to verify if robust stability can be guaranteed. What we aim to quantify is how much robustness can be guaranteed for different values of γ. To do so, we introduce the parameter $\delta \in [0, 1]$ which indicates what fraction of the total area of the dashed polygon of interest shown in Fig. 5.14 (which we denoted as Ω_1) can be guaranteed robustly stable. Each value of δ will correspond to a unique polygon Ω_δ which covers a fraction (equal to δ) of the total area. These polygons Ω_δ, $\delta \in [0, 1]$, are defined as indicated in Fig. 5.15, where we used the following philosophy to define them:

1. All polygons Ω_δ, $\delta \in [0, 1]$, include a central point $(h', \bar{\tau}')$, i.e. $(h', \bar{\tau}') \subset \Omega_\delta$ for all $\delta \in (0, 1]$ and $\Omega_0 = \{(h', \bar{\tau}')\}$.

2. For any $\delta' \in (0, 1]$ and all $0 \leq \delta < \delta'$, it holds that $\Omega_\delta \subset \Omega_{\delta'}$.

3. The polygons expand such that (to the best of our ability), first, the regions of highest probability (shown in Fig. 5.11) are covered and, then, expands into the regions of lower probability until the entire region Ω_1 is covered (i.e. $\delta = 1$).

Now that the robustness regions are characterized by δ, we can iteratively run the sufficient robust stability test to determine the maximum δ for which robust stability can be guaranteed for different values of γ (i.e. different performance characteristics). The robust stability test used the GNB overapproximation method and specified the overapproximation tightness, $\epsilon = 0.01$, which results in a worst-case maximum number of 35 grid points.

We care to stress that the most appealing aspect of the GNB overapproximation technique is the fact that it introduces arbitrarily little conservatism when employed in (quadratic) Lyapunov-based stability analysis. More specifically, in [38], it was proven that if the original system (without any overapproximation), is uniformly globally exponentially stable (UGES) in the sense that a parameter-dependent quadratic Lyapunov function exists, the presented LMI-based stability check based on the overapproximation will guarantee UGES and will find a respective parameter-dependent quadratic Lyapunov function, given that the overapproximation consists of a collection of grid points which are sufficiently refined, i.e. the overapproximation tightness ϵ is sufficiently small (see [38, Theorem V.1]). Therefore, in this sense, very little conservatism is introduced by making a convex overapproximation by using the GNB method (at some computational cost).

Remark 5.10. Notice that the GNB overapproximation technique included in the toolbox presented in Chapter 4 (which implemented the theory in [38]) assumes that $(h_k, \bar{\tau}_k) \in \Theta$ where

$$\Theta = \left\{(h,\tau) \in \mathbb{R}^2 \mid h \in [h_{min}, h_{max}], \bar{\tau} \in [\bar{\tau}_{min}, \min\{h, \bar{\tau}_{max}\})\right\}. \tag{5.10}$$

In Fig. 5.14, the region given by the dotted polygon is defined by Θ, as in (5.10), where the bounds h_{min}, h_{max}, τ_{min} and τ_{max} where extracted from the network measurements. We can see from Fig. 5.14 that Θ is a very conservatively chosen superset of the true uncertainty Ω_1 since a significant part of the uncertainty space is included in Θ that does not occur in practice. This introduces unnecessary conservatism in the stability analysis of this practical NCS. Fortunately, restricting our uncertainty region to be of the form (5.10) is not a necessary choice and, in fact, there is no obstruction in taking $(h_k, \bar{\tau}_k) \in \Phi$, $k \in \mathbb{N}$, where Φ is any arbitrary polygon. The toolbox indeed includes this more general case for robust stability analysis thereby avoiding this conservatism (when using the GNB approach). This extension was easily incorporated and further advocates the usefulness of developing software tools.

The extension to analyze arbitrary polygons in the $(h, \bar{\tau})$-space (i.e. $(h_k, \bar{\tau}_k) \in \Phi$, $k \in \mathbb{N}$) can be concluded from [38] by carefully examining the first step of [38, Procedure III.1]. The first step in the procedure specifies that distinct pairs $(h, \bar{\tau})$ be initially selected such that the uncertainty region defined by (5.10) is covered by the convex combination of these initially selected points. However, this initial set of points can be selected to cover *any* uncertainty region in the $(h, \bar{\tau})$-space (such that the small-delay assumption is guaranteed). Therefore, the ability

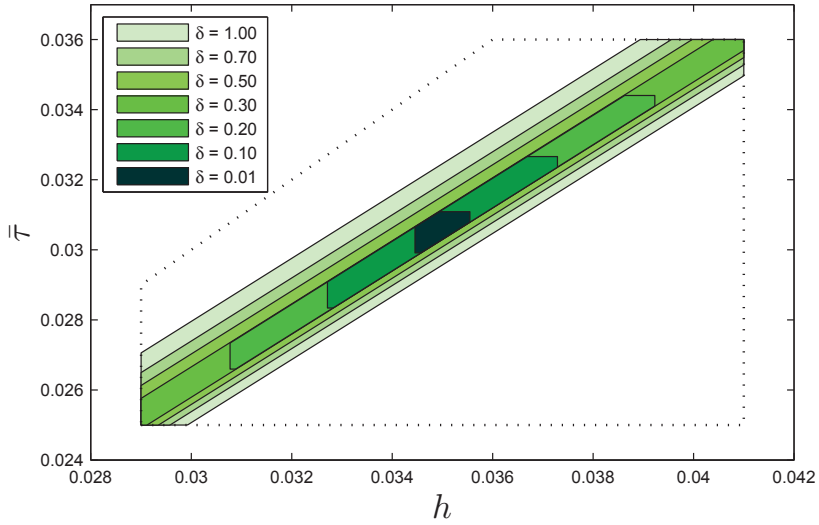


Figure 5.15: Polygons Ω_δ in the $(h, \bar{\tau})$-space corresponding to different values of δ.

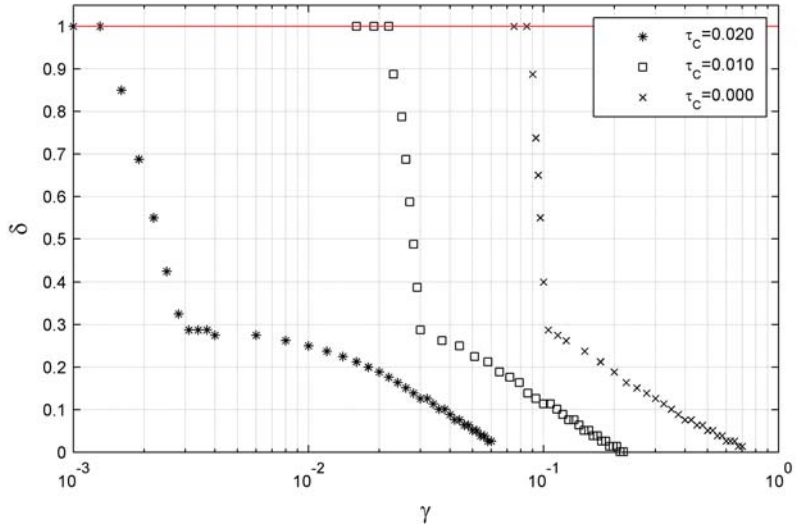


Figure 5.16: Tradeoff between the degree of controller aggressiveness, γ, and the degree of NCS robustness, δ, assuming different computational delays. The observer tuning parameter is set to $\beta = 1$.

of the GNB technique to assess robust stability of *arbitrary* polygons in the $(h, \bar{\tau})$-space avoids introducing the conservatism associated with including parts of the uncertainty space that do no occur in practice. Not all overapproximation techniques possess this property; the overapproximation techniques based on the real Jordan form, see, e.g. [29], the Cayley-Hamilton theorem [50] and Taylor series [65] cannot assess robust stability of arbitrary polygons directly. Although stability analysis can be envisioned by approximating an arbitrary polygon Φ by taking the union of smaller Θ regions, the computational penalty is enormous. $\lhd$

The stability regions in terms of the maximum δ for different values of γ that were obtained by using the sufficient test for robust stability are shown in Fig. 5.16. For the computational delay measured in this experimental setup, $\tau_C = 0.020$, $\gamma \approx 0.06$ is the largest value (best performance) such that robust stability can be guaranteed when $(h_k, \tau_k) \in \Omega_\delta$ for some $\delta > 0$. As γ is decreased (less performance) then the region which is robustly stabilizing increases. In order to guarantee robust stability for the entire region of operation (i.e. when $(h_k, \tau_k) \in \Omega_1$), $\gamma < 0.0015$ must be chosen. However, as explained in the previous section, $\gamma = 0.002$ was the lowest value able to keep the pendulum within the track limits and, hence, the case with $\gamma < 0.0015$ cannot be tested. To further improve the robustness margins, there are two options; redesign the controller for a longer transmission interval, i.e. $h_\star > 0.035$, hoping for more robustness, or consider purchasing faster hardware in order to reduce the computational delay at the controller node.

The results from performing the robust stability analysis assuming different computational delays can quickly be obtained and are also included in Fig. 5.16. The only modification that needs to be made to the robust stability test is that the uncertainty set Ω_1, shown in Fig. 5.14, needs to be shifted downward by the difference of the computational time. If we redo the analysis taking the computational delay, τ_C, to be either 10 ms or 0 ms, the toolbox results in the (γ, δ)-curves shown in Fig. 5.16. In these plots, it can be concluded that reducing the computational delay to 10ms results in being able to guarantee robust stability the closed-loop NCS with $(h_k, \tau_k) \in \Omega_1$ (i.e. $\delta = 1$) for $\gamma < 0.022$, which is an $\approx 1,300\%$ larger γ than when $\tau_C = 0.020$. Moreover, removing the computational delay entirely would enable us to use $\gamma < 0.085$ that will guarantee robust stability for any $(h_k, \tau_k) \in \Omega_1$. Given the network, this provides a procedure to tune the controller's parameters and provide hardware specifications to guarantee different levels of robust stability with a measure of performance.

Remark 5.11. In this analysis, the Lyapunov function candidate was of the form $V(\bar{x}_k) = \bar{x}_k^\top P_{\sigma_k} \bar{x}_k$, $\sigma_k \in \{1, ..., N\}$, (in this case $N = 2$) which is common for all the grid points. An alternative, less conservative, Lyapunov function candidate would be grid-point-dependent, e.g. $\tilde{V}(x_k) = \bar{x}_k^\top P_{\sigma_k, m_k} \bar{x}_k$, with $m_k \in \{1, ..., M\}$ the grid point index. However, although less conservative, guaranteeing arbitrary switching between the grid points using a grid-point-dependent Lyapunov

function $\tilde{V}$ would require $NM^2 + NM$ LMIs to be solved, whereas using a grid-point independent Lyapunov function V would require only $NM + N$ LMIs. This fact raises an interesting question from a computational complexity point of view regarding whether it is more efficient to verify robust stability using fewer grid points with a grid-point dependent Lyapunov function (i.e. a less conservative Lyapunov function candidate), or using many grid points with a grid-point-independent Lyapunov function (i.e. a less conservative overapproximation). ◁

Remark 5.12. The probabilistic information available in Fig. 5.11 was not used in the sufficient test for robust stability analysis. Robust mean-square stability with a higher γ than shown in Fig. 5.16 should be able to be proven using such probabilistic information since a more detailed network model provides more information about the system, thus, leading to less conservative results. Although not implemented in the toolbox yet, the results which include stochastic information, e.g. those in [5, 7, 37], should be investigated and their effectiveness at aiding in controller design should also be evaluated. ◁

Remark 5.13. Although not done here, the SOS stability analysis tools in Chapter 2 can also be applied to this pendulum/cart setup. Despite the fact that the analysis technique in Chapter 2 was based on a continuous-time controller, analyzing discrete-time controllers can be envisioned within the framework of Chapter 2. Moreover, similar to the discrete-time analysis performed in this chapter using the GNB approach, a very general class of shapes in the $(h, \bar{\tau})$-space can be analyzed for robust stability by modifying the hybrid model jump map to include additional polynomial constraints that describe the shape of the uncertainty region. The drawback of using the hybrid SOS analysis on this NCS is that the state dimension is comparable to that of the batch reactor example in Chapter 2 and Chapter 3, which induces a heavy computational burden. The previous work, see [58], which also used a hybrid modeling approach, is not able to exploit the particular shape of admissible $(h_k, \bar{\tau}_k)$, $k \in \mathbb{N}$, based on the assumption on zero lower bounds for transmission intervals and delays (i.e. $h_{min} = \bar{\tau}_{min} = 0$). It seems that incorporating arbitrary shapes in the $(h, \bar{\tau})$-space is not straightforward and, therefore, forms an interesting topic for future research. ◁

5.4.4 Experimental Analysis of the Wireless Control System

In this section we will investigate the consequences of implementing an LQG control law over a wireless communication link. We will compare two different controllers, one with $\gamma = 0.01$ and the other with $\gamma = 0.04$. The analysis results, shown in Fig. 5.16, indicate that $\Omega_{0.10}$ (10% of the uncertainty region) is guaranteed to be robustly stable for the controller corresponding to $\gamma = 0.04$ and $\Omega_{0.25}$ (25% of the uncertainty region) is guaranteed to be robustly stable for the

controller corresponding to $\gamma = 0.01$. The result of implementing the controller with $\gamma = 0.01$ and the controller with $\gamma = 0.04$ via wireless communication is shown in Fig. 5.21 and Fig 5.22, respectively. Although these plots depict that the inverted pendulum is stabilized, some experiments (not shown here) resulted in the cart exhibiting unstable behavior and, consequently, hitting the limits of the track. Nonetheless, comparing the wireless data in Fig. 5.21 and Fig 5.22 with the wired data in Fig. 5.19 and Fig 5.20, the signals associated with the wireless experiments are still periodic but slightly irregular, the amplitudes are noticeably larger and the input signals are noisier. Moreover, compared to their wired counterpart, the data associated with $\gamma = 0.04$ appears to be more irregular and noisy than that of $\gamma = 0.01$. This confirms an intuitive robustness versus performance tradeoff; the more performance we would like to have, the larger the (negative) influence of the network-induced effects becomes. Finally, considering the reception intervals of the individual nodes (Δr_k^i-plots) in Fig. 5.21 and Fig 5.22, we can see that no dropouts occurred during this experiment, as also assumed in the analysis.

The results of the analysis performed in Section 5.4.3 (shown in Fig. 5.16) indicated that the LQG controllers tested (corresponding to $\gamma = 0.04$ and $\gamma = 0.01$) were not completely robustly stabilizing in the presence of the network-induced effects. Although there existed segments of time where the controller was able to keep the pendulum upright, there were also instances where the cart exhibited unstable behavior and, consequently, hit the limits of the track. This suggests that although there might be long periods of time where the closed-loop appears stable, a worst-case sequence (of, e.g. delays) can occur at any time and destabilize the NCS, possibly resulting in damage to the system being controlled. This clearly indicates the importance of developing tools which are able to assess worst-case robustness properties for NCSs, especially for systems which are desired to be left unattended for extensive periods of time.

The fact that the 'traditional' LQG design method chosen was not able to produce a robustly stabilizing controller demonstrates the insufficiency of traditional controller design techniques when they are implemented in this (fairly basic) networked setting. Although we successfully constructed a wireless experimental setup for which current NCS theory can be validated, without a robustly stabilizing wireless controller (which can keep the pendulum upright within the track limits), we can not experimentally determine the amount of conservatism introduced by the sufficient test for robust stability analysis. A promising direction towards achieving a robustly stabilizing controller includes designing a switched LQG observer (via periodic Riccati equation techniques) to account for the shared communication medium, and, also incorporating information regarding the delay. A controller of this type should also be able to be analyzed by the prototype toolbox and, therefore, can also be used for validation purposes. In the next section, we will evaluate the effectiveness of the controller design technique proposed in Chapter 3.

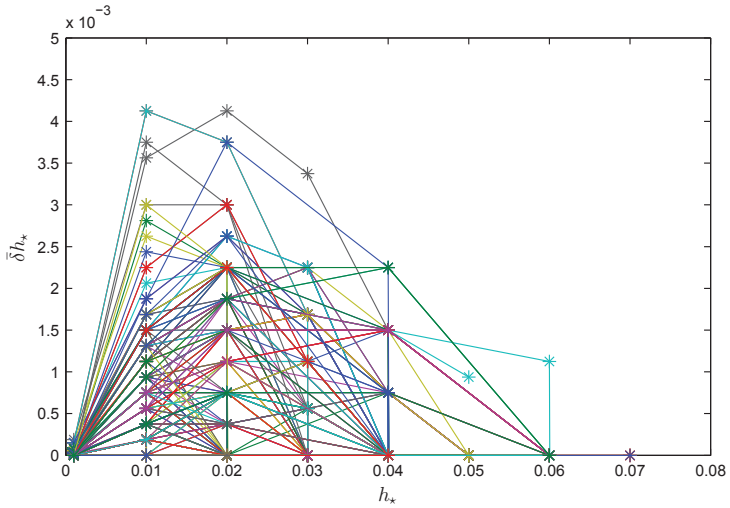


Figure 5.17: Maximum robustness achievable when using different linear coordinate transformation matrices $\tilde{T} \in \mathbb{R}^{4\times4}$, i.e. evaluating state-space models using (5.1) and the linear transformation $z = \tilde{T}x$, where the elements $\tilde{T}_{ij}$ were random values such that $\tilde{T}_{ij} \in [-1, 1]$ for all $i, j \in \{1, 2, 3, 4\}$. The transmission interval $h_k = [\bar{\delta} - h_\star, \bar{\delta} + h_\star]$, the computation delay $\tau_C = 0$ and the transmission delay $\tau_k = 0$.

5.5 Robust Observer-based Controller Synthesis

In this section we will apply the synthesis techniques developed in Chapter 3, to the linearized plant model (5.1). Unlike previously where $h_\star = 0.035$, we will synthesize controllers for different values of $h_\star$ and determine how much robustness can be guaranteed. Although, the synthesis technique presented in Chapter 3 did not explicitly include delays in the NCS model, in fact, delays can be included, see Remark 3.6. Moreover, the technique can be used to synthesize controllers which are robust to uncertainties of arbitrary shapes in the $(h_k, \bar{\tau}_k)$-space due to the fact that it is based on the GNB overapproximation technique, see Remark 5.10. Therefore, the framework is general enough to create a $(h_\star, \delta)$-plot, where δ was defined in Section 5.4.3, which is based on synthesizing of robustly stabilizing controllers. However, in this section, we limit the analysis to the case where delays are not considered, i.e. $\bar{\tau}_k = 0$. Specifically, for each $h_\star$ to be chosen in the observer-based controller (3.9), we aim to find the largest $\bar{\delta} \in [0, 1)$ such that the NCS (3.11) is stable for $[h_{min}, h_{max}] = [(1 - \bar{\delta})h_\star,\ (1 + \bar{\delta})h_\star]$, $k \in \mathbb{N}$, i.e. we consider a symmetric uncertainty interval around $h_\star$.

The synthesis techniques in Chapter 3 cannot *directly* produce any stabiliz-

ing controllers for any $h_\star$ using the plant model as defined in (5.1) (even without shared communication and $h_k = h_\star$). However, robustly stabilizing controllers (which include shared communication) could be found when the synthesis technique was applied to algebraically equivalent plant models (i.e. when using a linear state coordinate transformation). Fig. 5.17 provides a resulting plot of the maximum robustness achievable for different linear coordinate transformation matrices $\tilde{T} \in \mathbb{R}^{4\times4}$, i.e. evaluating state-space models using (5.1) and the linear transformation $z = \tilde{T}x$, where the elements $\tilde{T}_{ij}$ were random values such that $\tilde{T}_{ij} \in [-1, 1]$ for all $i, j \in \{1, 2, 3, 4\}$. Recalling that for this experimental NCS setup $h_k \in [0.029, 0.041]$ (implying that $h_\star = 0.035$ and $\bar{\delta}h_\star = 0.006$), we can conclude that the synthesis technique is not able to find a robustly stabilizing controller which covers this entire range, even with $\tau_k = 0$. So, although the synthesis technique was able to theoretically find stabilizing controllers, the robustness margins are small. Besides, is currently not equipped to provide performance guarantees. The origin of this conservatism is suspected to be due to the choice of the state coordinate representation, as indicated in [116]. To the best of the authors knowledge, the determination of a suitable linear coordinate transformation matrix $\tilde{T}$, even in the simpler setting of static output feedback stabilization of linear time-invariant systems, is currently still an open problem. This indicates that a future research topic could be the investigation into the origin of this conservatism and how it can be eliminated before extending the technique to include performance.

5.6 Conclusions and Future Recommendations

In this chapter, a wireless control experimental setup was presented. In this setup, two sensor nodes shared network access to a wireless network using a RR protocol when communicating to the controller. The controller itself was wired to the actuator. It was shown that the NCS toolbox presented in Chapter 4 was useful in quantifying the amount of robustness achievable for a given performance characteristic. In fact, a traditionally designed LQG controller could be tuned to increase robustness of the NCS at the cost of decreasing transient performance. Hence, the theoretical tools implemented in the prototype toolbox were successfully employed to aid in the stability analysis and controller design for this practical NCS. In addition to validating the developed theory, many new insights were gained.

We have seen that plotting a time-varying transmission interval versus time-varying delay histogram (based on experimental data) is extremely useful in order to accurately define a bounding set for these network-induced uncertainties. This experimentally determined bounding set was, in fact, only a subset of the corresponding bounding set that was commonly adopted in the NCS literature. Therefore, using this commonly adopted bounding set will introduce

conservatism in the stability analysis as it will include additional uncertainties which do not actually occur in practice. The limitation of the commonly adopted bounding set is due to the assumptions which predetermine its geometry (once the bounds on the transmission intervals and delays were determined). Therefore, the direct application of several analysis techniques such as those based on the real Jordan form, the Cayley-Hamilton theorem and the Taylor series expansion will be conservative as they can only analyze uncertain regions characterized by such assumptions. These techniques can approximate an experimentally-determined bounding set by taking the union of multiple sets which obey the geometric assumptions; however, this will incur a heavy computational burden on the stability analysis. Therefore to reduce this type of conservatism in the analysis, analysis techniques that are developed in the future should be able to directly analyze arbitrarily shaped sets that bound the transmission intervals and delays (such as the GNB technique and the SOS technique).

Also new insights were gained related to the different ways that packet drops can occur in the experimental setup. It was determined that, using the implemented communication logic, there were two communication links that could fail which result in possible three failure types for each sensor node. Each failure affects the reception/transmission intervals and the transmission sequence differently. Therefore, to accurately capture this behavior, dropout models should result in inducing a combination of both a perturbation on the reception/transmission interval and a perturbation on the transmission sequence. Dropout models which only consider dropouts to prolong the transmission interval cannot capture dropouts that only perturb the transmission sequence (i.e. between two sensor nodes). Therefore, it of interest to develop packet dropout models capturing this combined behavior and to develop corresponding robust stability analysis techniques.

Moreover, when a dropped packet did perturb the transmission interval, the perturbed transmission interval was approximately twice that of the average 'regular' transmission interval. As a result, there will be two disjoint bounding sets in the uncertainty space, one centered at the average of the 'regular' transmission interval and one centered at twice this 'regular' value. Assuming a prolongation of the transmission interval would result in analyzing a superset which includes all the points which define these two disjoint polygons and, therefore, include parts of the uncertainty space which do not occur. This will, in general, significantly increase the conservatism. Hence, it would be beneficial to develop techniques (based on LMI conditions) which can analyze the union of two disjoint bounding sets in order to reduce conservatism compared to the standard prolongation of the transmission interval.

The developed theoretical tools which were applied to this NCS were based on a 'worst-case' modeling of the uncertainties on the delays and transmission intervals. However, the wireless communication based on the CSMA/CA mechanism is fairly 'well-behaved' in the sense that a particular sequence, which might

be considered the worst-case (e.g. $\bar{\tau}_k = \bar{\tau}_{max}$ for all k), might never occur (or with a low probability). Therefore, it is of future interest to analyze stochastic tools for NCSs, to implement them in the toolbox and to study the application of these results in an experimental setup.

Other than the research topics raised from the insight mentioned above, this experiment has lead to the discovery of several other possible future research topics:

1. A comparison between non-beacon enabled communication and beacon enabled communication should be performed and analyzed.
2. A comparison between the current IEEE 802.15.4 setup and setups which use IEEE 802.11 (WiFi) and IEEE 802.15.1 (Bluetooth) compliant transmitters should be done.
3. The implementation and investigation of other protocols (i.e. the try-once-discard (TOD) protocol or the roll-out protocol [4]) over a wireless link should be investigated.
4. An investigation with other controller types (e.g. a switched Kalman filter) might provide more robustness with respect to network-induced effects.
5. Implementing the controller on a node which has more computational power (i.e. lowering the computational delay τ_C) should be investigated to determine if robust stability of this experimental NCS can be achieved.
6. In addition to having the sensors transmitting data wirelessly to the controller, the controller-to-actuator channel could also be made wireless. In this scenario, the input is updated much less frequently, so computing and sending a packet containing several future control actions to the actuator node could be experimentally studied and validated.
7. Adapting the SOS techniques presented in Chapter 2 such that this experimental NCS containing a discrete-time controller can be investigated and is of practical intrest.
8. The synthesis techniques developed in Chapter 3 were shown to be fairly conservative when applied to this inverted pendulum model. It is of interest to investigate what is the exact cause of the conservatism and how it can be eliminated.
9. The validation of NCS simulation software, such as ones, e.g. described in [24], should also be investigated with this experimental NCS setup.

Hence, this experimental exploration provided several new insights and introduced many new and exciting research questions. The open issues raised above demonstrate that clearly.

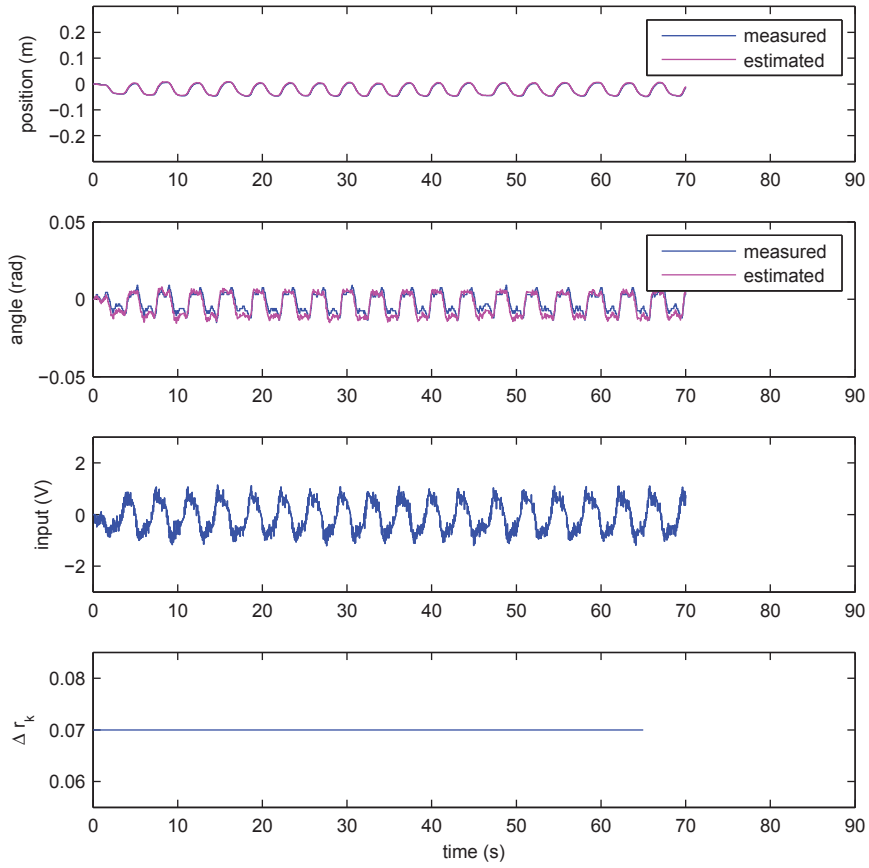


Figure 5.18: Controller performance with $\gamma = 1$ and $\beta = 1$ and sensor communication is wired.

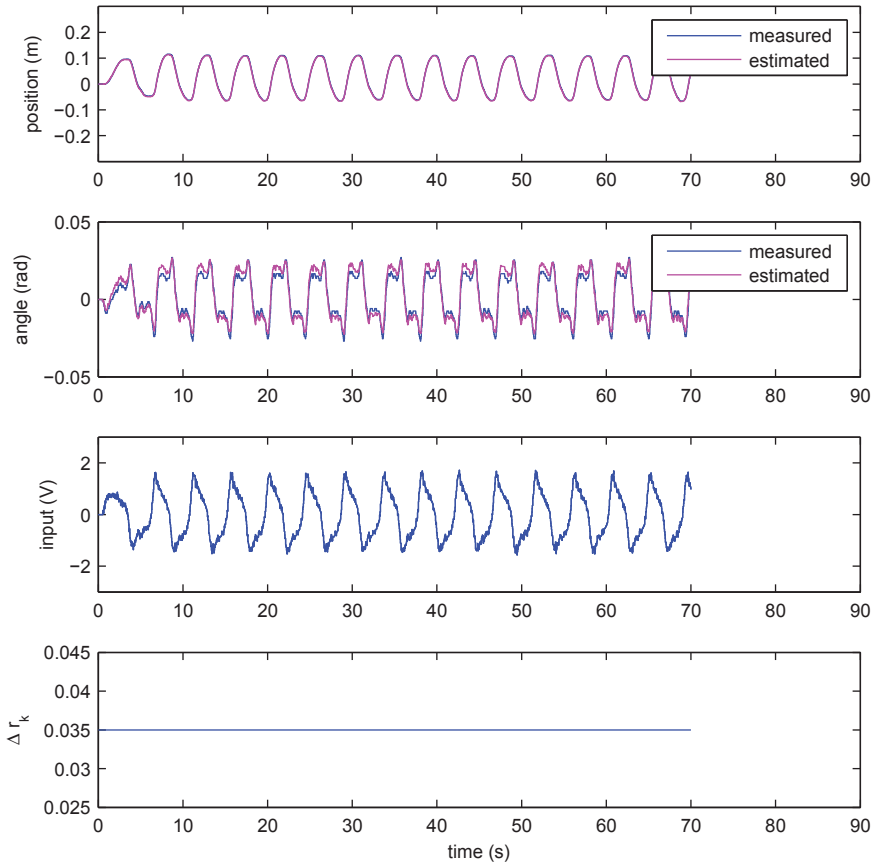


Figure 5.19: Controller performance with $\gamma = 0.04$ and $\beta = 1$ and sensor communication is wired.

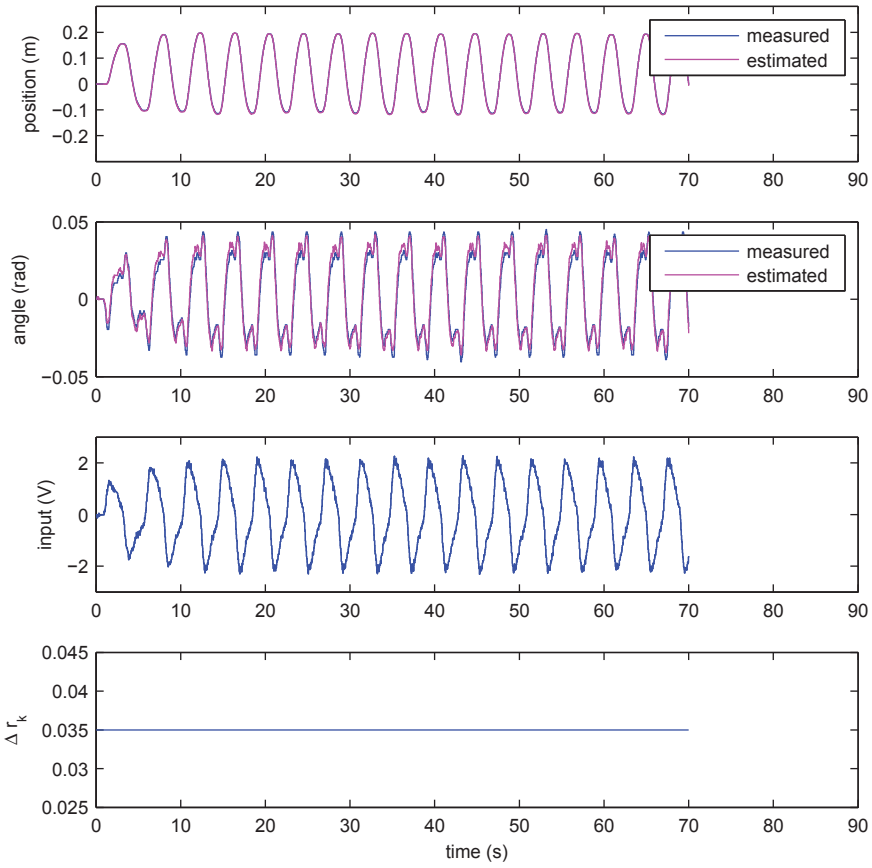


Figure 5.20: Controller performance with $\gamma = 0.01$ and $\beta = 1$ and sensor communication is wired.

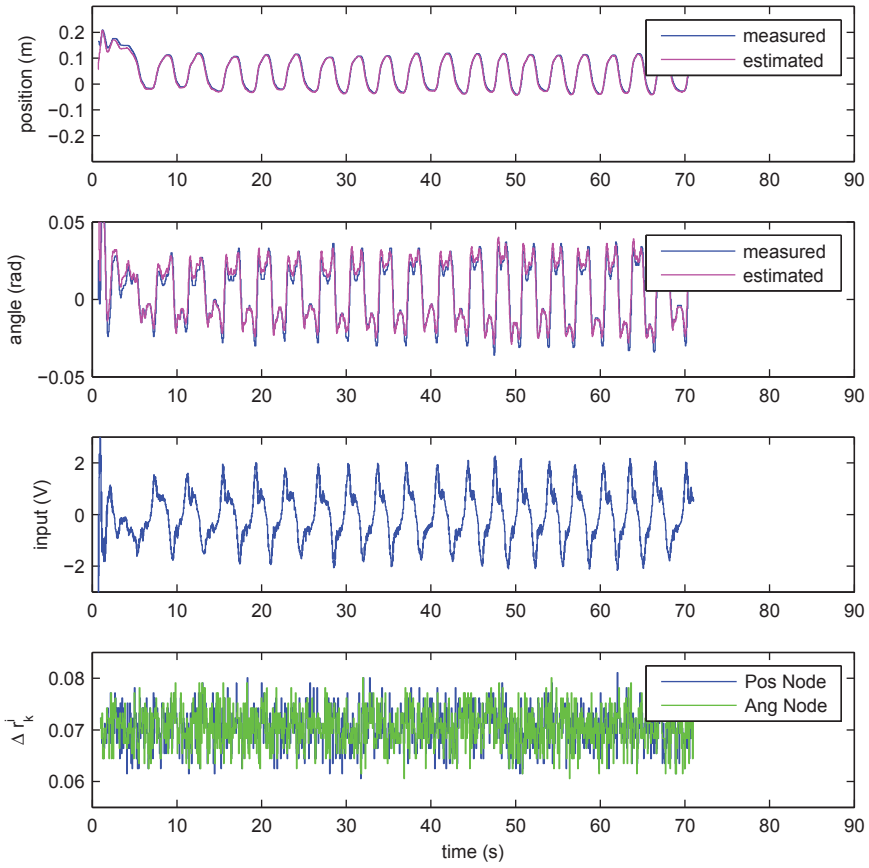


Figure 5.21: Controller performance with $\gamma = 0.04$ and $\beta = 1$ and sensor communication is wireless.

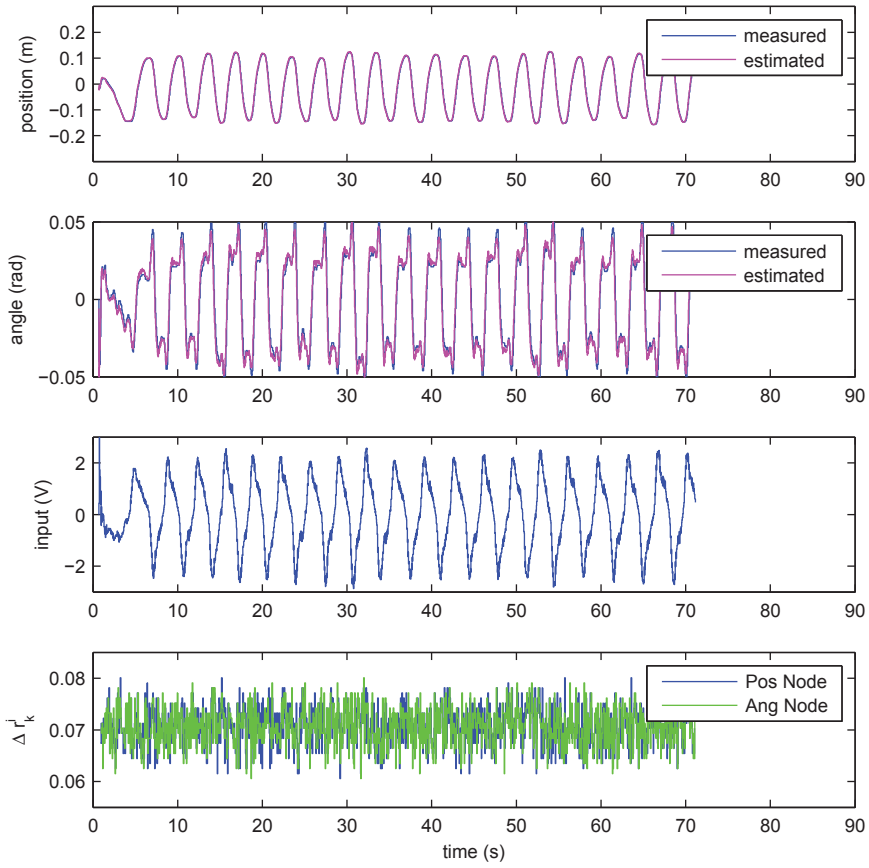


Figure 5.22: Controller performance with $\gamma = 0.01$ and $\beta = 1$ and sensor communication is wireless.

Chapter 6

Conclusions, Recommendations and Final Thoughts

This thesis will close with some concluding remarks, recommendations for future research and final thoughts.

6.1 Conclusions

In this thesis, specific challenges in the field of networked control systems (NCSs) were addressed. This field of research is motivated by the realization that in many current control problems, communication between sensors, actuators and controllers is not always perfect. These imperfections stem from the fact that current telecommunications advances, such as wireless communication, cannot always guarantee the high quality of service (QoS) levels which are required to successfully implement controllers that are based on 'classic' control theory. Specifically, these imperfections perturb the timing associated with control-related data (i.e. when data is transmitted and received) and limit the amount of control-related data which can be communicated at any given time. If not accounted for, these imperfections can degrade performance and even threaten closed-loop stability of the control system. Therefore, it is crucial to develop tools which can quantify control-relevant properties such as stability and performance in the case of imperfect communication between sensors, actuators and controllers. To cope with the corresponding challenges, this thesis provided the following main contributions:

(i.) the development of theoretical tools for robust stability analysis of NCSs was provided in Chapter 2,

(ii.) the development of theoretical tools for robust (decentralized) controller synthesis for NCSs was provided in Chapter 3,

(iii.) the development of NCS analysis software to provide easy access to stability analysis tools for control engineers and to facilitate the future development of the current theory was provided in Chapter 4,

(iv.) the validation of tools and theory on a wireless control experimental case study was provided in Chapter 5.

We will now describe these contributions in more detail.

In the area of robust stability analysis, Chapter 2 presented a method which can be used to analyze robust stability of nonlinear (piecewise) polynomial plants and controllers with respect to network-induced effects. The proposed solution was formulated in a hybrid system modeling framework and was based on sum of squares (SOS) techniques. The network-induced effects considered were bounded time-varying delays, bounded time-varying transmission intervals and a shared communication medium. This technique was shown to be less conservative than previously proposed techniques which can analyze robust stability of nonlinear (piecewise) polynomial plants and polynomial controllers, see, e.g. [58, 94]. Moreover, there are several other beneficial features of the technique proposed in Chapter 2: (i) it allows for non-zero lower bounds on the delays and transmission intervals in contrast with various existing approaches, e.g. [22,25,58,90–92,94,95], (ii) it allows more flexibility in the Lyapunov functions thereby obtaining less conservative estimates of the maximal allowable transmission intervals (MATI) and maximal allowable delay (MAD) than prior emulation-based approaches, see, e.g. [58, 94], (iii) it provides an automated one-shot method to address stability analysis problems of nonlinear systems in the NCS setting, and finally (iv) the framework was general enough to incorporate arbitrarily shaped uncertainty sets that bound the transmission intervals and delays, as mentioned in Remark 5.10. Although the technique was shown to be less conservative than previously proposed techniques, the price payed was that the computation time required to solve the SOS program grows quite significantly with the size of the state space dimension.

In the area of robust decentralized controller synthesis, Chapter 3 presented one of the first techniques which can synthesize decentralized observer-based controllers which are robustly stabilizing in the presence of network-induced effects. The proposed solution was formulated in the discrete-time linear switched system modeling framework and was based on polytopic overapproximations with norm-bounded uncertainties. The network effects considered were bounded time-varying transmission intervals and a shared communication medium, where the extension to bounded time-varying delays is straightforward using the work in [38]. To effectively deal with the shared communication medium using observer-based controllers, we adopted a switched observer structure that switched based on the received measured outputs and a switched controller structure that switched

based on received control inputs at each transmission time. The developed linear matrix inequality (LMI)-based synthesis conditions, if satisfied, provided stabilizing (observer and controller) gains for both the decentralized problem setting and the NCS problem setting in isolation, as well as the unification of these two problem settings. Using a benchmark example in the NCS literature, it was shown that this synthesis technique was able to find an entire set of controllers that significantly improved the closed-loop robustness compared to that of an existing dynamical controller, extensively studied in the NCS literature. Due to the fact that this synthesis technique is based on the gridding and norm bounding (GNB) overapproximation procedure, the extension toward the synthesis of controllers which must be robust for arbitrarily shaped uncertainty sets that bound the transmission intervals and delays can easily be done. Although we aimed to develop a synthesis technique applicable to large-scale problems (hundreds to thousands of states), the applicability of this technique is limited to analyzing small- to medium-scale problems (i.e. tens of states) due to computational complexity. This limitation is primarily due to the fact that, although offering low levels of conservatism and efficient verification for small-scale problems, the number of variables that must be solved using a (switched) quadratic Lyapunov function candidate grows polynomially with respect to the state dimension.

In an effort to make our developed theory available to the control community, Chapter 4 presented a prototype NCS toolbox. Specifically, the toolbox can be employed to efficiently verify if a linear time-invariant (LTI) plant and an LTI controller interconnected with a shared network is robust to certain network imperfections. The toolbox was developed so that it appealed to both basic users, e.g. students or control engineers that are not so familiar with NCS theory, and advanced users, e.g. researchers that would like to use custom models and custom analysis conditions. By using this toolbox, the user is able to make multi-disciplinary design tradeoffs between control properties, such as stability and performance, and network-related properties, such as delays, scheduling, bandwidth limitations, etc., in a user-friendly manner. This toolbox provides a user-friendly way to interact with the existing theoretical developments and removes the burden of implementing some of the more complex algorithms employed in the theory.

Finally, experiments with wireless control were conducted in Chapter 5. The experimental setup chosen was the wireless control of an inverted pendulum/-cart system. The wireless nodes used to communicate between the sensors and the controller were programmed in such a way that the theory and prototype toolbox developed in this thesis could be applied and validated. After a detailed experimental analysis of the network effects induced by the wireless communication, it was determined that indeed all five network-induced effects were present; however, the three effects focused on in Chapter 2 and Chapter 3, namely time-varying transmission intervals, time-varying delays and a shared communication medium, were shown to be dominant. Although dropouts were present with

the introduction of traffic-inducing nodes and new insights into the dropout behavior were gained, dropouts were not considered in the analysis. Applying the prototype toolbox to analyze robust stability of the NCS with respect time time-varying transmission intervals, time-varying delays and the presence of a shared communication medium resulted in providing tradeoff plots which quantified how much robustness could be gained at the cost of decreasing transient performance. Experiments indicated that the controllers, although able to induce stabilizing behavior for a period of time, were not robustly stabilizing in the presence of the network-induced effects, as indicated in the analysis. This suggests that although there might be long periods of time where the closed-loop appears stable, a worst-case sequence (of, e.g. delays) can occur at any time and destabilize the NCS, possibly resulting in damage to the system being controlled. This clearly indicates the importance of developing tools which enable us to assess worst-case robustness properties for NCSs, especially for systems which are desired to operate autonomously.

6.2 Recommendations for Future Research

There are still many directions to explore in the area of networked control systems. Determining which direction to pursue in order to make the theory more applicable in practice is an important, though challenging, task. As a result of the work done in this thesis, several directions could be considered.

Reducing the curse of dimensionality: Although being able to demonstrate low levels of conservatism, the applicability of both the theoretical developments in Chapter 2 and Chapter 3 are limited by the dimension of the closed-loop state vector due to computational burden imposed by LMI-based stability analysis and controller synthesis techniques. One of the main benefits of using shared communication channels is the ability to inexpensively install large-scale control systems which are able to regulate systems characterized by hundreds or even thousands of states. Although improving the efficiency of the LMI- or SOS-based solvers offers one solution, we recommend that the NCS community not only focuses on developing techniques which have low levels of conservatism but also focuses on developing techniques which also have low levels of computational complexity. Hence, we are stressing the importance to not only decentralize the controllers themselves, but also to decentralize or distribute the computational conditions which are used to design them.

Reducing the conservatism in the stability analysis for time-varying transmission intervals and delays: We have observed that an experimentally determined bounding set for time-varying transmission intervals and time-varying delays was, in fact, only a subset of the corresponding bounding set that was commonly assumed in the NCS literature, see, e.g. the recent results [38,58]. Therefore, using this commonly assumed bounding set will introduce conservatism in the stability analysis as it will include additional uncertainties which

do not actually occur in practice. To reduce this type of conservatism in the analysis, analysis techniques to be developed in the future must be able to directly analyze arbitrarily shaped sets that bound the transmission intervals and delays. In this thesis, we already showed that the gridding-and-norm bounding (GNB) technique (employed in Chapter 3) and the SOS technique (developed in Chapter 2) can easily be adapted to incorporate these arbitrarily shaped sets in a non-conservative manner. However, how to do this for other (existing) methods is presently unclear.

Improving dropout modeling and related NCS stability analysis: We have experimentally observed in Chapter 5 that when a packet dropout occurs, a perturbation of both the transmission interval and the transmission sequence can occur. Hence, modeling dropouts as only a prolongation of the transmission interval (as done in the current version of the toolbox) does not capture the dropout modes which also induce a perturbation in the transmission sequence. Therefore a more accurate dropout modeling approach is needed such that the robustness for controllers that are implemented in wireless networks (such as ones in the experimental setup) can be analyzed in the presence of a traffic-congested wireless medium.

Extending the toolbox functionality: The prototype toolbox developed currently only includes robust stability *analysis* tools. This was done according to a modular software structure in order to create a platform onto which functionality can easily be added. Incorporating the additional functionality of robust controller *synthesis* and some *stochastic analysis* techniques should be pursued and is relatively straightforward due to the modular software structure of the toolbox. Additionally, extending the toolbox to analyze delays which are longer than the transmission interval is recommended. Once implemented, the aforementioned extensions should also be tested on experimental setups such as the wirelessly controlled inverted pendulum setup.

Continuing the experimental exploration: The insights gained from the experimental setup that was constructed as a result of this thesis clearly demonstrate the considerable number of benefits that experimental exploration offers. We strongly encourage the NCS community to construct similar (but different) experimental setups in order to investigate, verify and validate many other (common) network modeling assumptions and theoretical results. The resulting collection of such experimental work will help to steer the NCS community towards developing theory that is more readily applicable to practical NCSs.

6.3 Final Thoughts

One day, reliable control systems will be effortlessly and inexpensively implemented by using existing (uncertain) telecommunication infrastructures which are shared by other applications. So far, in the first decade of the twenty-first century, the NCS community has been very successful in achieving swift theoretical progress towards achieving this goal. As an example, today, the maximum allowable transmission interval (MATI) able to be proven for the benchmark batch reactor example has been improved by an impressive 650,000%, see [38], compared to the robustness margins able to be proven in the pioneering work [122]. Although NCS theory is much stronger than just a decade ago, it is far from achieving the 'ultimate' goal of solving large-scale control system design problems. There are, of course, many paths which can be taken to arrive at this goal. At this point in time, the most natural direction to proceed seems to be that of combined experimental validation and inspired theoretical progress.

Fortunately, the construction of experimental NCS setups is a recently growing trend within the field. Research groups which build experimental setups have the advantage of experimentally validating their network models and theoretical results. However, there are many other research groups which have results which could also benefit from experimental validation. Collaboration between research groups that have experimental setups and research groups that would benefit from having their theory experimentally validated is eased when the theory is accessible in the form of a toolbox. Given the amount of theoretical breakthroughs over the past decade, now is the time to increase collaboration within the community by developing software which can be easily applied to different experimental setups. Such collaboration will result in identifying which network assumptions work well in practice for different network settings and which analysis techniques are most effective, thereby, paving an experimentally-driven path to identify the most promising research directions and gear new (theoretical) advances accordingly. The success of building such an experimentally-driven research path will grab the attention of high-tech companies who will then, in turn, also invest in NCS research, until, ultimately, the design of a reliable NCS becomes a common engineering practice.

Bibliography

[1] A.T. Al-Hammouri, M.S. Branicky, V. Liberatore, and S.M. Phillips. Decentralized and dynamic bandwidth allocation in networked control systems. In *20th Int. Parallel and Distributed Processing Symposium*, page 8 pp., April 2006.

[2] A. Alvarez Aguirre. *Remote Control and Motion Coordination of Mobile Robots*. Ph.D. Thesis, University of Technology Eindhoven, Eindhoven, The Netherlands, Sept. 2011.

[3] B.D.O. Anderson and J. Moore. Time-varying feedback laws for decentralized control. *IEEE Trans. Autom. Control*, 26(5):1133–1139, October 1981.

[4] D. Antunes, W.M.P.H. Heemels, J.P. Hespanha, and C. Silvestre. Scheduling measurements and controls over networks - part I: Rollout strategies for protocol design. In *Proc. IEEE American Control Conf.*, Jun. 2012.

[5] D. Antunes, J.P. Hespanha, and C. Silvestre. Stochastic hybrid systems with renewal transitions. In *Proc. IEEE American Control Conf.*, pages 3124 –3129, jul. 2010.

[6] D. Antunes, J.P. Hespanha, and C. Silvestre. Stochastic networked control systems with dynamic protocols. In *Proc. IEEE American Control Conf.*, pages 1686 –1691, jul. 2011.

[7] D. Antunes, J.P. Hespanha, and C. Silvestre. Volterra integral approach to impulsive renewal systems: Application to networked control. *IEEE Trans. Autom. Control*, 57(3):607 –619, march 2012.

[8] J. Araujo, A. Anta, M. Mazo, J. Faria, A. Hernandez, P. Tabuada, and K.H. Johansson. Self-triggered control over wireless sensor and actuator networks. In *Proc. Int. Conf. Distributed Computing in Sensor Systems & Workshops*, pages 1 –9, june 2011.

[9] J. Arauz and P. Krishnamurthy. Markov modeling of 802.11 channels. In *Proc. IEEE Vehicular Technology Conf.*, volume 2, pages 771 – 775, oct. 2003.

[10] K.J. Åström. *An Introduction to Stochastic Control Theory*. Dover Publications Inc., Mineloa, N.Y., 1970.

[11] M. Athans. The role and use of the stochastic linear-quadratic-gaussian problem in control system design. *IEEE Trans. Autom. Control*, 16(6):529 – 552, dec 1971.

[12] J. Baillieul and P.J. Antsaklis. Control and communication challenges in networked real-time systems. *Proc. IEEE*, 95(1):9 –28, jan. 2007.

[13] L. Bakule. Decentralized control: An overview. *Annual Reviews in Control*, 32(1):87–98, 2008.

[14] N.W. Bauer, M.C.F. Donkers, N. van de Wouw, and W.P.M.H. Heemels. Decentralized observer-based control via networked communication. In *submitted*.

[15] N.W. Bauer, M.C.F. Donkers, N. van de Wouw, and W.P.M.H. Heemels. Decentralized static output-feedback control via networked communication. In *Proc. IEEE American Control Conf.*, pages 5700–5705, June 2012.

[16] N.W. Bauer, P.J.H. Maas, and W.P.M.H. Heemels. Stability analysis of networked control systems: A sum of squares approach. In *Proc. IEEE Conf. Decision & Control*, pages 2384 –2389, December 2010.

[17] N.W. Bauer, P.J.H. Maas, and W.P.M.H. Heemels. Stability analysis of networked control systems: A sum of squares approach. *Automatica*, 48(8):1514 – 1524, 2012.

[18] N.W. Bauer, S.J.L.M. van Loon, M.C.F. Donkers, N. van de Wouw, and W.P.M.H. Heemels. Networked control systems toolbox: Robust stability analysis made easy. In *Proc. 3rd IFAC Workshop on Distributed Estimation & Control in Networked Systems*, pages 55–60, June 2012.

[19] G. Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE J. Selected Areas in Communications*, 18(3):535 –547, mar. 2000.

[20] F. Blaabjerg, R. Teodorescu, M. Liserre, and A.V. Timbus. Overview of control and grid synchronization for distributed power generation systems. *IEEE Trans. Industrial Electronics,*, 53:1398–1409, October 2006.

[21] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, volume 15 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, June 1994.

[22] D. Carnevale, A. R. Teel, and D. Nešić. A Lyapunov proof of an improved maximum allowable transfer interval for networked control systems. *IEEE Trans. Autom. Control*, 52(5):892–897, May 2007.

[23] G. Cembrano, G. Wells, J. Quevedo, R. Prez, and R. Argelaguet. Optimal control of a water distribution network in a supervisory control system. *Control Engineering Practice*, 8:1177–1188, 2000.

[24] A. Cervin, D. Henriksson, B. Lincoln, K.E. J. Eker, and Årzén. How does control timing affect performance? Analysis and simulation of timing using Jitterbug and TrueTime. *IEEE Control Systems Mag.*, 23(3):16 –30, June 2003.

[25] A. Chaillet and A. Bicchi. Delay compensation in packet-switching networked controlled systems. In *Proc. IEEE Conf. Decision & Control*, pages 3620–3625, December 2008.

[26] G. Chesi. LMI techniques for optimization over polynomials in control: A survey. *IEEE Trans. Autom. Control*, 55(11):2500–2510, November 2010.

[27] O. Chipara, C. Lu, T.C. Bailey, and G. Roman. Reliable clinical monitoring using wireless sensor networks: experiences in a step-down hospital unit. In *Proc. ACM Conf. Embedded Networked Sensor Systems*, SenSys '10, pages 155–168, New York, NY, USA, 2010. ACM.

[28] 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*, 46(10):1584–1594, 2010.

[29] M.B.G. Cloosterman, N. van de Wouw, W.P.M.H. Heemels, and H. Nijmeijer. Stability of networked control systems with uncertain time-varying delays. *IEEE Trans. Autom. Control*, 54(7):1575–1580, July 2009.

[30] A. Cunha, A. Koubaa, R. Severino, and M. Alves. Open-ZB: an open-source implementation of the IEEE 802.15.4/ZigBee protocol stack on TinyOS. In *Proc. IEEE Int. Conf. Mobile Adhoc & Sensor Systems*, pages 1 – 12, 2007.

[31] R. Cunha, A. Severino, N. Pereira, A. Koubaa, and M. Alves. Zigbee over TinyOS: Implementation and Experimental Challenges. In *Proc. 8th Portuguese Conf. Autom. Control*, Apr. 2008.

[32] J. Daafouz, M.D. Di Benedetto, V.D. Blondel, G. Ferrari-Trecate, L. Hetel, M. Johansson, A.L. Juloski, S. Paoletti, G. Pola, E. de Santis, and R. Vidal. In J. Lunze and F. Lamnabhi-Lagarrigue, editors, *Handbook of Hybrid Systems Control: Theory, Tools, Applications*. Cambridge University Press, Princeton, New Jersey, 2009.

[33] J. Daafouz, P. Riedinger, and C. Iung. Stability analysis and control synthesis for switched systems: A switched Lyapunov function approach. *IEEE Trans. Autom. Control*, 47(11):1883–1887, November 2002.

[34] D.B. Dačić and D. Nešić. Quadratic stabilization of linear networked control systems via simultaneous protocol and controller design. *Automatica*, 43(7):1145–1155, 2007.

[35] D.B. Dačić and D. Nešić. Observer design for wired linear networked control systems using matrix inequalities. *Automatica*, 44(11):2840–2848, 2008.

[36] M. D. Di Benedetto, A. D. Innocenzo, C. Fischione, A. J. Isaksson, K.H. Johansson, S. I Niculescu, S. Olaru, G. Sandou, F. Santucci, E. Serra, S. Tennina, U. Tiberi, and E. Witrant. Wireless ventilation control for large-scale systems: The mining industrial case. *Int. J. Robust & Nonlinear Control*, 20:387–411, 2010.

[37] M.C.F. Donkers, W.P.M.H. Heemels, D. Bernardini, A. Bemporad, and V. Shneer. Stability analysis of stochastic networked control systems. *Automatica*, 48(5):917 – 925, 2012.

[38] M.C.F. Donkers, W.P.M.H. Heemels, N. van de Wouw, and L. Hetel. Stability analysis of networked control systems using a switched linear systems approach. *IEEE Trans. Autom. Control*, 56(9):2101–2115, Sep. 2011.

[39] D.M. Doolin and N. Sitar. Wireless sensors for wildfire monitoring. *Smart Structures and Materials*, 5765, 2005.

[40] J. Eker, A. Cervin, and A. Hörjel. Distributed wireless control using bluetooth. In *Proc. IFAC Conf. New Technologies for Computer Control*, nov. 2001.

[41] G.F. Franklin, J.D. Powell, and A. Emami-Naeini. *Feedback Control of Dynamic Systems: Fifth Edition*. Pearson Prentice Hall, New Jersey, 2006.

[42] N.M. Freris, S.R. Graham, and P.R. Kumar. Fundamental limits on synchronizing clocks over networks. *IEEE Trans. Autom. Control,*, 56(6):1352–1364, June 2011.

[43] E. Fridman, A. Seuret, and J.P. Richard. Robust sampled-data stabilization of linear systems: an input delay approach. *Automatica*, 40(8):1441–1446, 2004.

[44] H. Fujioka. Stability analysis for a class of networked/embedded control systems: A discrete-time approach. In *Proc. IEEE American Control Conf.*, pages 4997–5002, June 2008.

[45] H. Gao, T. Chen, and J. Lam. A new delay system approach to network-based control. *Automatica*, 44(1):39–52, 2008.

[46] H. Gao, X. Meng, T. Chen, and J. Lam. Stabilization of networked control systems via dynamic output-feedback controllers. *SIAM J. Control and Optimization*, 48(5):3643–3658, 2010.

[47] M. Garcia-Rivera and A. Barreiro. Analysis of networked control systems with drops and variable delays. *Automatica*, 43(12):2054–2059, 2007.

[48] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler. The nesc language: A holistic approach to networked embedded systems. In *Proc. Conf. Programming Language Design & Implementation*, PLDI '03, pages 1–11, New York, NY, USA, 2003. ACM.

[49] J.C. Geromel, J. Bernussou, and P.L.D. Peres. Decentralized control through parameter space optimization. *Automatica*, 30(10):1565–1578, 1994.

[50] R.H. Gielen, S. Olaru, M. Lazar, W.P.M.H. Heemels, N. van de Wouw, and S.-I. Niculescu. On polytopic inclusions as a modeling framework for systems with time-varying delays. *Automatica*, 46(3):615 – 619, 2010.

[51] R. Goebel, R. Sanfelice, and A. Teel. Hybrid dynamical systems. *IEEE Control Systems Mag.*, 29(2):28–93, April 2009.

[52] R. Goebel, Ricardo G. Sanfelice, and Andrew R. Teel. *Hybrid Dynamical Systems: Modeling, Stability and Robustness*. Princeton University Press, 2012.

[53] V. Handziski, J. Polastre, J.H. Hauer, C. Sharp, A. Wolisz, and D. Culler. Flexible hardware abstraction for wireless sensor networks. In *Proc.* 2^{nd} *European Workshop on Wireless Sensor Networks*, pages 145 – 157, jan.-2 feb. 2005.

[54] F. Hao and X. Zhao. Linear matrix inequality approach to static output-feedback stabilisation of discrete-time networked control systems. *IET Control Theory Applications*, 4(7):1211–1221, July 2010.

[55] J.H. Hauer and A. Wolisz. An IEEE 802.15.4 MAC Implementation for TinyOS 2. Technical Report JN-AN-1035, Technical University Berlin - Telecommunication Networks Group, 2009.

[56] W.P.M.H. Heemels, B. de Schutter, J. Lunze, and M. Lazar. Stability analysis and controller synthesis for hybrid dynamical systems. *Philosophical Trans. of the Royal Society A*, 368:4937–4960, 2010.

[57] W.P.M.H. Heemels, D. Nešić, A.R. Teel, and N. van de Wouw. Networked and quantized control systems with communication delays. In *Proc. IEEE Conf. Decision & Control*, pages 7929–7935, December 2009.

[58] W.P.M.H. Heemels, A.R. Teel, N. van de Wouw, and D. Nešić. Networked control systems with communication constraints: Tradeoffs between transmission intervals, delays and performance. *IEEE Trans. Autom. Control*, 55(8):1781–1796, August 2010.

[59] W.P.M.H. Heemels, N. van de Wouw, R.H. Gielen, M.C.F. Donkers, L. Hetel, S. Olaru, M. Lazar, J. Daafouz, and S. Niculescu. Comparison of overapproximation methods for stability analysis of networked control systems. In *Proc. Int. Conf. Hybrid Systems: Computation & Control*, pages 181–190, New York, NY, USA, 2010. ACM.

[60] D. Henrion and A. Garulli. *Positive Polynomials in Control*, volume 312 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, Berlin, Heidelberg, 2005.

[61] A. Hernández. *Wireless Inverted Pendulum using IEEE 802.15.4 Protocol*. Master's Degree Project, California Institute of Technology, KTH Royal Institute of Technology, Stockholm, Sweden, April 2011.

[62] J.P. Hespanha. *Linear System Theory*. Princeton University Press, Princeton, New Jersey, 2009.

[63] J.P. Hespanha, P. Naghshtabrizi, and Y. Xu. A survey of recent results in networked control systems. *Proc. IEEE*, 95(1):138–162, January 2007.

[64] L. Hetel, J. Daafouz, and C. Iung. Stabilization of arbitrary switched linear systems with unknown time-varying delays. *IEEE Trans. Autom. Control*, 51(10):1668–1674, October 2006.

[65] L. Hetel, J. Daafouz, and C. Iung. Stabilization of aribitrary switched linear systems with unknown time-varying delays. *IEEE Trans. Autom. Control*, 51(10):1668 – 1674, sept. 2006.

[66] K. Hikichi, H. Morino, I. Arimoto, K. Sezaki, and Y. Yasuda. The evaluation of delay jitter for haptics collaboration over the internet. In *Proc. IEEE Global Telecommunications Conf.*, volume 2, pages 1492 – 1496, nov. 2002.

[67] S.H. Hong. Scheduling algorithm of data sampling times in the integrated communication and control systems. *IEEE Trans. Control Systems Technology*, 3(2):225 –230, jun 1995.

[68] R. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[69] K.E. Johnson and N. Thomas. Wind farm control: Addressing the aerodynamic interaction among wind turbines. In *Proc. IEEE American Control Conf.*, pages 2104–2109, June 2009.

[70] B. Kosucu, K. Irgan, G. Kucuk, and S. Baydere. FireSenseTB: a wireless sensor networks testbed for forest fire detection. In *Proc. Int. Conf. Wireless Communications & Mobile Computing: Connecting the World Wirelessly*, IWCMC '09, pages 1173–1177, New York, NY, USA, 2009. ACM.

[71] A. Koubâa, M. Alvez, and E. Tovar. IEEE 802.15.4 for Wireless Sensor Networks: A Technical Overview. Technical Report HURRAY-TR-050702, IPP-HURRAY! Polytechnic Institute of Porto, 2005.

[72] S. Lall, D. Barcelli, N. van de Wouw, M. Lemmon, H. Ishii, L. Schenato, and P.R. Kumar. In A. Bemporad, W.P.M.H. Heemels, and M. Johansson, editors, *Networked Control Systems*, volume 406 of *Lecture notes in control and information sciences*. Springer-Verlag, 2010.

[73] M. Laurent. Sums of squares, moment matrices and optimization over polynomials. In *Emerging Applications of Algebraic Geometry*, volume 149 of *IMA Volumes in Mathematics and its Applications*, pages 157–270. Springer, 2009.

[74] W. Lawrenz. *CAN System Engineering: From Theory to Practical Applications*. Springer-Verlag, New York, 1997.

[75] P. Levis and D. Gay. *TinyOS Programming*. Cambridge University Press, New York, 2009.

[76] D. Liberzon. *Switching Systems and Control*. Birkhäuser, 2003.

[77] K. Liu and E. Fridman. Stability analysis of networked control systems: A discontinuous Lyapunov functional approach. In *Proc. IEEE Conf. Decision & Control*, pages 1330–1335, December 2009.

[78] J. Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proc. CACSD Conf.*, 2004.

[79] S.J.L.M. van Loon, M.C.F. Donkers, N van de Wouw, and W.P.M.H. Heemels. Stability analysis of networked control systems with periodic protocols and uniform quantizers. In *Proc. IFAC Conf. Analysis & Design of Hybrid Systems*, pages 186–191, 2012.

[80] Jennic Ltd. Application note: Coexistence of IEEE 802.15.4 at 2.4GHz. Technical Report JN-AN-1079, Jennic Ltd., 2008.

[81] Y. Ma, M. Richards, M. Ghanem, Y. Guo, and J. Hassard. Sensors. *Air pollution monitoring and mining based on sensor girld in London*, 8(6):3601–3623, 2008.

[82] P. Mason, M. Sigalotti, and J. Daafouz. On stability analysis of linear discrete-time switched systems using quadratic lyapunov functions. In *Proc. IEEE Conf. Decision & Control*, pages 5629 –5633, dec. 2007.

[83] L. Mirkin. Some remarks on the use of time-varying delay to model sample-and-hold circuits. *IEEE Trans. Autom. Control*, 52(6):1109–1112, June 2007.

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

[85] J.R. Moyne and D.M. Tilbury. The emergence of industrial control networks for manufacturing control, diagnostics, and safety data. *Proc. IEEE*, 95:29–47, 2007.

[86] R.M. Murray, K.J. Åström, S.P. Boyd, R.W. Brockett, and G. Stein. Future directions in control in an information-rich world. *IEEE Control Systems Mag.*, pages 20–33, 2003.

[87] P. Naghshtabrizi and J.P. Hespanha. Designing an observer-based controller for a network control system. In *Proc. IEEE Conf. Decision & Control*, pages 848–853, December 2005.

[88] P. Naghshtabrizi and J.P. Hespanha. Stability of network control systems with variable sampling and delays. In *Proc. of the Annual Allerton Conf. on Communication, Control, & Computing*, 2006.

[89] P. Naghshtabrizi, J.P. Hespanha, and A.R. Teel. On the robust stability and stabilization of sampled-data systems: A hybrid system approach. In *Proc. IEEE Conf. Decision & Control*, pages 4873–4878, December 2006.

[90] P. Naghshtabrizi, J.P. Hespanha, and A.R. Teel. Stability of delay impulsive systems with application to networked control systems. In *Proc. IEEE American Control Conf.*, pages 4899–4904, July 2007.

[91] P. Naghshtabrizi, J.P. Hespanha, and A.R. Teel. Exponential stability of impulsive systems with application to uncertain sampled-data systems. *Syst. Control Lett.*, 57(5):378–385, 2008.

[92] D. Nešić and D. Liberzon. A unified framework for design and analysis of networked and quantized control systems. *IEEE Trans. Autom. Control*, 54(4):732–747, April 2009.

[93] D. Nešić, A. R. Teel, and E. D. Sontag. Formulas relating KL stability estimates of discrete-time and sampled-data nonlinear systems. *Syst. Control Lett.*, 38(1):49–60, 1999.

[94] D. Nešić and A.R. Teel. Input-output stability properties of networked control systems. *IEEE Trans. Autom. Control*, 49(10):1650–1667, 2004.

[95] D. Nešić and A.R. Teel. Input-to-state stability of networked control systems. *Automatica*, 40(12):2121–2128, 2004.

[96] S. Öncü, N. van de Wouw, W.M.P.H. Heemels, and H. Nijmeijer. String stability of interconnected vehicles under communication constraints. In *Proc. IEEE Conf. Decision & Control*, Dec. 2012.

[97] S. Öncü, N. van de Wouw, and H. Nijmeijer. Cooperative adaptive cruise control: Tradeoffs between control and network specifications. In *Proc. IEEE Conf. Intelligent Transportation Systems*, pages 2051 –2056, oct. 2011.

[98] L.Y. Pao and K.E. Johnson. Control of wind turbines: Approaches, challenges, and recent developments. *IEEE Control Systems Mag.*, pages 44–62, April 2011.

[99] A. Papachristodoulou and S. Prajna. A tutorial on sum of squares techniques for systems analysis. In *Proc. IEEE American Control Conf.*, pages 2686–2700 vol. 4, June 2005.

[100] A. Papachristodoulou and S. Prajna. Robust stability analysis of nonlinear hybrid systems. *IEEE Trans. Autom. Control*, 54(5):1035–1041, May 2009.

[101] P.A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. Ph.D. Thesis, California Institute of Technology, Pasadena, California, May 2000.

[102] M. Petrova, J. Riihijärvi, P. Mähönen, and S. Labella. Performance study of IEEE 802.15.4 using measurements and simulations. In *IEEE Wireless Communications and Networking Conf.*, volume 1, pages 487 –492, april 2006.

[103] J. Ploeg, A.F.A. Serrarens, and G.J. Heijenk. Connect and drive: design and evaluation of cooperative adaptive cruise control for congestion reduction. *J. Modern Transportation*, 19(3):207 – 213, 2011.

[104] N.J. Ploplys, P.A. Kawka, and A.G. Alleyne. Closed-loop control over wireless networks. *IEEE Control Systems Mag.*, 24(3):58 – 71, jun 2004.

[105] J. Polastre, R. Szewczyk, and D. Culler. Telos: enabling ultra-low power wireless research. In *Proc. Int. Symposium on Information Processing in Sensor Networks*, pages 364 – 369, april 2005.

[106] I.G. Polushin, P.X. Liu, and C.-H. Lung. On the model-based approach to nonlinear networked control systems. *Automatica*, 44(9):2409–2414, 2008.

[107] R. Postoyan and D. Nešić. A framework for the observer design for networked control systems. In *Proc. IEEE American Control Conf.*, pages 3678–3683, 2010.

[108] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo. *SOSTOOLS: Sum of squares optimization toolbox for MATLAB*, 2004.

[109] Quanser. Single Inverted Pendulum (SIP) Instructor Manuel. Technical Report 513 r4.1, Quansern Inc.

[110] M. Rotkowitz and S. Lall. A characterization of convex problems in decentralized control. *IEEE Trans. Autom. Control*, 50(12):1984–1996, December 2005.

[111] Jr. Sandell, N., P. Varaiya, M. Athans, and M. Safonov. Survey of decentralized control methods for large scale systems. *IEEE Trans. Autom. Control*, 23(2):108–128, April 1978.

[112] P. Seiler and R. Sengupta. An H_∞ approach to networked control. *IEEE Trans. Autom. Control*, 50(3):356 – 364, march 2005.

[113] Y. Shi and B. Yu. Output feedback stabilization of networked control systems with random delays modeled by markov chains. *IEEE Trans. Autom. Control*, 54(7):1668 –1674, july 2009.

[114] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M.I. Jordan, and S.S. Sastry. Kalman filtering with intermittent observations. *IEEE Trans. Autom. Control*, 49(9):1453 – 1464, sept. 2004.

[115] J. Skaf and S. Boyd. Analysis and synthesis of state-feedback controllers with timing jitter. *IEEE Trans. Autom. Control*, 54(3):652–657, March 2009.

[116] C.E. de Souza and A. Trofino. An LMI approach to stabilization of linear discrete-time periodic systems. *Int. J. Control*, 73(8):696–703, 2000.

[117] S.S. Stanković, D.M. Stipanović, and D.D. Šiljak. Decentralized dynamic output feedback for robust stabilization of a class of nonlinear interconnected systems. *Automatica*, 43(5):861–867, 2007.

[118] J.F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999.

[119] Y.S. Suh. Stability and stabilization of nonuniform sampling systems. *Automatica*, 44(12):3222–3226, 2008.

[120] M. Tabbara and D. Nešić. Input-output stability of networked control systems with stochastic protocols and channels. *IEEE Trans. Autom. Control*, 53(5):1160 –1175, june 2008.

[121] D.D. Šiljak. *Decentralized control of complex systems*. Academic Press, Boston, 1991.

[122] G.C. Walsh, H. Ye, and L.G. Bushnell. Stability analysis of networked control systems. *IEEE Trans. Control Systems Technology*, 10(3):438–446, May 2002.

[123] N. van de Wouw, P. Naghshtabrizi, M.G.B. Cloosterman, and J.P. Hespanha. Tracking control for sampled-data systems with uncertain time-varying sampling intervals and delays. *Int. J. Robust & Nonlinear Control*, 20(4):387–411, 2009.

[124] N. Xu, S. Rangwala, K.K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor network for structural monitoring. In *Sensys '04*, pages 13–24. ACM Press, 2004.

[125] M. Yu, L. Wang, and T. Chu. Sampled-data stabilisation of networked control systems with nonlinearity. *IEEE Proc. Control Theory and Applications*, 152(6):609–614, November 2005.

[126] L. Zhang and D. Hristu-Varsakelis. Communication and control co-design for networked control systems. *Automatica*, 42(6):953–958, 2006.

[127] L. Zhang, Y. Shi, T. Chen, and B. Huang. A new method for stabilization of networked control systems with random delays. *IEEE Trans. Autom. Control*, 50(8):1177–1181, August 2005.

[128] W. Zhang, M.S. Branicky, and S.M. Phillips. Stability of networked control systems. *IEEE Control Systems*, 21(1):84–99, feb 2001.

[129] Y. Zhu and P.R. Pagilla. Decentralized output feedback control of a class of large-scale interconnected systems. *IMA J. Mathematical Control and Information*, 24(1):57–69, March 2007.

Summary

Networked Control Systems
From Theory to Experiments

Driven by recent telecommunication demand, shared communication channels are more abundant today than ever before. These shared wired/wireless networks for communication are being exploited more and more in control systems to connect sensors, controllers and actuators resulting in so-called networked control systems (NCSs). NCSs replace the more traditional control systems where dedicated point-to-point (wired) connections are being used. The advantages of using NCS technology are inexpensive and easily modifiable communication links which allow control algorithms to be easily implemented in situations where dedicated connections are not possible (either economically or physically). However, the drawback is that the control system is susceptible to undesirable (possibly destabilizing) side-effects such as time-varying transmission intervals, time-varying delays, packet dropouts, quantization and a shared communication medium. These network-induced effects undermine fundamental assumptions on which traditional control theory is built and, therefore, it is essential to develop new techniques and tools that can be used to analyze and/or design control systems which communicate via a shared network. This thesis advances NCS analysis and design methodologies by contributing new theoretical developments, new software tools and new experimental validation results.

The first new theoretical development is in the area of analyzing robust stability properties with respect to network-induced effects. A sum of squares (SOS) approach for a class of nonlinear NCSs incorporating bounded time-varying delays, bounded time-varying transmission intervals and a shared communication medium is developed. Mathematical models that describe these nonlinear NCSs are cast into suitable hybrid system formulations. Based on these hybrid system formulations, (families of) Lyapunov functions are constructed using SOS techniques. Amongst other benefits, it is shown that this technique improves the

guaranteed robustness margins compared to existing work.

The second new theoretical development concerns the design of network-aware decentralized controllers guaranteeing robust stability properties with respect to network-induced effects. We develop one of the first approaches based on semidefinite programming techniques to synthesize stabilizing decentralized observer-based output-feedback controllers for linear plants where the controllers, sensors and actuators are connected via a shared communication network subject to time-varying transmission intervals and delays. To effectively deal with the shared communication medium, a switched observer structure is adopted that switches based on the transmitted measured outputs and a switched controller structure is also adopted that switches based on the transmitted control inputs at each transmission time. By taking a discrete-time switched linear system perspective on modeling these decentralized NCSs, we are able to derive a general model that captures all these networked and decentralized control aspects. We provide linear matrix inequality (LMI)-based synthesis conditions which, if satisfied, provide stabilizing observer-based controllers that are both decentralized and robust to network effects.

Regarding new software tools, the first prototype of a toolbox is developed to automate (robust) stability analysis (and controller design) for NCSs. Specifically, it is shown that the toolbox can be employed to efficiently verify if a linear time-invariant (LTI) plant and an LTI controller interconnected with a shared network are robust to certain network imperfections. The main intention of the toolbox is to make the available theory readily accessible to and applicable for the general control community. Additionally, the chosen software structure enables the incorporation of custom models or custom stability/performance analysis conditions in an easy manner, thereby allowing the control community to contribute and to further develop the toolbox.

Finally, an experimental case study involving a wirelessly controlled inverted pendulum/cart system is investigated. In particular, the communication network itself is analyzed such that the network-induced effects can be characterized in terms of bounds on the transmission intervals and transmission delays. Based on these bounds, the prototype toolbox is applied to analyze the robustness regions for different performance specifications, which aid in tuning the controller to achieve more closed-loop robustness with respect to the network-induced effects. This leads to a validation of the developed theory in an experimental setting. In addition to the validation of the developed theory, many new insights into the network behavior are obtained and explained, thereby raising new interesting questions for future research on NCSs.

Acknowledgements

Four years ago, I was given the opportunity to embark on an adventure. Like any good adventure, the need to travel to a land far far away and being confronted with many new challenges were required. I made the decision to go on this adventure primarily based on a good feeling. Although this might not have been the most sound reasoning, looking back, I can confidently say this was the best decision I have ever made. As a result of this journey, I have grown to understand mathematics and science better, to understand myself better and to understand the world better. I owe many thanks to the people who have made this journey possible and the people who have made the outcome successful by providing their continuous support.

First, I would like to thank my advisor Maurice Heemels. Maurice, your positive attitude has always motivated me to get back up and keep trying when I fell (which believe me, was quite often). You have not only been a great scientific advisor, but also a great mentor and for that I feel truly blessed. Thank you for your exemplary enthusiasm and constant support throughout my journey.

Next, I would like to thank my co-advisor Nathan van de Wouw. Nathan, your calm personality has made every meeting enjoyable and your critical eye for detail has helped refine mine tremendously. I have truly enjoyed working with you and have learned many valuable lessons and skills that I will carry with me throughout my career.

I would like to especially thank Andrew Teel for guiding me during my Master's study and being a large part of the reason that I have produced this thesis.

I would also like to thank Jamal Daafouz, Mikael Johansson and Siep Weiland for being a part of the reading committee and for giving detailed comments on the draft version of my thesis.

In addition to my own blood, sweat and tears, this thesis contains the hard work of two former Master's students, Paul Maas and Bas van Loon. These two guys were a pleasure to work with and really did great work towards making this thesis possible.

The WIDE project team has also played a large role in the production of this thesis. I would like to thank Vicenç Puig, Juli Romera and Diego Garcia for hosting me during the project experiments in Barcelona. To Davide Barcelli, thanks for the pleasant discussions we have had during our work in Barcelona. Also, I would like to thank Alberto Bemporad, Valdimir Havlena and Mikael Johansson, working alongside you on the project was a honor and a pleasure.

To all the CST, DCT and HNS colleagues at the TU/e, you have made a city half way around the world feel more like home. A special thanks to Menno Lauret and Sinan Öncü. I could always count on you two to impulsively have a few craft brews, accompanied with an interesting conversation or two, which really helped to keep my sanity. To the HNSers, Bas van Loon, Tom Gommans and Duarte Antunes, sharing a few laughs with you guys throughout the day (and many evenings) made work really enjoyable. Tijs Donkers, thanks for all your guidance during the first few years of my study, both scientifically and (more importantly) socially. To my roomies (past and present), Chris Criens, Emmanuel Feru, Menno Lauret, Bart Hennen, and Dennis van Raaij, thanks for the random conversations through out the day to break the monotony. Finally, I can't forget Matthijs van Berkel, who was always there offering his refreshingly unique perspective on many issues to keep things interesting.

A special thanks to my 'extracirricular' friends. To the snowboard crew, Ewout van der Laan, Willem-Jan Evers, Wouter Aangenent, Arjen den Hamer, Thijs van Keulen, and Stan van der Meulen, I am super happy that I was able to partake in the 'epic' trips we have had each and every year. To the basketball crew, Andelko, Sergey, Koos, Valdamir, Nandra, (and sometimes Robbert and Sava) playing with you every weekend always resulted in a bruise or two but was always a lot of fun.

Finally, to my support system back home that without I would not have made it this far. Mom and dad, I guess this thesis is a pretty good sign that all your hard work and loving support for these past twenty-nine years has paid off. Courtney and Curtis, you guys have always been there for me and you both mean the world to me. Kurt, Ray, Christine, Kerry, Angelica, Josh and Diana, you guys know that you are also a large part of what I consider my family and I can't express in words how much I enjoy just hanging out with you guys (even if it has been only a few times each year). Thanks for always being there for me.

As this adventure comes to an end, I know that the support I have been fortunate enough to receive will give me the confidence to carry me through my next adventure, wherever that may be.

Nick Bauer
February 2013

Curriculum Vitae

Nick Bauer was born on October 2, 1983 in San Francisco, CA, USA.

He received his Bachelor of Science (Honors) and Master of Science degrees from the department of Electrical and Computer Engineering at the University of California, Santa Barbara, in 2007 and 2008, respectively. During his Master of Science study, he worked as an intern at Motion Engineering Inc. (MEI).

Since January 2009, he started his PhD project within the Hybrid and Networked Systems group of the department of Mechanical Engineering at the Eindhoven University of Technology, the Netherlands, under the guidance of Maurice Heemels and Nathan van de Wouw. His research project was part of the EU project WIDE, entitled "Decentralized and Wireless Control of Large-Scale Systems". The results of his research are printed in this thesis.